МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий
(наименование института полностью)

Кафедра «Прикладная математика и информатика»
(наименование)

01.03.02 Прикладная математика и информатика
(код и наименование направления подготовки)

Компьютерные технологии и математическое моделирование
(направленность (профиль) / специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему «Идентификация объектов на основе данных, получаемых с системы видеонаблюдения»

Обучающийся М.Д. Вершинин
(Инициалы Фамилия) (личная подпись)

Руководитель к.т.н., доцент, А.Б. Кузьмичёв
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Консультант к.п.н., доцент, Т.С. Якушева
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Тольятти 2022

Аннотация

Тема выпускной квалификационной работы «Идентификация объектов на основе данных, получаемых с системы видеонаблюдения».

В выпускной квалификационной работе исследуется вопрос разработки и применения алгоритмов идентификации объектов на видео в контексте видеонаблюдения на парковке.

Цель выпускной квалификационной работы - разработать алгоритм идентификации объектов на данных с системы видеонаблюдения.

Объект исследования - процесс идентификации объектов на видеоматериале.

Предмет исследования - это программная реализация алгоритма идентификации объектов на данных с системы видеонаблюдения.

В ходе работы выполнены следующие задачи: выбран оптимальный способ распознавания объектов YOLOv3; разработана программная реализация алгоритма идентификации объектов в видеопотоке; программа протестирована; зафиксированы результаты улучшения программы.

Во введении описаны актуальность проводимого исследования, цели и задачи на исследование.

В первом разделе рассмотрены методы идентификации объектов, произведён выбор объектов для идентификации, области применения идентификации объектов, изложена информация о камерах видеонаблюдения, поставлена задача на разработку.

Во втором разделе информация о преимуществах и недостатках методов идентификации объектов на видео, выборе метода для идентификации, сформировано математическое и алгоритмическое описание.

В третьем разделе описана реализации программы и её тестирование.

В заключении представлены выводы по проделанной работе.

В работе присутствуют 55 страниц, 27 рисунков, 24 формулы, 12 таблиц.

Abstract

The title of the graduation work is « Identification of objects based on data obtained from a video surveillance system».

The senior paper consists of an introduction, three parts, a conclusion, 12 tables, a list of references including foreign sources.

The key issues of the thesis are the design and implementation of an algorithm for identifying objects based on data obtained from a video surveillance system.

The aim of the work is development of an algorithm for identifying objects on data obtained from a video surveillance system. The object of the graduation work is the process of identifying objects on video material. The subject of the graduation work is a software implementation of an object identification algorithm based on data from a video surveillance system.

The graduation work may be divided into several logically connected parts which are the following: theoretical justification of the task of identifying objects in a video stream; designing an object identification algorithm; software implementation of object identification based on data from a video surveillance system.

Finally, we present the work on the successful implementation of the developed algorithm for identifying objects in the video stream. The main part of the algorithm is using object detection method YOLOv3 to get object data from frames. The developed algorithm makes it possible to identify objects at a speed close to real time. The results of the study showed that implemented algorithm had a positive impact on the speed of the program.

In conclusion we'd like to stress this work it is relevant not only in providing more data for video analytics, but also in further research of identification algorithms in the video stream.

Оглавление

Введение

Идентификация объектов на видеоконтенте и изображениях является актуальной задачей в настоящее время. Данная задача из сферы компьютерного зрения может быть решена для видеонаблюдения. Идентификацию объектов на данных, получаемых с системы видеонаблюдения, можно использовать как на уже сохранённых видеозаписях, так и в режиме реального времени. Определение объектов на видеоматериалах сводится к распознаванию объектов на изображении.

В настоящее время методы распознавания объектов используются во многих системах видеонаблюдения, поэтому метод идентификации объектов на данных с системы видеонаблюдения оптимально разрабатывать на основе метода распознавания. Для идентификации объектов в видеопотоке актуальной задаче является сохранение уникальных данных объекта между кадрами.

Объектом исследования является процесс идентификации объектов на основе данных, получаемых с системы видеонаблюдения. Предмет исследования – это программная реализация алгоритма, которая идентифицирует объекты на данных, получаемых с системы видеонаблюдения. Целью выпускной квалификационной работы является разработать алгоритм идентификации объектов на данных с системы видеонаблюдения.

Для достижения указанной цели необходимо выполнить следующие задачи:

- изучить способы идентификации объектов на данных с камер видеонаблюдения с применением распознавания;
- выбрать оптимальный метод распознавания объектов на данных с системы видеонаблюдения;
- разработать алгоритм идентификации объектов на данных, получаемых с системы данных видеонаблюдения;

- разработать программную реализацию разработанного алгоритма идентификации объектов на данных видеонаблюдения;
- протестировать работоспособность реализации метода идентификации объектов в видеопотоке;
- разработать дополнения алгоритма для повышения скорости работы;
- проанализировать полученные результаты;
- сформировать заключение по проделанной работе.

ВКР имеет следующую структуру: введение, три главы, заключение, список использованной литературы.

Во введении затронута предметная область видеонаблюдения, показана актуальность темы. Сформулированы объект и предмет исследования, а также поставлена цель выпускной квалификационной работы. Определены задачи, выполнение которых сопутствует достижению поставленной цели.

В первой главе поставлена задача идентификации, произведён выбор объектов для идентификации, а также определена система видеонаблюдения.

Во второй главе происходит сравнение методов идентификации, описана алгоритмическая и математическая реализация выбранной модели, а также описание разработанного алгоритма идентификации.

В третьей главе продемонстрирована программная реализация идентификации объектов с применением метода распознавания YOLOv3. Проведено тестирование программы, показано увеличение скорости работы.

Таким образом, идентификация объектов на видеоматериалах, получаемых с системы видеонаблюдения является актуальной темой по сей день. Была установлена цель бакалаврской работы и поставлены задачи для достижения этой цели.

1 Теоретическое обоснование задачи идентификации объектов в видеопотоке

1.1 Процесс идентификации объектов в системах видеонаблюдения

Под идентификацией понимается присвоение объектам уникальных значений, выступающих идентификаторами. Во время идентификации происходит заполнение следующих данных для объекта:

- ограничивающая рамка объекта на кадре;
- принадлежность объекта классу;
- значение объектности;
- уникальный в рамках кадра идентификатор.

Частью идентификации объекта является его локализация на изображении. Локализованный объект занимает прямоугольную область, заключённую в ограничивающую рамку. Методы определения местонахождения объекта приведены в таблице 1.

Таблица 1 – методы локализации объекта на изображении

Номер метода	Название метода	Описание метода	Преимущества	Недостатки
1	Метод градиентов	Использование функции яркости с расчётом градиента	Высокая скорость вычислений.	Высокая чувствительность к шуму.
2	Пересечения нулевого уровня	Вычисление вторых производных, контур определяется при переходе через границу яркости	инвариантность к вращению, низкая вычислительная сложность	Высокая чувствительность к шуму, ложные границы контура.
3	Метод Марра-Хилдрет	Переход границы яркости в определённом пикселе	Высокая скорость работы	Ложные границы контура, разрывы контура.

Продолжение таблицы 1

4	Детектор границ Кэнни	Сглаживание, вычисление градиентов, подавление немаксимумов, двойная пороговая фильтрация, трассировка области неоднозначности	Хорошее определение границ в шуме, высокая точность локализации	Ресурсоёмкость, невысокая скорость работы.
5	ISEF детектор	Свёртка исходного изображения, вычисление аппроксимации лапласиана, Вычисление бинарного изображения лапласиана, подавление ложных пересечений, адаптивная пороговая фильтрация, трассировка области неоднозначности	высокая точность определения границ	Чувствительность к шуму, высокая вычислительная сложность
6	Активные контуры	расчёт отклонения исходного контура, минимизация функции энергии для точки контура	Устойчивость к шуму, работа с контуром как с последовательностью точек	высокая вычислительная сложность, начальное приближение
7	Методы на основе вейвлет-преобразования	Итерационное применение декомпозирующего вейвлет-фильтра	Эффективное выделение на полутоновом изображении	Сложность реализации и настройки
8	Методы на основе нечёткой логики	Преобразование изображения к нечёткому множеству, заполнение степени принадлежности пикселя в зависимости от соседних, преобразование в чёткие значение яркости.	Устойчивость к шуму, возможность объединения с другими методами	Низкая адаптивность под разные изображения
9	Методы на основе нейронной сети	Выбор контура на основе похожих изображений, на который обучена нейросеть	Получение нескольких границ	Настройка параметров сети, обучение на обучающей выборке[18]

Идентификация объекта может производиться как в два этапа, так и в один. При двухэтапном процессе сначала выделяется регион интереса, который может содержать в себе объект, что и решает задачу локализации. При одноэтапном способе результаты классификации и степень уверенности поступают вместе с координатами ограничивающих рамок [15].

Для идентификации объектов используются методы, которые описаны в таблице 2.

Таблица 2 – Методы идентификации объектов

Номер метода	Название метода	Описание метода
1	Метод наименований	присвоение искомому объекту уникального имени или идентификатора
2	Метод цифровых номеров	проставление объектам порядкового номера или уникального цифрового кода. Также нумерация может снабжаться серийными префиксами
3	Классификационный метод	выделяется подмножество однородных объектов из множества доступных классов
4	Ссылочный метод	позволяет получить указание на объект в некой коллекции значений
5	Описательный метод	работает с самим объектом и занимается описанием его параметров и характеристик
6	Описательно-ссылочный метод	предоставляет как описание основных признаков объекта, так и ссылку на место, где содержатся все характеристики [20]

Идентификация объектов на изображении или в видеопотоке производится при помощи классификационного метода, то есть идентифицированному образу с изображения поставлен в соответствие класс, к которому он относится. Наряду с классификационным применяется метод цифровых номеров. Классификационный метод имеет два способа классификации объектов: фасетный и иерархический. Описание способов классификационного метода идентификации представлено в таблице 3.

Таблица 3 – Способы классификации

Название способа	Описание способа	Преимущества	Недостатки
Иерархический	исходное множество логически разделяется на подмножества, причём одни подмножества могут являться родителями для других. Разделение на подмножества должно производиться только по одному признаку деления	логичность разделения и удобство для ручной и машинной обработки данных	негибкость к разным задачам и невозможность быстро добавлять новые уровни.

Продолжение таблицы 3

Фасетный	работает по разделению исходного множества на подмножества при помощи условий	Гибкость систематизации объектов	Выбор признаков для разделения по условиям. Признаки не должны пересекаться

После определения класса объекта для установления большей информации о нём можно провести аутентификацию в некой системе. Более того, есть возможность измерить параметры классифицированного объекта, либо дополнительно классифицировать экземпляры, из которых он состоит.

Таким образом, поставлена задача идентификации. Объект считается идентифицированным, если для него установлен некий уникальный ключ в виде номера, принадлежности классу, значения объектности. По результатам классификации объект относится к одному из классов. Классы из множества классов подразделяются на иерархические группы. Также для объекта выявляется его местоположение на изображении.

1.2 Выбор объектов идентификации

Чтобы соотнести образ на изображении с существующим объектом, нужно отнести этот образ к одному из классов, каждый из которых в свою очередь может иметь подклассы. Для рассматриваемой области видеонаблюдения на парковке специфика применения системы видеонаблюдения не предполагает наблюдение за редкими объектами, поэтому идентифицируются распространённые особи. Для идентификации выбраны следующие классы: человек, велосипед, автомобиль, мотоцикл, автобус, кошка, собака, рюкзак, сумка, чемодан, Иерархическая классификация для этого множества классов представлена на рисунке 1.

Рисунок 1 – Иерархическая классификация объектов

Данная классификация позволяет подобрать для идентифицированного объекта подходящий класс. Объекты, относящиеся к одним и тем же классам, обладают схожими признаками, которые обрабатываются обучающейся нейронной сетью. Нейронная сеть в свою очередь для обучения требует набор данных, состоящий из многочисленных примеров изображений, содержащих объект поиска. Выделенные для идентифицируемых объектов идентификационные признаки обладают следующими критериями:

- специфичность – наиболее точное и полное отражение свойств объекта;
- воспроизводимость – отображение признака при многократном появлении образа;
- устойчивость – способность признака противостоять изменениям;
- частота – чем реже появляется признак, тем больше оказывает влияния на идентификационную значимость;
- зависимость – степень влияния других признаков;

- класс признака – случайный или закономерный, для группы объектов или для одного, внутренний или внешний, количественный или атрибутивный, собственный или приобретённый [16].

