Казахский национальный университет им. аль-Фараби

УДК 004.021:004.421 На правах рукописи

БУРИБАЕВ ЖОЛДАС АЛЛАДИНОВИЧ

Разработка эффективных параллельных алгоритмов машинного обучения для системы ориентации робота в пространстве

6D075100 - «Информатика, вычислительная техника и управление»

Диссертация на соискание степени
доктора философии (PhD)

Научные консультанты:
д.т.н., профессор,
член-корр. НАН РК Е.Н. Амиргалиев
PhD, профессор Миачи Тайзо
(Токайский университет , Япония)

Республика Казахстан
Алматы, 2022

СОДЕРЖАНИЕ

Нормативные ссылки

В данной диссертации использовались ссылки согласно следующим стандартам:

ГОСО РК 5.04.034-2011 «Государственный общеобязательный стандарт образования Республики Казахстан. Послевузовское образование. Докторантура». Основные правила заверены Министерством Образования и Науки РК. «17» июнь 2011 ж. №261. Астана 2011.

«Инструкция по оформлению диссертационных работ и авторефератов, МОН РК, Комитет высшей аттестации, Алматы, 2004, МЕСТ 7.1-2003. Библиографическая опись.

ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

ИИ – искусственный интеллект
ML (МЛ) – Machine Learning (Машинное обучение)
АПК – Агропромышленный комплекс
API – Application Programming Interface
BPNN – Backpropagation Neural Network
CLAHE -
CMYK – Cyan Magenta Yellow Black
CNN – Convolutional Neural Network
DT – Decision Tree
FN – False Negative
FP – False Positive
GNB – Gaussian Naïve Bayes
GPS – Global Positioning System
HSI – Hue Saturation Intensity
HSV – Hue Saturation Value
KNN – K-Nearest Neighbors
LDA – Linear Discriminant Analysis
LR – Linear Regression
Mask R-CNN – Mask Fully Convolutional Neural Network
ML – Machine Learning
NIR – Near Infrared
PLSDA – Partial Least Squares Discriminant Analysis
PR – Precision-Recall
PRF – Parallel Random Forest
RGB – Red Green Blue
RMSE – Root Mean Square Error
SDF – Stochastic Decision Field
SLAM – Simultaneous Localization and Mapping
SQL – Structured Query Language
SR – Spatial Resolution
SSC – Soluble Solids Content
SVM – Support Vector Machine
SVMDA – Support Vector Machine Discriminant Analysis
TN – True Negative
TP – True Positive
XGBoost – Extreme Gradient Boosting
YIQ – Y Inphase Quadrature
YOLO – You Only Look Once

ВВЕДЕНИЕ

Актуальность работы. Современное развитие робототехники тесно связано с искуственным интеллектом (ИИ), и многие перспективы кроются во внедрении в автоматизацию элементов ИИ. Также, можно смело утверждать, что понятие ИИ неразрывно связано с робототехникой [1, 2]. Робот – это машина или програмно-аппаратное оборудование способные воспринимать окружающую среду, включая объекты и взаимосвязи между объектами и различные процессы, интерпретировать их и самостоятельно выполнять различные ответные действия, влияющие на данную среду. Если роботы ранее использовались в основном с целью замены человеческого труда в трудных участках или по производительными исполнителями по скорости на заводах, то с началом новой эры роботов, взгляды на разработки сменились развитием и дополнением роботов к человеческому труду и творческому подходу [3]. Новая эра робототехники на прямую саязана внедрением искусственного интеллекта [4] в инструменты изучения, анализа и обоснованого принятия решения робота, в зависимости окружающей среды и их влияния на него, так как на протяжении долгого времени робототехника развивалась без неё. Основными проблемами по внедрению ИИ являлись ограниченность вычислительных ресурсов, а также малоизученность подходов, решений, методов и алгоритмов искусственого интеллекта, позволяющие повысить функциональную интеллектуальность робота по принятию решению. Современные технологии при своих сегоднящних возможностях уже позволяют проводить вычисления подобного рода, и теперь задача состоит в решении второй проблемы – внедрения.

Функциональное определение для робота можно привести как определение STA – SENSE/THINK/ACT [5], которая изображена на рисунке 1.