Идентификация производится по изображению с камеры видеонаблюдения, что говорит о том, что образы рассматриваются с определённого ракурса, следовательно, основными признаками для классификации являются внешние атрибуты. Прежде всего важна форма искомого объекта, его строение, количественные признаки, цветовая палитра, размеры. Размеры объекта могут быть отображены через ограничивающую рамку, которая может строиться при помощи ключевых точек объекта.

Таким образом, выбранные для идентификации объекты относятся к сфере видеонаблюдения на парковках. Выбраны следующие классы для идентификации: транспорт, животные, аксессуары. Всего выбрано 10 подклассов, к которым могут относиться объекты.

1.3 Область применения идентификации объектов в видеопотоке

Идентификация объектов на изображении или в видеопотоке применяется в технологии компьютерного зрения, что позволяет автоматизировать процесс видеонаблюдения и снизить участие человека в процессе. Идентификация объектов имеет следующие варианты применения:

- Видеонаблюдение по периметру какого-либо участка – для обеспечения безопасности, обнаружение подозрительных лиц или предметов [14];
- Наблюдение со стороны беспилотного транспорта – определение возможных действий в дорожном или воздушном потоке;
- Сбор распознанных образов людей на входе в супермаркет – добавление счётчика вошедших, возможность выделения постоянных посетителей;

- Видеонаблюдение за животными в сфере животноводства – подсчёт голов скота, наблюдение за скотом во время отсутствия хозяев;
- Видеонаблюдение в лифте для сбора данных о вошедших людях и оставленных предметов – уведомление о находящихся внутри людях во время аварийных ситуаций;
- Выявление и распознавание подозрительных ситуациях в местах скопления людей – для предотвращения возможных угроз безопасности населению [17];
- Видеонаблюдение на сложных участках дороги для обнаружения аварийных ситуаций – возможность оперативно локализовать дорожное происшествие;
- Идентификация брендов и рекламы в торговых центрах – для маркетинга и проверки работы таргетированной рекламы;
- Наблюдение за техникой для определения неисправностей – ускорение исправления неполадки;

Подводя итог, можно сформировать перечень применения идентификации объектов. Основными вариантами применения идентификации объектов являются безопасность объектов, наблюдение за посетителями, помощь в аварийных ситуациях, эффективность маркетинга. Идентификация объектов позволяет собрать больше данных для видеоаналитики в различных сферах делает решение задачи идентификации объектов актуальной по сей день. Для идентификации объектов выбрана область видеонаблюдения на парковке для сбора дополнительных данных для аналитики и безопасности.

1.4 Обзор системы видеонаблюдения для идентификации объектов

Для видеонаблюдения на парковке выбор камер обуславливается тем, находятся парковочные места на улице или в помещении. Камеры могут

располагаться на стенах или потолке, преимущества и недостатки камер в зависимости от расположения описаны в таблице 4.

Таблица 4 – Преимущества и недостатки камер в зависимости от расположения

Расположение камеры	Преимущества	Недостатки
Настенная	Обширная зона наблюдения.	Осложнение видимости при перекрытии идентифицируемого объекта другими.
Потолочная	Точность для определения координат образа в помещении около 75%, на улице 50%. Удобство при подсчёте объектов.	Использование ограниченного набора признаков для идентификации, неудобно контролировать оператору [13].

Для видеонаблюдения можно использовать следующие виды камер:

- Web камеры;
- Аналоговые;
- IP камеры;
- Камеры машинного зрения.

Преимущества и недостатки указанных камер описаны в таблице 5.

Таблица 5 - Преимущества и недостатки камер

Вид камер	Преимущества	Недостатки
Web камеры	Стоит дёшево, можно использовать в домашних условиях, не требует особых навыков в подключении. Применение в локальных тестах.	Низкое разрешение, небольшое число кадров в секунду. Функционирование с персональным компьютером. Неудобство использования на улице. Сами по себе не имеют функций машинного зрения. Ухудшение качества при ослаблении освещения.
Аналоговые	Недорогая цена. Возможность установки как в помещении, так и на улице. Поддержка Full HD. Лёгкая установка и минимальная задержка при работе.	Низкая масштабируемость системы. Сложность в подключении видеоаналитики. Для трансляции или сохранения видео требуется видеорегистратор. Не поддерживается передача данных без провода.

Продолжение таблицы 5

IP камеры	Поддержка высокого качества видео, в том числе и 8K. Возможность осуществлять передачу данных без использования проводов. Удалённая настройка системы. Возможность подключения к серверу множества камер. Удачное решение для бизнес аналитики.	Вероятность небольшого прерывания видеопотока, либо задержки при передаче видео.
Камеры машинного зрения	Высокая скорость работы. Встроенное программное обеспечение с технологиями машинного зрения.	Высокая стоимость. Дорогое обслуживание и стоимость самих камер. Узконаправленность встроенного программного обеспечения. Распознавание более мелких объектов [11].

Исходя из описанных в таблице 5 характеристик камер видеонаблюдения, можно прийти к выводу, что наиболее оптимальным вариантом является установка IP камер. Для небольших объектов с малой численностью камер видеонаблюдения подходят и аналоговые камеры, однако стоит учитывать, что от этого страдает как масштабируемость системы, так и качество видео. Видеопоток с камеры обрабатывается программой для идентификации объектов. Если используется камера с видеорегистратором, то программа для идентификации может обрабатывать видеоряд, предварительно загруженный на жёсткий диск.

Система видеонаблюдения с применением IP камер состоит из следующих элементов:

- IP камеры, передающие сигнал по проводу или без него;
- Видеорегистратор не является обязательным, но позволяет сохранять отснятые видеоматериалы;
- Жёсткий диск для хранения записей с видеорегистратора;
- Кабель;
- POE коммутатор для видеонаблюдения;
- Wi-fi роутер при использовании беспроводных камер;
- Персональный компьютер для управление системой.

Схема подключения IP камеры представлена на рисунке 2 [24].

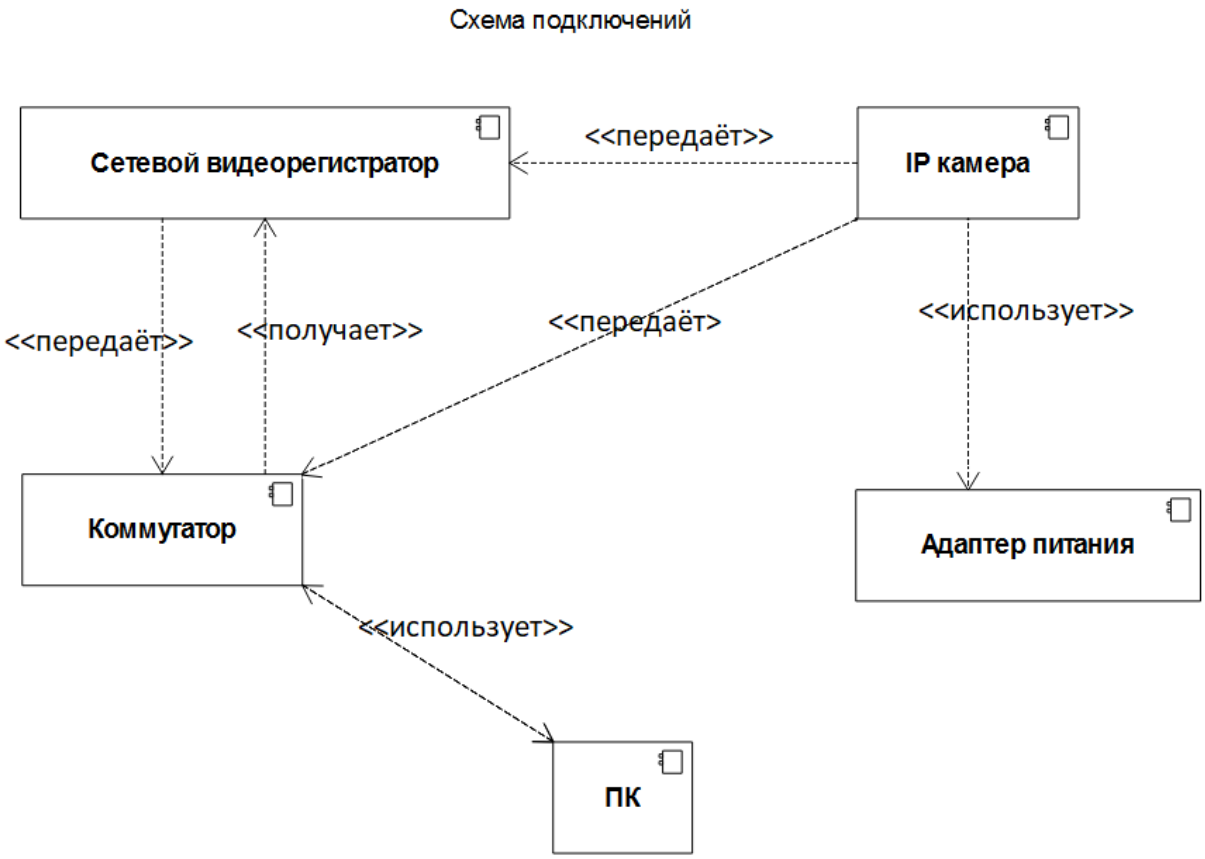

Рисунок 2 – Схема подключения IP камеры

Выбор места для установки камеры видеонаблюдения является важным фактором, так как это может повлиять на результаты идентификации объектов. Важно не только снизить влияние окружающей среды такие, как блики света от искусственного или натурального освещения, помехи в виде листвы деревьев или кустов, но и использовать подходящий угол наклона камеры, который рассчитывается по формуле (1).

$$\alpha = 2*arctg(d/2*F), \quad (1)$$

где

- α – угол обзора объектива в градусах;
- d – размер светочувствительного элемента в миллиметрах;
- F – фокусное расстояние объектива камеры.

Изображение объектива камеры с применением формулы 1 продемонстрировано на рисунке 3.

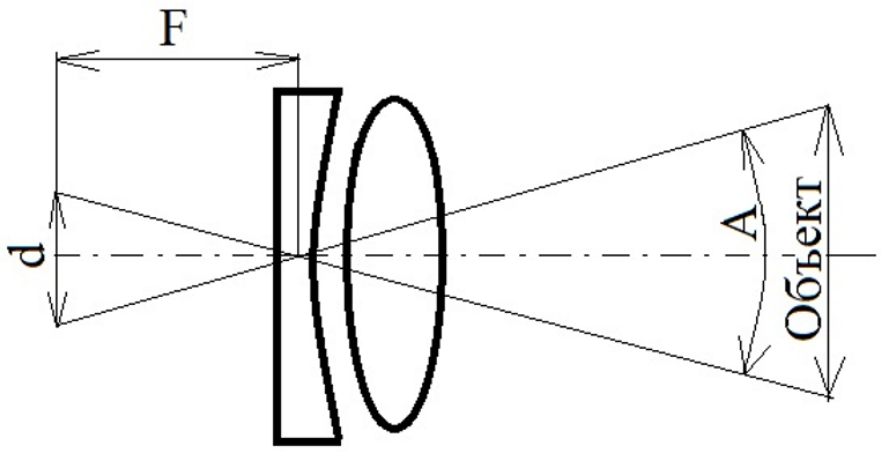

Рисунок 3 - Объектив камеры

Оптимальным углом наклона, который соответствует углу зрения человека, является значение, равное 36 градусов. Фокусное расстояние при этом будем примерно равно 6,9 миллиметров. Соответственно, камеру следует размещать с наклоном в диапазоне от 30 до 40 градусов. В таком случае дальность, на которой производится идентификация объектов, варьируется от 5 до 20 метров [23].

Итак, для системы видеонаблюдения наиболее подходит применение IP камер. Расположение камеры важно для обеспечения оптимального угла наклона и ракурса, которому не препятствуют посторонние объекты. Оптимальный диапазон угла наклона от 30 до 40 градусов, расстояние до объекта рекомендуется соблюдать от 5 до 20 метров.

1.5 Постановка задачи на реализацию программы по идентификации объектов на данных, получаемых с системы видеонаблюдения

Требуется разработать программу, реализующую один из способов идентификации объектов.

Реализованная программа должна обеспечивать следующие функциональные возможности:

- Программа должна поддерживать работоспособность при идентификации объектов как на статичном изображении, так и на видеопотоке;
- Программа должна иметь возможность идентифицировать объекты как в реальном времени, так и на записанном видеоматериале;
- Программная реализация должна идентифицировать несколько объектов в кадре;
- Необходимо выделить идентифицированный объект на изображении;
- Идентифицированные объекты должны сопровождаться значением степени уверенности, что в выделенной области находится указанный объект.

На вход программе может подаваться изображение, видео или видеотрансляция. На выходе реализованная программа должна отобразить ограничивающие рамки идентифицированных объектов, а также степень уверенности.

На основании изложенного выше функциональные требования следует учесть при разработке программной реализации идентификации объектов на данных, получаемых с системы видеонаблюдения.

2 Проектирование алгоритма идентификации объектов

2.1 Сравнение способов идентификации объектов

Для идентификации объектов на изображении или на видео существует несколько способов. Описание данных способов, а также их преимущества и недостатки представлены в таблице А.1 в Приложении А.

Таким образом, описаны основные методы идентификации объектов на основе метода распознавания. Рассмотрено 8 способов, которые имеют различный подход в работе, имеют разные преимущества и недостатки. Для выбора наиболее подходящего метода необходимо сравнить их в разрезе определённых показателей.

2.2 Выбор подходящего метода идентификации объектов

Для выделения наиболее подходящего метода требуется дать оценку по критериям, после чего выбрать метод либо с наибольшей средней оценкой, либо с самыми высокими оценками по наиболее важным параметрам. Оценке подвергаются следующие признаки:

- Время, затраченное на идентификацию – один из основных параметров, определяющий возможность применения метода для идентификации объектов на достаточной для реального времени скорости;
- MAP – средняя точность идентификации – также крайне важна при идентификации объектов;
- FPS – число кадров в секунду;
- Идентификация нескольких объектов на изображении;
- Как много различных объектов может идентифицировать метод без дополнительного обучения.

Наиболее важными критериями при оценке являются mAP и время. Чем выше точность, тем выше оценка. Чем меньше время, тем оценка выше.

Оценки способов идентификации по шкале от 0 до 10 баллов на основе описанных критериев представлены в таблице 6.

Таблица 6 - Оценка методов идентификации

Название метода \ критерий	Время	mAP	FPS	Несколько объектов	Объекты без доп. Обучения	Средняя оценка
Метод формирования штрих-кодов	6	7	5	5	4	5,4
Метод гибкого сравнения на графах	6	7	6	6	5	6
Метод главных компонент	7	8	6	6	5	6,4
Метод Виолы Джонса	7	8	7	8	6	7,2
Метод нечёткой триангуляции Делоне	6	7	6	7	5	6,2
R-CNN	5	8	6	8	7	6,8
R-FCN	8	9	8	5	7	7,4
YOLOv3	9	9	9	9	10	9,2

По итогам оценки методов идентификации из 8 представленных методов наивысшую среднюю оценку 9,2 получил YOLOv3. Данный метод по затраченному на идентификацию времени и точности идентификации является наиболее оптимальным для реализации, так как получил оценки 9. Важным причиной выбора метода оказалась возможность идентифицировать широкий перечень объектов без дополнительного обучения модели.

2.3 Математическое и алгоритмическое описание способа идентификации с использованием YOLOv3

Модель YOLOv3 состоит из следующих фрагментов:

- Экстрактор особенностей;
- Ограничивающая рамка и вывод;
- Представление ограничивающей рамки;
- Сигмоиды для каждого класса;
- Функция потери;

- Многомасштабные прогнозы;
- Пересечение над объединением ограничивающих рамок;
- Фильтрация с помощью подавления немаксимумов [7].

Алгоритм идентификации объектов на изображении с использованием модели YOLOv3 представлен на рисунке 4.

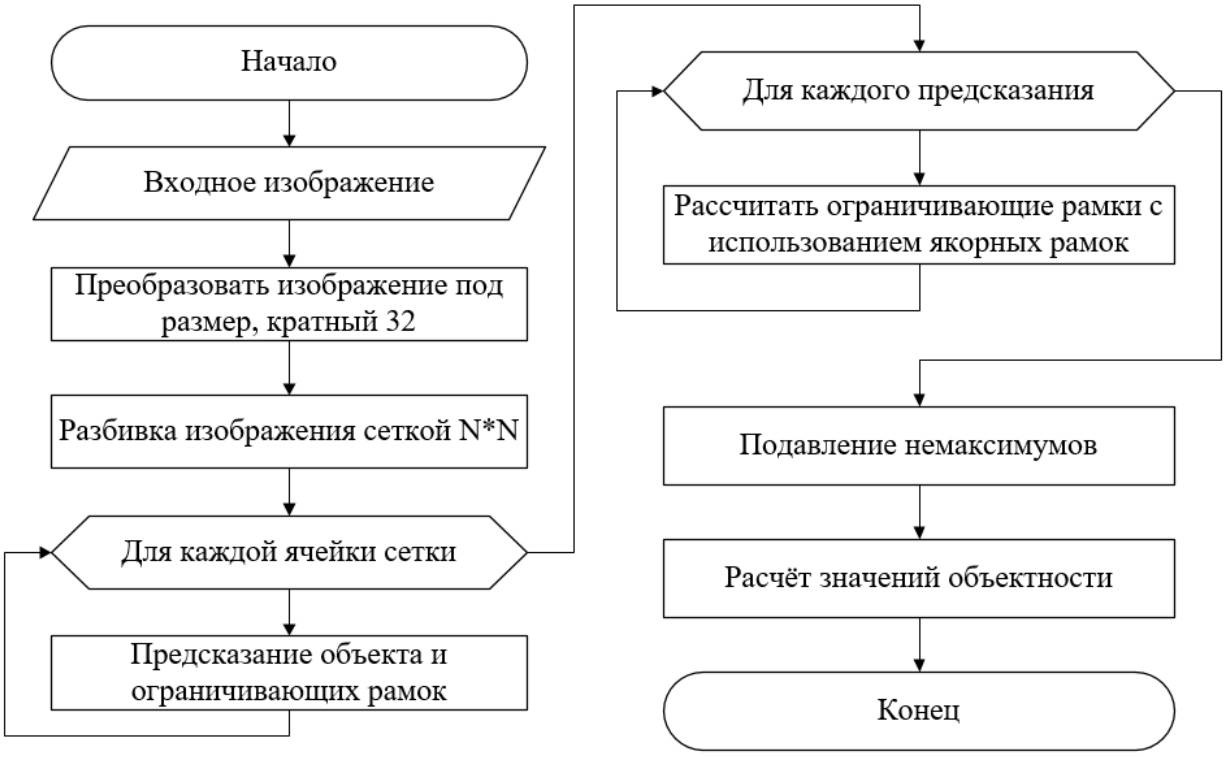

Рисунок 4 - Алгоритм идентификации YOLOv3

Структура нейронной сети, применяемой в YOLO, показана на рисунке 5.

	Тип слоя	Фильтры	Размер	Выход
	Свёрточный	32	3 × 3	256 × 256
	Свёрточный	64	3 × 3 / 2	128 × 128
1×	Свёрточный	32	1 × 1	
	Свёрточный	64	3 × 3	
	Остаточный			128 × 128
	Свёрточный	128	3 × 3 / 2	64 × 64
2×	Свёрточный	64	1 × 1	
	Свёрточный	128	3 × 3	
	Остаточный			64 × 64
	Свёрточный	256	3 × 3 / 2	32 × 32
8×	Свёрточный	128	1 × 1	
	Свёрточный	256	3 × 3	
	Остаточный			32 × 32
	Свёрточный	512	3 × 3 / 2	16 × 16
8×	Свёрточный	256	1 × 1	
	Свёрточный	512	3 × 3	
	Остаточный			16 × 16
	Свёрточный	1024	3 × 3 / 2	8 × 8
4×	Свёрточный	512	1 × 1	
	Свёрточный	1024	3 × 3	
	Остаточный			8 × 8
	Среднее объединение		Общий	
	Связанный		1000	
	Обобщение логистической функции			

Рисунок 5 - Слои нейронной сети

YOLO для идентификации принимает изображения определённого размера. Исходное изображение может отличаться по размерам. Подходящий размер, к которому надо привести изображение, рассчитывается по формуле (2).

$$Size = width * height, \quad (2)$$

где

- Size – размер изменённого изображения;
- width – ширина, кратная 32;
- height – высота, кратная 32.

Преобразование размеров изображения с помощью билинейной интерполяции производится по формулам (3), (4), (5) [12].

$$F(x,y_1) = \frac{x_2 - x}{x_2 - x_1} * F(x_1, y_1) + \frac{x - x_1}{x_2 - x_1} * F(x_2, y_1), \quad (3)$$

где

- F(x,y) – значение функции;
- x_2 – новое значение;
- x_1 – исходное значение;
- y_1 – исходное значение;

$$F(x,y_2) = \frac{x_2-x}{x_2-x_1} * F(x_1, y_2) + \frac{x-x_1}{x_2-x_1} * F(x_2, y_2), \quad (4)$$

где

- F(x,y) – значение функции;
- x_2 – новое значение;
- x_1 – исходное значение;
- y_2 – новое значение;

$$F(x,y) = \frac{y_2-y}{y_2-y_1} * F(x, y_2) + \frac{y-y_1}{y_2-y_1} * F(x, y_2), \quad (5)$$

где

- F(x,y) – значение функции;
- y_1 – исходное значение;
- y_2 – новое значение;

Нейросеть, заложенной в основу модели YOLOv3, состоит из 106 слоёв. Предсказание проводится на слоях 82, 94, 106. Предсказание объекта и центра его ограничивающей рамки происходит для трёх масштабов сетки на изображении:

- 32 – размер меньше масштабированной изображения в 32 раза;
- 16 – размер меньше масштабированного изображения в 16 раз;
- 8 – размер меньше масштабированного изображения в 8 раз.

Масштаб 32 нужен для определения объектов малых размеров, 16 – для средних размеров, 8 – для больших соответственно. Ширина и длина для масштабов сетки рассчитываются по формуле (6).

$$Width_{resized} * height_{resized} = Width/resize * Height/resize, \quad (6)$$

где

- $Width_{resized}$ – масштабированная ширина сетки;
- $height_{resized}$ – масштабированная высота сетки;
- Width – ширина изображения из пакета;
- Height - высота изображения из пакета;
- resize – изменение размера, принимает значения 8, 16, 32.

Для каждой из этих масштабированных сеток имеется ядро обнаружения 1x1. Форма каждого ядра имеет глубину, которая рассчитывается по формуле (7).

$$depth = b*(5+c), \quad (7)$$

где

- depth – глубина ядра;
- b – число ограничивающих рамок;
- c –число классов для идентификации.

Форма карты признаков представляет собой произведение масштабированного размера сетки и глубины ядра обнаружения. Карта особенностей рассчитывается по формуле (8).

$$Feature\ map = Width_{resized}*height_{resized}*\ depth, \quad 8)$$

где

- Feature map – карта особенностей;
- $Width_{resized}$ – масштабированная ширина сетки;
- $height_{resized}$ – масштабированная высота сетки;
- depth – глубина ядра.

Карта признаков от слоя к слою дополняется повышающей дискретизацией. Представление карты признаков показано на рисунке 6.

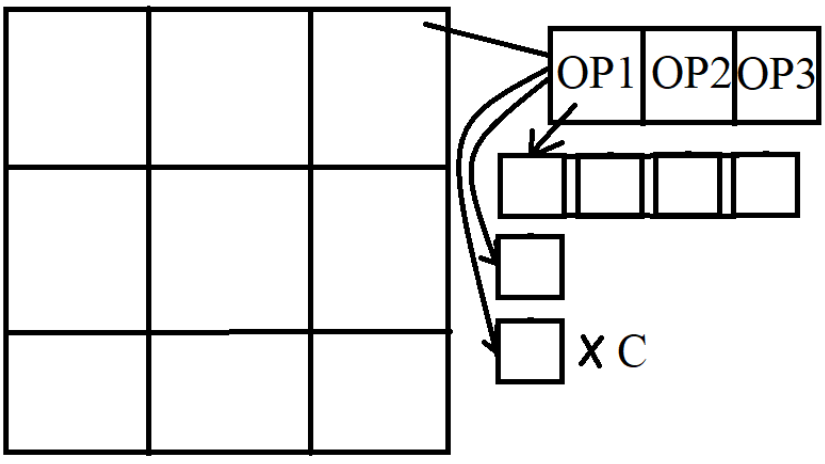

Рисунок 6 - Представление карты признаков

Так как при обучении модели предсказывается несколько ограничивающих рамок для объекта, то необходимо рассчитать функцию потерь, чтобы ориентироваться в том, насколько предсказание истинно. Функция потерь состоит из следующих ошибок:

- ошибка классификации;
- ошибка локализации;
- ошибка уверенности при обнаружении объекта;
- ошибка уверенности в случае, когда объект не удалось обнаружить.

Ошибка классификации, применяемая для обучения модели и проверки её обучения, рассчитывается по формуле (9).

$$\sum_{i=0}^{S^2} 1_i^{obj} * \sum_{c \,\epsilon\, classes} \left(p_i(c) - \tilde{p_i}(c)\right)^2, \tag{9}$$

где

- i – номер ячейки;
- S^2 – число ячеек в сетке;
- 1_i^{obj} – принимает значение 0 или 1 в зависимости от нахождения объекта в ячейке;
- c – класс объекта;
- $\tilde{p_i}(c)$ – вероятность того, что в ячейке находится объект класса.

Ошибка локализации ограничивающей рамки, отвечающей за объект, вычисляется по формуле (10).

$$\lambda_{coord} * \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} * [(x_i - \tilde{x_i})^2 + (y_i - \tilde{y_i})^2] + \lambda_{coord} * \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} * \left[\left(\sqrt{w_i} - \sqrt{\tilde{w_i}}\right)^2 + \left(\sqrt{h_i} - \sqrt{\tilde{h_i}}\right)^2\right], \tag{10}$$

где

- λ_{coord} – коэффициент увеличения веса ошибки, по умолчанию принимает значение 5;
- S^2 – число ячеек в сетке;

- B – число ограничивающих рамок;
- x,y – координаты центра рамки;
- w – ширина ограничивающей рамки;
- h – высота ограничивающей рамки;
- 1_{ij}^{obj} – принимает значение 0 или 1 в зависимости от нахождения объекта в ячейке;

Ошибка уверенности при обнаружении объекта рассчитывается по формуле (11).

$$\sum_{i=0}^{S^2}\sum_{j=0}^{B} 1_{ij}^{obj} * (C_i - C_i^{\sim})^2, \quad (11)$$

где

- i – номер ячейки;
- j – номер ограничивающей рамки;
- S^2 – число ячеек в сетке;
- B – число ограничивающих рамок;
- 1_{ij}^{obj} – принимает значение 0 или 1 в зависимости от нахождения объекта в ячейке;
- C_i – значение уверенности в ячейке.

Для случая, когда объект не найден, ошибка уверенности рассчитывается по формуле (12).

$$\lambda_{noobj}\sum_{i=0}^{S^2}\sum_{j=0}^{B} 1_{ij}^{noobj} * (C_i - C_i^{\sim})^2, \quad (12)$$

где

- i – номер ячейки;
- j – номер ограничивающей рамки;
- S^2 – число ячеек в сетке;
- λ_{noobj} – коэффициент снижения веса ошибки, по умолчанию принимает значение 0,5;

- 1_{ij}^{noobj} – дополнение к величине 1_{ij}^{obj};
- C_i – значение уверенности в ячейке.

Функция потерь представляет собой сумму из ошибок, рассчитанных по формулам (9), (10), (11), (12).

Для расчёта ограничивающих рамок используются предопределённые якорные рамки [8]. Принцип перехода от якорной рамки к ограничивающей показан на рисунке 7.

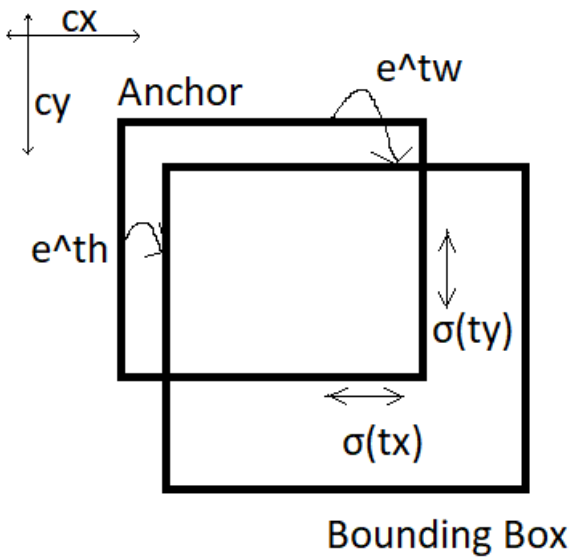

Рисунок 7 - Переход от якорной рамки к ограничивающей

Координаты центра ограничивающей рамки получены по формулам (13), (14).

$$b_x = \sigma(t_x) + c_x, \tag{13}$$

где

- $\sigma(t_x)$ – сигмоида абсциссы центра предсказанной рамки;
- c_x – смещение от левого верхнего угла изображения по оси OX.

$$b_y = \sigma(t_y) + c_y, \tag{14}$$

где

- $\sigma(t_y)$ – сигмоида ординаты центра предсказанной рамки;
- c_y – смещение от левого верхнего угла изображения по оси OY.

Ширина ограничивающей рамки рассчитывается по формуле (15).

$$b_w = p_w * e^{t_w}, \quad (15)$$

где

- p_w – ширина якорной рамки;
- t_w – ширина предсказанной рамки.

Длина ограничивающей рамки рассчитывается по формуле (16).

$$b_h = p_h * e^{t_h}, \quad (16)$$

где

- p_h – длина якорной рамки;
- t_h – длина предсказанной рамки.

Чтобы найти размеры ограничивающих рамок для исходного изображения, рассчитанные значения необходимо умножить на размеры исходного изображения. Для выявления наиболее значимых ограничивающих рамок используется величина IoU – пересечение над объединением. Пересечение и объединение показано на рисунке 8.

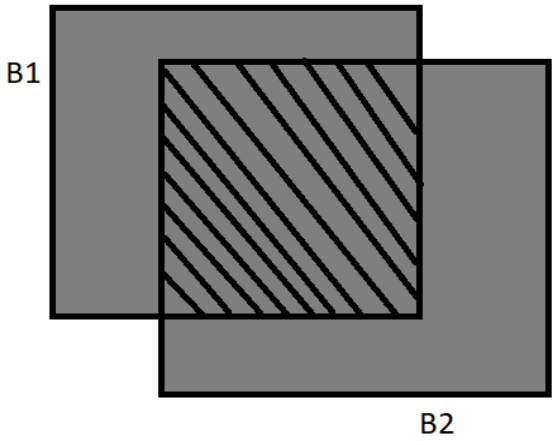

Рисунок 8 - Пересечение и объединение ограничивающих рамок

На рисунке 8 пересечением является заштрихованная область, объединением – закрашенная. Пересечение над объединением рассчитывается по формуле (17).

$$IoU = \frac{B_1 \cap B_2}{B_1 \cup B_2}, \quad (17)$$

где

- B_1, B_2 – ограничивающие рамки.

Для каждой пары ограничивающих рамок производится подавление немаксимумов. Рассчитывается пересечение над объединением, после чего из полученных значений для пар выбирается пара рамок с максимальным значением IoU, а остальные подавляются. По итогу для объекта остаётся одно предсказание его ограничивающей рамки, принадлежности классу и его объектность, что и требуется для его идентификации.

Значение объектности для объекта возрастает при приближении к центру его ограничивающей рамки и уменьшается при приближении к углам ограничивающей рамки. Значение объектности отвечает за нахождение объекта в ограничивающей рамке, рассчитывается по формуле (18).

$$p_0 = IoU * p_{predicted}, \quad (18)$$

где

- IoU – пересечение над объединением;
- $p_{predicted}$ – предсказанное значение объектности.

Таким образом, описан метод идентификации объектов с использованием модели YOLOv3. На формулах (2) – (18) указана математическая модель способа YOLOv3. Заложены составляющие метода идентификации YOLOv3 для программной реализации.

2.4 Математическое и алгоритмическое описание алгоритма идентификации объектов в видеопотоке

Идентификация объектов на данных с системы видеонаблюдения с применением метода распознавания объектов хорошо справляется со своей задачей в рамках одного изображения. Для идентификации объектов в видеопотоке разработан алгоритм, представленный на рисунке 9.

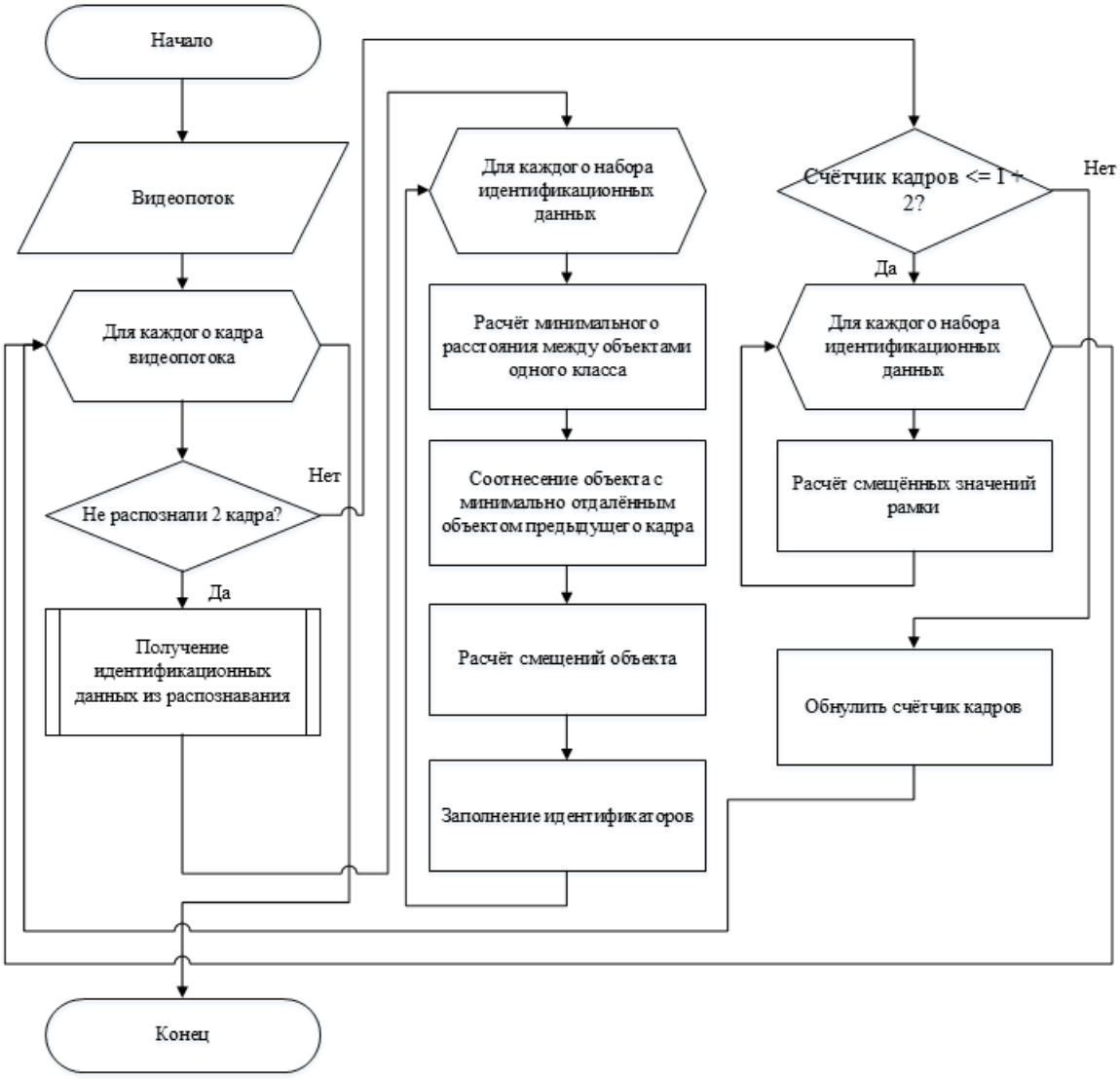

Рисунок 9 - Алгоритм идентификации объектов в видеопотоке

Данный алгоритм работает следующим образом:

На вход подаётся видеопоток. С помощью метода распознавания Yolov3 для двух кадров происходит формирование идентификационных данных для объектов. На основе полученных данных формируются идентификаторы для объектов. Чтобы определить, где объект с 1 кадра находится на 2 кадре применяется расчёт дистанции между центрами ограничивающих рамок объектов, принадлежащих классу текущего объекта. Координаты центра ограничивающей рамки рассчитывается по формулам (19), (20).

$$x = \frac{x_1 + x_2}{2}, \quad (19)$$

где

- x_1 – абсцисса левого верхнего угла;
- x_2 – абсцисса правого нижнего угла;

$$y = \frac{y_1 + y_2}{2}, \quad (20)$$

где

- y_1 – ордината левого верхнего угла;
- y_2 – ордината правого нижнего угла.

Расстояние между центрами ограничивающих рамок рассчитывается по формуле (21).

$$dist = \sqrt{(ax - bx)^2 + (ay - by)^2}, \quad (21)$$

где

- ax, ay – центр рамки объекта на первом кадре;
- bx, by – центр рамки объекта на втором кадре.

Для соотнесения объектов между кадрами необходимо выбрать минимальное расстояние и проверить на теоретическую пройденную дистанцию между кадрами. Теоретически пройденная между кадрами дистанция рассчитывается по формуле (22).

$$dist_{teory} = i * shift, \quad (22)$$

где

- i – число пропускаемых кадров между распознаванием;
- shift – смещение объекта.

Смещение объекта представляет собой разность между координатами ограничивающих рамок объектов на 1 и на 2 кадрах распознавания.

После соотнесения объектов между кадрами, происходит обновление идентификаторов. Для объектов, не нашедших пару в предыдущем кадре, идентификаторы используются без обновлений. Идентификаторы объектов проверяются на уникальность, после чего объектам с неуникальными

идентификаторами выдаются новые. Алгоритм присвоения идентификаторов представлен на рисунке 10.

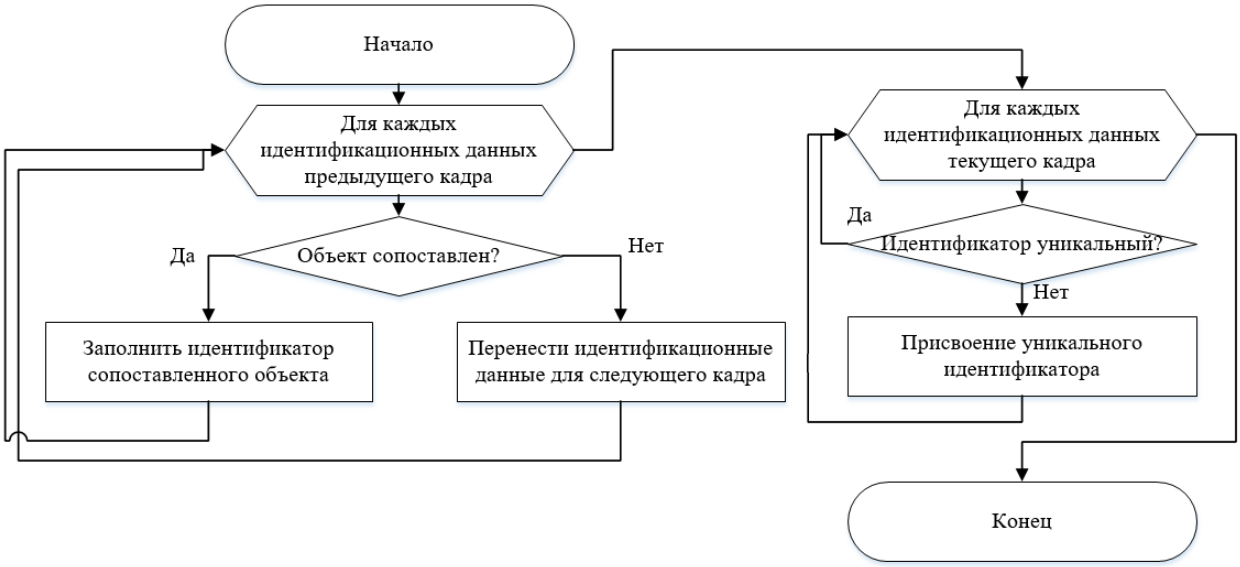

Рисунок 10 - Формирование идентификаторов

После распознавания объектов на двух кадрах и присвоения им идентификаторов кадры обрабатываются без этих шагов. Для следующих i кадров ограничивающие рамки рассчитываются на основе смещения. Координата рамки рассчитывается по формуле (23).

$$x_{new} = x + shift_x, \tag{22}$$

где

- x_{new}– новая координата рамки;
- x - старая координата рамки;
- $shift_x$ – смещение изменяемой координаты.

Когда кадр i + 2 обработан, процесс распознавания и идентификации продолжается в цикле.

Таким образом, для алгоритма идентификации объектов на данных видеопотока с камеры сформировано алгоритмическое описание. Математическое описание алгоритма представлено в формулах (19)-(22).

3 Программная реализация идентификации объектов на данных с системы видеонаблюдения

3.1 Описание реализованного метода для идентификации объектов на основе метода распознавания YOLOv3

Для реализации идентификации по методу YOLOv3 требуется применять нейронные сети. Для реализации программ с применением нейронных сетей наиболее подходящим языком программирования является Python. Выбранный язык программирования обладает следующими преимуществами:

- наличие библиотек и фреймворков для работы с нейросетями;
- высокая скорость разработки алгоритмов с использованием машинного обучения;
- гибкость с возможностью подобрать подходящую архитектуру программы во время разработки [19].
- Для работы программы требуется подключить следующие библиотеки:
- cv2 – работа с изображениями и видео [4];
- numpy – работа с математическими функциями;
- tensorflow – быстрые числовые вычисления [2];
- logging – ведение логов;
- repeat – создание итератора, который сам себя вызывает;
- sys – работа с параметрами командной строки;
- time – расчёт времени выполнения.

Программный код для подключения используемых библиотек показан на рисунке 11.

```
import sys
from keras import Model
from keras.layers import Conv2D, Input, LeakyReLU
from keras.layers import Add, Concatenate, Lambda
from keras.layers import UpSampling2D, ZeroPadding2D
from keras.regularizers import l2
import tensorflow as tf
import numpy as np
import cv2
from absl import logging
from itertools import repeat
import time
```

Рисунок 11 - Подключаемые библиотеки

Для работы с предварительно обученной моделью YOLOv3 необходимо указать следующие параметры:

- ограничения для немаксимального подавления;
- путь к файлу весов нейронной сети;
- путь к файлу чекпоинтов;
- размер, к которому приводятся входные изображения;
- список слоев в YOLOv3 FCN;

Заполнение этих параметров указано на рисунке 12.

```
yolo_intersection_over_union_limit   = 0.6 # порог пересечения относительно объединения (iou)
yolo_obj_score_limit = 0.6
yolov3_custom_weights = 'yolov3_custom.weights'# путь к файлу весов
input_image_size = 416            # приводим изображения к этому размеру
saved_checkpoints = 'checkpoints/yolov3.tf'# путь к файлу checkpoint'ов
number_of_classes = 10      # количество классов в модели
YOLOV3_Model_LAYERS = ['yolo_darknet', 'yolo_conv_0', 'yolo_output_0',
    'yolo_conv_1', 'yolo_output_1',  'yolo_conv_2',  'yolo_output_2',]
```

Рисунок 12 - Параметры для YOLOv3

Для инициализации модели YOLOv3 требуется использовать фреймворк darknet. Применение фреймворка отражается в следующих функциях:

- Darknet_Main_Structure (name=None);
- DarknetBlock(x, filters, blocks);
- Darknet_Residual_Layer (x, filters);
- Darknet_Convolutional_Layer (x, filters, size, strides=1, batch_norm=True).

Также для инициализации модели требуются функции, которые взаимодействуют с представленными функциями darknet. Имеются следующие функции реализации модели YOLOv3:

- yolov3_convolutional_layer (filters, name=None);
- yolov3_output (filters, anchors, classes, name=None);
- create_yolov3_boxes(prediction, anchors, classes);
- create_model_yolov3(size=None, channels=3, anchors=yolo_anchors, masks=yolo_anchor_masks, classes=10, training=False).

Основная функция, инициализирующая модель YOLOv3, представлена на рисунке 13.

```
def create_model_yolov3(size=None, channels=3, anchors=yolo_anchors,
          masks=yolo_anchor_masks, classes=10):
  x = inputs = Input([size, size, channels]) # формирование карты признаков
  x2, x3, x = Darknet_Main_Structure(name='yolo_darknet')(x) # заполнение карты признаков
  # заполнение данных на вывод для разных масштабов
  x = yolov3_convolutional_layer(512, name='yolo_conv_0')(x)
  output_0 = yolov3_output(512, len(masks[0]), classes, name='yolo_output_0')(x)
  x = yolov3_convolutional_layer(256, name='yolo_conv_1')((x, x3))
  output_1 = yolov3_output(256, len(masks[1]), classes, name='yolo_output_1')(x)
  x = yolov3_convolutional_layer(128, name='yolo_conv_2')((x, x2))
  output_2 = yolov3_output(128, len(masks[2]), classes, name='yolo_output_2')(x)
  # формирование ограничивающих рамок
  boxes_0 = Lambda(lambda x: create_yolov3_boxes(x, anchors[masks[0]], classes), name='yolo_boxes_0')(output_0)
  boxes_1 = Lambda(lambda x: create_yolov3_boxes(x, anchors[masks[1]], classes), name='yolo_boxes_1')(output_1)
  boxes_2 = Lambda(lambda x: create_yolov3_boxes(x, anchors[masks[2]], classes), name='yolo_boxes_2')(output_2)
  # подавление немаксимумов рамок
  results = Lambda(lambda x: nonMaximumSuppression(x),
                   name='nonMaximumSuppression')((boxes_0[:3], boxes_1[:3], boxes_2[:3]))
  return Model(inputs, results, name='yolov3')
```

Рисунок 13 - Функция инициализации модели YOLOv3

Данная функция заполняет карту признаков, получает контейнеры со значениями ограничивающих рамок и степеней уверенности нахождения объекта того или иного класса в рамке. После чего производится подавление не максимумов для отправки на вывод более точного предсказания. Функция, отвечающая за подавление немаксимумов, представлена на рисунке 14.

```
@tf.function
def nonMaximumSuppression(results):
  bounding_boxes, confidences, output_types = [], [], []
  for res in results:
    bounding_boxes.append(tf.reshape(res[0], (tf.shape(res[0])[0], -1, tf.shape(res[0])[-1])))
    confidences.append(tf.reshape(res[1], (tf.shape(res[1])[0], -1, tf.shape(res[1])[-1])))
    output_types.append(tf.reshape(res[2], (tf.shape(res[2])[0], -1, tf.shape(res[2])[-1])))
  bbox = tf.concat(bounding_boxes, axis=1) # ограничивающие рамки
  confidence = tf.concat(confidences, axis=1) # степени уверенности
  class_probs = tf.concat(output_types, axis=1) # значения классов
  scores = confidence * class_probs # расчёт объектности
  # применение подавления немаксимумов
  bounding_boxes, scores, classes, valid_detections = tf.image.combined_non_max_suppression(
    boxes=tf.reshape(bbox, (tf.shape(bbox)[0], -1, 1, 4)),
    scores=tf.reshape(scores, (tf.shape(scores)[0], -1, tf.shape(scores)[-1])),
    max_output_size_per_class=100,
    max_total_size=100,
    iou_threshold=yolo_intersection_over_union_limit,
    score_threshold=yolo_obj_score_limit)
  return bounding_boxes, scores, classes, valid_detections
```

Рисунок 14 - Функция подавления немаксимумов

Работа с параметрами командной строки при запуске программы, загрузка весов, инициализация модели YOLOv3, отправка изображений для идентификации происходят в главной функции программы main(). На рисунке 15 представлен фрагмент, отвечающий за работу с параметрами запуска.

```
def _main(args):
  if len(args) < 1 or len(args) > 3 :
    print("Ivalid arguments! Use -help as parameter.")
  else:
    source_key = ""
    can_run = True
    if len(args) == 1:
      source_key = '-i'
      source_value = "test.jpg"
    else:
      if len(args) == 2:
        if (args[1] == "-help"):
          print("The 1st parameter should be one of those parameters you want to identify:
          print("The 2nd parameter should be path to source you want to identify")
          print("For example add this when you launch the programme: -i test.jpg")
        else:
          if (args[1] == "-i"):
            source_key = "-i"
            source_value = "test.jpg"
          else:
            if (args[1] == "-v"):
              source_key = "-v"
              source_value = "v1.mp4"
            else:
              if (args[1] == "-c"):
                source_key = "-c"
                source_value = 0
```

Рисунок 15 - Фрагмент главной функции для обработки входных параметров при запуске программы

Фрагмент главной функции с реализацией идентификации на видео продемонстрирован на рисунке 16.

```
ret, frame = cap.read()
if ret:
  # Обработка изображения для идентификации
  img = tf.expand_dims(frame, 0)
  img = resize_image(img, input_image_size)
  if frames % 2 == 1:
    boxes, scores, classes, nums = yolo(img)
  img = draw_outputs(frame, (boxes, scores, classes, nums), class_names)
  # Выводим получившееся изображение на экран
  cv2.imshow('frame', img)
  if frames > 1:
    current_time = time.time() - start_time
    sum_time += current_time
    print("%s " % (current_time))
  frames += 1
```

Рисунок 16 - Фрагмент главной функции, в котором проводится идентификация на видео

По итогу описана программная реализация алгоритма YOLOv3. Изложены требующиеся для работы компоненты. Указаны функции, при помощи которых происходит инициализация модели, а также функции, которые обрабатывают предсказание обученной модели. Также рассмотрена передача на экран полученных по результатам идентификации данных. Основными библиотеками для реализации программы являются opencv 4.5.5, tensorflow-gpu 2.8, numpy 1.2.1.6. С полным текстом программы можно ознакомиться в приложении А.

3.2 Программная реализация алгоритма идентификации объектов

Для получения идентификационных данных объектов на кадре необходимо воспользоваться методом распознавания YOLOv3. Использование метода Yolo для получения уникальных значений ограничивающей рамки, принадлежности класса, значения объектности и уникального идентификатора в рамках кадра представлено на рисунке 17.

```
# Обработка изображения для идентификации
if j < 2:
  img = tf.expand_dims(frame, 0)
  img = resize_image(img, input_image_size)
  boxes, scores, classes, nums = yolo(img)
  j = j + 1
  img, object_info = draw_outputs(frame, (boxes, scores, classes, nums), class_names)
  # Выводим получившееся изображение на экран
  cv2.imshow('frame', img)
  if frames == 1:
    mxm = len(object_info)
```

Рисунок 17 - Получение идентификационных данных с помощью YOLOv3

Для поиска соответствующих объектов между распознанными кадрами применяется метод расчёта расстояния между их ограничивающими рамками. Расчёт расстояния представлен на рисунке 18.

```
def euclidean_dist(bb1, bb2):
  ax1, ay1, ax2, ay2 = bb1
  bx1, by1, bx2, by2 = bb2
  ax = (ax1 + ax2) / 2
  ay = (ay1 + ay2) / 2
  bx = (bx1 + bx2) / 2
  by = (by1 + by2) / 2
  return math.sqrt((ax - bx)**2 + (ay - by)**2)
```

Рисунок 18 - Расчёт расстояния между ограничивающими рамкам

Определение минимального расстояния между объектами класса показано на рисунке 19.

```
for k1 in object_info:
  mn = 100000
  mnid = 0
  for k2 in previous_object_info:
    if not k2[7] and k2[5] == k1[5]:
      val = euclidean_dist(k1[:4], k2[:4])
      if val < mn:
        mn = val
        mnid = k2[4]
```

Рисунок 19 - Определение минимального расстояния

После соотнесения объектов между кадрами выполняется расчёт смещения рамок объектов. Фрагмент кода, отвечающий за расчёт смещения, представлен на рисунке 20.

```
for k2 in previous_object_info:
  if k2[4] == mnid:
    k2[7] = True
    if j == 2:
      for z in range(0, 4):
        k1[z + 8] = k1[z] - k2[z]
    break
```

Рисунок 20 - Расчёт смещения ограничивающих рамок

После соотнесения объектов между кадрами и вычисления смещений рамок необходимо проверить уникальность идентификаторов. Идентификаторы текущего кадра дополняются идентификаторами предыдущего кадра. Если возникают неуникальные значения, то происходит замена на уникальные. Принцип работы замены идентификаторов представлен на рисунке 21.

```
if j == 2:
  for k2 in previous_object_info:
    if not k2[7]:
      object_info.append(k2)
for k1_ind in range(len(object_info) - 1):
  for k2_ind in range(k1_ind + 1, len(object_info)):
    if object_info[k1_ind][4] == object_info[k2_ind][4]:
      mxm = mxm + 1
      object_info[k1_ind][4] = mxm
```

Рисунок 21 - Замена неуникальных идентификаторов

Кадры, которые пропускаются при распознавании требуют идентификационные данные, сформированные на предыдущем шаге. Показанный на рисунке 22 фрагмент кода отвечает за перенос ограничивающей рамки объекта между такими кадрами.

```
if j > 2 and j <= i + 2:
  j = j + 1
  for k1 in object_info:
    for k2 in previous_object_info:
      if k1[4] == k2[4]:
        for z in range(0, 4):
          k1[z] = k1[z] + k2[z+8]
```

Рисунок 22 - Перенос ограничивающей рамки

В ходе работы алгоритм оперирует с идентификационными данными объекта, представленными в следующем виде:

- координаты ограничивающей рамки объекта на исходном кадре;
- уникальный идентификатор в виде номера;
- имя класса, к которому объект принадлежит;
- значение объектности;
- сопоставлен ли объект;
- координаты смещения ограничивающей рамки.

Таким образом, рассмотрена программная реализация идентификации объектов в видеопотоке. Для реализации использовалась библиотека math.

3.3 Тестирование и повышение скорости работы программной реализации алгоритма идентификации объектов на данных с системы видеонаблюдения

Для тестирования необходимо проверить работоспособность программы в процессе идентификации объектов. По поставленной задаче на реализацию, идентификация объектов может осуществляться на изображении, видео или видеопотоке с камеры. Для разграничения этих возможностей применяются параметры команды python при запуске программы. В общем виде параметры при запуске из терминала выглядят следующим образом:

- полный путь к основному файлу и его имя;
- ключ источника для идентификации – изображение, видео, видеопоток;
- полный путь и имя источника для идентификации.

Для идентификации объектов на изображении необходимо запустить программу из терминала с ключом источника –i и полным названием входного изображения. Пример запуска в терминале показан ниже:

python main.py -i test.jpg.

Входное изображение для тестирования представлено на рисунке 23.

Рисунок 23 - Входное изображения для тестирования

После прохождения процесса идентификации на исходное изображение с крытой автопарковки выносятся ограничивающие рамки, метка класса объекта с подклассом и значение уверенности объектности. Пример идентификации объектов на изображении представлен на рисунке 24.

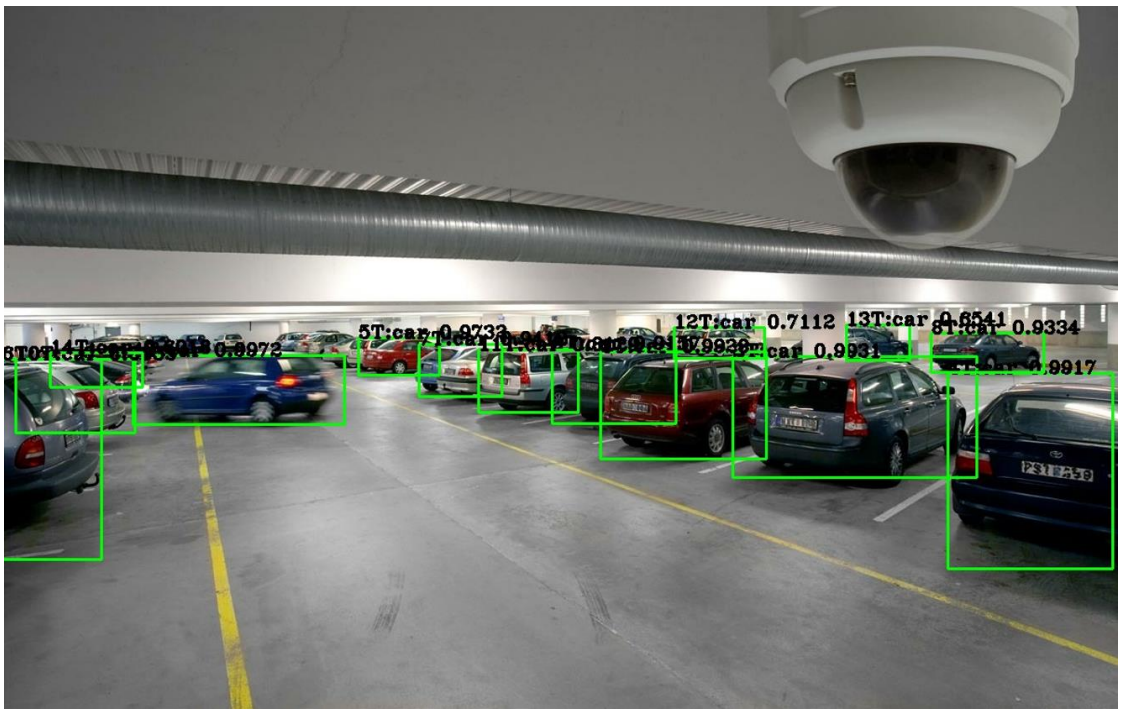

Рисунок 24 - Пример идентификации объектов на изображении

Зелёным цветом выделены ограничивающие рамки объекта, чёрным цветом указывается принадлежность идентифицированного образа в рамке к классу, также чёрным цветом указано значение объектности для каждого объекта, идентифицированного на изображении. Мелкие объекты вдали не были идентифицированы, т.к. были отброшены по ограничивающим значениям объектности. Выполнение работы по изображению произошло примерно за 0,08 секунд. Идентификация успешно произведена для всех распознанных объектов.

Для идентификации объектов на видео необходимо запустить программу из терминала с ключом источника –v и полным названием входного видео. Пример запуска в терминале с помощью следующей команды:

python main.py -v video.mp4.

Кадр из видео после выполнения идентификации представлен на рисунке 25.

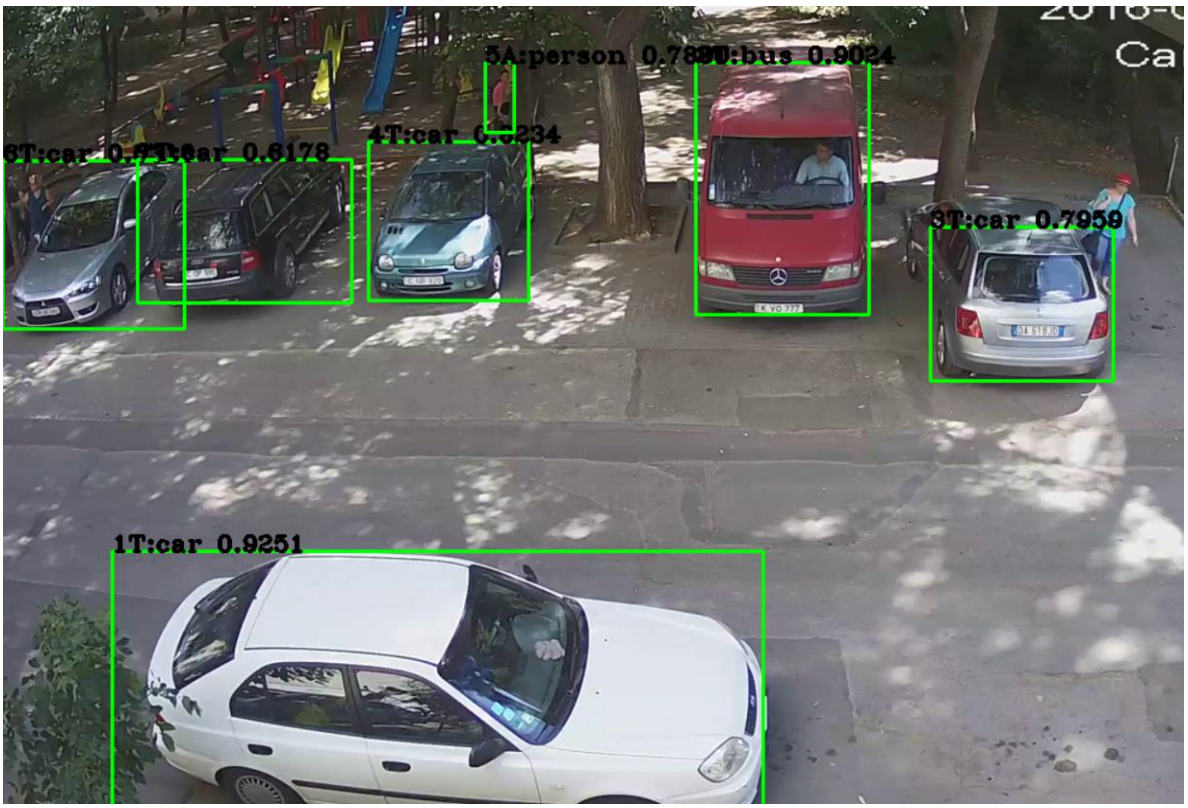

Рисунок 25 - Кадр из видео после выполнения идентификации

Выполнение идентификации объектов на каждом кадре заняло примерно 0,14 секунд. Число кадров в секунду на выходном видеопотоке с идентифицированными объектами составило примерно 8 кадров в секунду.

Для идентификации объектов на видеопотоке с камеры необходимо запустить программу из терминала с ключом источника –c и идентификатором камеры, с которой программа получает входной видеопоток. Пример запуска с помощью команды показан ниже:

python main.py -c 0.

Кадр с идентифицированными объектами из видеопотока с камеры показан на рисунке 26.

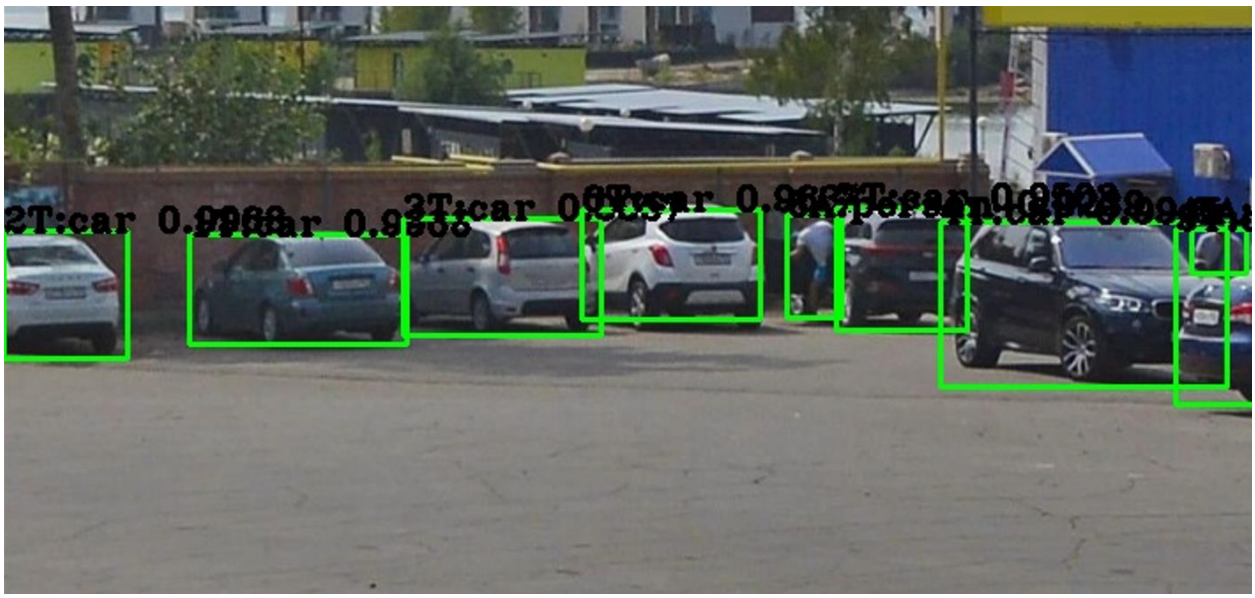

Рисунок 26 - Кадр с идентифицированными объектами из видеопотока с камеры

Идентификация объектов на видеопотоке с камеры заняла примерно 0,13 секунд. Число кадров в секунду примерно равно 8. От запуска к запуску среднее время выполнения и количество кадров в секунду могут варьироваться с отклонением 0,07 и 0,6 соответственно.

Для увеличения скорости работы программы следует применить два способа:

- ускорение при помощи специальных функций;
- ускорение при помощи воздействия на алгоритм вывода идентифицированных объектов.

Так как в реализации используется tensorflow с поддержкой вычислений на GPU, то для ускорения работы программы применяется встроенная функция-декорация @tf.function. Данная декорация применяется совместно с функцией, вычисления в которой необходимо ускорить. Декорация применима не для всех функций и имеет следующие ограничения [3]:

- рекомендуется, чтоб функция возвращала на выходе тензор из tensorflow;

- функция не может выполнить операцию print() несколько раз – будет выводиться только первый объект;
- использование декорации на разных типах возвращаемых значений может негативно повлиять на производительность;
- использование данной функции к функциям с малым числом вызовов, малым числом вычислений, малой сложностью вычислений может не дать положительного эффекта.

С применением функции @tf.function время обработки для одиночного изображения стало 0,055 секунд. Отсюда следует, что время выполнения программы для идентификации изображения уменьшилось на 0,025. Ускорение работы вышло в 1,46 раз.

Для идентификации объектов на видео с использованием функции @tf.function время выполнения стало примерно равно 0,11 секунд и 9,5 кадров в секунду соответственно. Разница между замерами до ускорения и после составила 0,03 для среднего времени и для 1,5 числа кадров в секунду. Ускорение работы программы вышло в 1,27 раз, число кадров в секунду увеличилось в 1,19 раз.

Для идентификации объектов на видеопотоке с камеры вместе с применением декорации @tf.function время выполнения стало примерно равно 0,11 секунды и 9,4 кадров в секунду соответственно. Разница между замерами до ускорения и после составила 0,02 для среднего времени и 1,4 для числа кадров в секунду. Ускорение работы программы вышло в 1,18 раз, число кадров в секунду увеличилось в 1,18 раз.

Данный способ ускорения программы улучшил временные показатели. Для дополнительного повышения скорости работы программы применяется алгоритмический метод ускорения. Он основан на пропуске каждого кадра, кроме i-го, видеопотока при идентификации объектов на видео с диска или с камеры. Так как конечный пользователь видит лишь видеопоток с идентифицированными объектами на нём, а человеческий глаз зачастую неспособен воспринимать разницу между быстро меняющимися кадрами, то

пропуск шага с идентификацией для изображений, кроме i-го в потоке позволяет приобрести выигрыш в скорости работы. Однако, если взять относительно большое значение i, то реализация потеряет в точности определения идентификационной ограничивающей рамки.

Данный метод не используется для ускорения программы в случае идентификации изображения, так как изображение одно, и его необходимо отправить на идентификацию.

Число кадров в секунду рассчитывается по формуле (23).

$$FPS = \frac{Time}{Frames}, \tag{23}$$

где

- Time – время работы программы;
- Frames – число кадров, которое программа обработала.

Для отображения результатов повышения скорости работы программной реализации замеры времени помещены в таблицу 7.

Таблица 7 - Замеры времени работы программы с методами ускорения

Способ\замер	Изображение, время (с)	Видео, время (с)	Видеопоток, время (с)	Видео, FPS (кадры / с)	Видеопоток, FPS (кадры / с)
Без ускорения	0,08	0,14	0,13	8	8
@tf.function	0,055	0,11	0,11	9,5	9,4
Пропуск кадров, i = 2	0,08	0,07	0,064	14,2	15
Пропуск кадров, i = 3	0,08	0,054	0,047	18,5	21,3
Пропуск кадров, i = 4	0,08	0,049	0,038	20,6	25,8

Из таблицы 7 уже видно, что метод с пропуском кадров работает эффективнее, т.к. понижает среднее время выполнения программы и повышает число кадров в секунду при идентификации видеопотока. Во время тестирования при использовании метода пропуска со значениями i, большими или равными трём, растёт ошибка локализации. Ошибка

локализации ограничивающей рамки видна невооружённым взглядом, так как рамка отстаёт от движущегося объекта.

Ускорение работы программы рассчитывается по формуле (24).

$$a = \frac{T_s}{T_f}, \tag{24}$$

где

- T_s – время работы программы без ускорений;
- T_f - время работы ускоренной программы.

По замерам времени с использованием методов ускорения рассчитаны значения ускорения. Рассчитанные значения ускорения приведены в таблице 8.

Таблица 8 - Рассчитанные значения ускорения

Способ\ускорение	Изображение	Видео	Видеопоток
@tf.function	1,46	1,27	1,18
Пропуск кадров, i = 2	1	2	2,03
Пропуск кадров, i = 3	1	2,59	2,76
Пропуск кадров, i = 4	1	2,86	3,42

Из таблицы 8 видно, что функция-декорация слабо ускоряет работу программы. Её полезное применение ярче заметно при использовании для идентификации изображений. Эффективность метода пропуска кадров доказана при помощи замеров времени в ходе тестирования. Данный метод ускоряет работу программы, повышает стабильность предсказания, но при значениях параметра i больше двух использует предсказания с возрастающей ошибкой локализации.

Для сравнения с разработанным алгоритмом идентификации объектов в видеопотоке использованы методы слежения за объектом. Проведены замеры времени при пропуске 10 кадров для распознавания. Данные со средним значением кадров в секунду и размерами входного видеопотока на примере видео представлены в таблице 9.

Таблица 9 - Сравнение с методами слежения за объектом

Методы	440x360	450x360	1280x720	1280x720	960x720	Среднее
MOSSE	48,3	51,1	6,27	27,6	29,87	32,63
MIL	14,3	11,04	2,08	10,47	5,46	8,67
KCF	21	40,08	1,29	6,89	7,64	15,38
TLD	7,1	0	0	0	9,74	3,37
MedianFlow	50,04	47,4	22,14	38,97	34,28	38,57
Boosting	6,2	0	0	0	6,5	2,54
CSRT	12,03	22,67	0	0	0	6,94
MY	66,81	61,79	34,95	54,5	51,1	53,83
MY_ID	56,51	56,18	47,14	48,2	44,56	50,52

Сравнительный анализ показал, что методы слежения за объектом работают достаточно медленно и являются менее предпочтительными для применения, чем разработанный алгоритм. Самый быстрым методом является идентификация на основе распознавания, однако имеет значительные ошибки локализации при пропуске кадров боле 3, поэтому наиболее оптимальным методом является метод идентификации с расчётами смещения рамки и заполнения идентификаторов. Методы слежения за объектом имеют низкие значения кадров в секунду, ошибки локализации при слежении за множеством объектов. Следовательно, их применение нецелесообразно для идентификации объектов на видеопотоке в реальном времени.

Для выбора оптимального числа пропуска кадров распознавания i необходимо проанализировать работу программы при различных значениях числа пропуска. Для тестирования выбраны значения пропуска кадров от 1 до 40. Так как изменения числа кадров в секунду при изменении числа пропуска кадров меняются незначительно, то принято решение увеличивать число пропускаемых кадров на 5. Замеры кадров в секунду в зависимости от увеличения пропуска кадров представлены в таблице 10.

Таблица 10 - Замеры скорости в зависимости от изменения пропусков i

Число пропускаемых кадров	Среднее число кадров в секунду
1	15,85
5	31,39
10	50,52
15	65,72
20	82,46
25	97,38
30	109,4
35	113,4
40	134,16

По таблице 10 построен график роста скорости в зависимости от увеличения пропущенных при распознавании кадров. График роста скорости показан на рисунке 27.

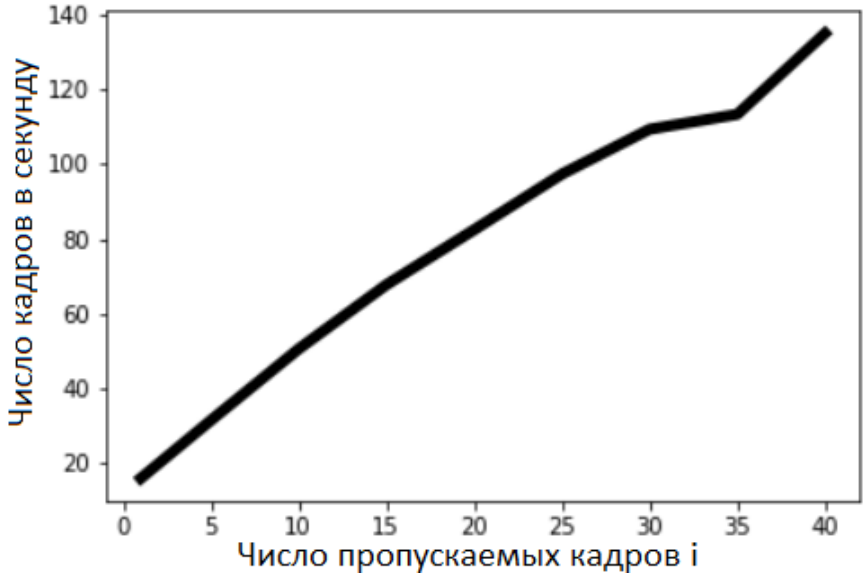

Рисунок 27 - График роста числа кадров в секунду в зависимости от числа пропускаемых кадров

Стоит учесть, что с увеличением числа пропускаемых кадров растёт ошибка локализации, следовательно, нужно остановиться на более оптимальном значении пропуска кадров 20 или 25.

Таким образом, была протестирована работоспособность реализованного метода идентификации объектов на данных, получаемых с системы видеонаблюдения. Работоспособность программы удалось ускорить с 8 кадров в секунду до 97 кадров в секунду. Оптимальным циклом работы программы при идентификации в видеопотоке является 2 кадра с использованием метода распознавания YOLOv3 и 25 кадров с переносом рассчитанных смещений. Идентифицированные объекты сопровождаются ограничивающей рамкой, уникальным идентификатором, значением объектности, принадлежностью к классу. Следовательно, поставленные на реализацию задачи выполнены.

Заключение

В ходе выполнения выпускной квалификационной работы на тему «Идентификация объектов на основе данных, получаемых с системы видеонаблюдения» сформированы выводы и подведены следующие итоги:

- Произведено исследование по актуальной теме идентификации объектов в видеопотоке с применением актуального метода распознавания YOLOv3;
- для объекта выделено 4 вида данных, идентифицирующих его;
- для идентификации выбрано 10 объектов, относящиеся к парковке;
- для видеонаблюдения лучше подходят IP камеры с углом наклона от 30 до 40 градусов и фокусным расстоянием 6,9 мм;
- для идентификации объектов в видеопотоке с системы видеонаблюдения за основу выбран один из восьми методов распознавания;
- оптимальным методом является YOLOv3, получивший среднюю оценку 9,2;
- построена математическая модель YOLOv3 с на основе 17 формул;
- построена математическая модель алгоритма идентификации на основе 4 формул;
- программная реализация выполнена с использованием являются opencv 4.5.5, tensorflow-gpu 2.8, numpy 1.2.1.6.
- тестирование реализованного алгоритма показало ускорение с 8 до 98 кадров в секунду.

По итогу для достижения цели - разработать алгоритм идентификации объектов на данных с системы видеонаблюдения - все поставленные задачи выполнены. Реализация программы приобрела усовершенствование в скорости работы. Следовательно, цель выпускной квалификационной работы полностью достигнута.

Список используемой литературы

1. Breaking Down Facial Recognition: The Viola-Jones Algorithm [Электронный ресурс] – Режим доступа: https://towardsdatascience.com/the-intuition-behind-facial-detection-the-viola-jones-algorithm-29d9106b6999 (Дата обращения: 17.03.2022).
2. Deep learning with Applications Using Python. N.K. Manaswi– 2018, - 227 с.
3. How to use tf.function to speed up Python code in Tensorflow [Электронный ресурс] – Режим доступа: https://www.machinelearningplus.com/deep-learning/how-use-tf-function-to-speed-up-python-code-tensorflow/ (Дата обращения: 10.05.2022).
4. Learning OpenCV 3 Computer Vision with Python Second Edition. J. Minichino, J. Howse – 2015, - 356 с.
5. Object recognition. Распознавай и властвуй [Электронный ресурс] – Режим доступа: https://habr.com/ru/company/jetinfosystems/blog/498294/ (Дата обращения: 14.03.2022).
6. Understanding Region-based Fully Convolutional Networks (R-FCN) for object detection [Электронный ресурс] – Режим доступа: https://jonathan-hui.medium.com/understanding-region-based-fully-convolutional-networks-r-fcn-for-object-detection-828316f07c99 (Дата обращения: 15.03.2022).
7. YOLOv3 Architecture: Best Model in Object Detection [Электронный ресурс] – Режим доступа: https://bestinau.com.au/yolov3-architecture-best-model-in-object-detection/ (Дата обращения: 03.04.2022).
8. YOLOv3: An Incremental Improvement. J. Redmon, A. Farhadi – 2018, - 6 с.

9. You Only Look Once: Unified, Real-Time Object Detection. J. Redmon, S. Divvala, R. Girshick, A. Farhadi – 2015, - 10 с.

10. Анализ существующих подходов к распознаванию лиц [Электронный ресурс] – Режим доступа: https://habr.com/ru/company/synesis/blog/238129/ (Дата обращения: 11.03.2022).

11. Аналоговые камеры наблюдения [Электронный ресурс] – Режим доступа: https://habr.com/ru/post/573312/ (Дата обращения: 08.03.2022).

12. Билинейная интерполяция [Электронный ресурс] – Режим доступа: https://www.hmong.press/wiki/Bilinear_interpolation (Дата обращения: 27.03.2022).

13. Видеоаналитика: задачи и решение [Электронный ресурс] – Режим доступа: https://www.osp.ru/lan/2014/06/13041879 (Дата обращения: 08.03.2022).

14. Видеоналитика, системы и их сравнение. К. А. Рылов - Томск, 5 с.

15. Задача нахождения объектов на изображении [Электронный ресурс] – Режим доступа: https://neerc.ifmo.ru/wiki/index.php?title=Задача_нахождения_объектов_на_изображении (Дата обращения: 20.02.2022).

16. Идентификационные признаки и их классификация [Электронный ресурс] – Режим доступа: https://studme.org/90711/pravo/identifikatsionnye_priznaki_klassifikatsiya (Дата обращения: 09.03.2022).

17. Интеллектуальные системы видеонаблюдения [Электронный ресурс] – Режим доступа: https://www.mascom-vostok.ru/service/intellektualnye-sistemy-videonablyudeniya/ (Дата обращения: 08.03.2022).

18. Модель, метод и комплекс программ выделения контуров на

изображениях с использованием энергетических признаков. Г. В. Костюхина - Казань, 238 с.

19. Нейронные сети на Python: как всё устроено [Электронный ресурс] – Режим доступа: https://gb.ru/blog/nejronnye-seti-python/ (Дата обращения: 05.05.2022).

20. Основные методы идентификации объектов [Электронный ресурс] – Режим доступа: https://ria-stk.ru/stq/adetail.php?ID=5817 (Дата обращения: 11.03.2022).

21. Распознавание лиц на основе применения метода Виолы-Джонса, вейвлет преобразования и метода главных компонент. Б.Т.Т. Чанг, Ф.Н. Хоанг, Спицын В.Г. – Томск, - 2012, - 5 с.

22. Способ идентификации объектов на цифровых изображениях подстилающей поверхности методом нечеткой триангуляции Делоне [Электронный ресурс] – Режим доступа: https://patenton.ru/patent/RU2729557C2 (Дата обращения: 13.03.2022).

23. Стандарты идентификации, распознавании и детектировании людей [Электронный ресурс] – Режим доступа: https://infotech.com.ua/article/standarty-identifikacii-raspoznavanii-i-detektirovanii-ludei (Дата обращения: 09.03.2022).

24. Установка системы видеонаблюдения [Электронный ресурс] – Режим доступа: https://bezopasnik-info.turbopages.org/bezopasnik.info/s/установка-системы-видеонаблюдения (Дата обращения: 08.03.2022).

25. Формирование штрих-кода по изображениям лиц на основе градиентов яркости. Кухарев Г.А., Матвеев Ю.Н., Щеголева Н.Л. – Научно-технический вестник информационных технологий, механики и оптики, - 2014, - 91 с.

Приложение А

Файловые ресурсы реализованной программы для идентификации объектов на основе данных, получаемых с системы видеонаблюдения

Исходный код программной реализации алгоритма идентификации объектов на данных, получаемых с системы видеонаблюдения, находится на приложенном компакт-диске. Все файловые ресурсы находятся в папке под названием «Приложение А».

Таблица А.1 - Методы идентификации объектов

Номер способа	Название	Описание	Преимущества	Недостатки
1	Метод формирования штрих-кодов [25]	Формирует штрих-код EAN-8 на идентифицируемом лице.	Устойчивость к вращению изображения	Требуется дополнительно преобразовать изображение в чёрно-белое. Преимущественно применяется для распознавания лиц.
2	Метод гибкого сравнения на графах [10]	Сопоставление графов, которые отражают образ на изображении в виде взвешенных рёбер и вершин. Сравнение графов происходит с данными из базы.	Даёт возможность представления объекта в трёхмерном пространстве. Точность идентификации составляет 95%.	Подходит преимущественно для распознавания лиц. Длительное вычисление функции деформации. Ошибки при неверном построении графа из-за низкого качества изображения.

Продолжение таблицы А.1.

3	Метод главных компонент [21]	Линейное ортогональное преобразование вектора одной размерности в вектор иной размерности. Для вычисления главных компонент используется вычисление собственных векторов. Процесс идентификации заключается в сравнении главных компонент известных объектов с компонентами нового.	Относительно небольшой размер выборки для обучения.	Подходит преимущественно для распознавания лиц. Увеличение вычислительной нагрузки при более высокой требуемой точности идентификации.
4	Метод Виолы-Джонса [1]	Использует окно поиска для нахождения образов на изображении, представленном в интегральном виде. Построение функции классификации при помощи алгоритма бустинга. Итеративное построение классификатора.	Обнаружение сразу нескольких объектов на изображении. Возможность дополнительно обучить классификатор на распознавание новых объектов. Малое число ложных срабатываний.	Долгое обучение классификатора. Для обучения требуется большой набор данных. Подходит преимущественно для распознавания лиц.
5	Метод нечёткой триангуляции Делоне [22]	Матрица изображения с высоким разрешением нормируется. Затем разбивается на треугольники. После чего происходит сравнение площадей	Идентификация объектов на нечётких изображениях.	Неоднозначность разбиения изображения на мозаику участков, что приводит к погрешности вычислений. Понижение качества

		треугольников с площадями эталона.		идентификации.

Продолжение Приложения А

Продолжение таблицы А.1.

6	R-CNN [5]	Определяется набор гипотез. Для каждой гипотезы из регионов извлекаются признаки и обрабатываются свёрточной нейронной сетью. Далее признаки из региона передаются классификатору, который идентифицирует объект. Происходит корректировка гипотезы.	Проведение обучения модели независимо от самой идентификации объектов. Распознавание нескольких объектов на изображении.	Дублирующиеся гипотезы. Долгое время работы, что не даёт работать в режиме реального времени. Нет обучения для поиска гипотез.
7	R-FCN [6]	Создаётся 9 карт признаков. Проверяется, насколько вероятно область содержит часть объекта. Рассчитывается средняя вероятность.	Извлечение признаков при взгляде на изображение 1 раз. Высокая скорость идентификации.	Чувствительность к местоположению объекта.
8	YOLOv3 [9]	Разбивает изображение на сетки. Применение якорных рамок. Подавление не максимумов для уменьшения числа рамок для объекта.	Изображение рассматривается один раз. Высокая скорость работы. Идентификация сразу нескольких объектов на изображении.	Длительное предварительное обучение. Требуется производительная аппаратура.

Описание файловых ресурсов представлено в таблице А.2.

Таблица А.2 - Описание файловых ресурсов

Наименование файла	Описание файла

Продолжение таблицы А.2.

main.py	Программный код на языке Python, который отвечает за реализацию алгоритма идентификации объектов на данных, получаемых с системы видеонаблюдения
yolov3_custom.weights	Файл содержит веса модели YOLOv3, обученной на датасете объектов, полученных с системы видеонаблюдения на парковке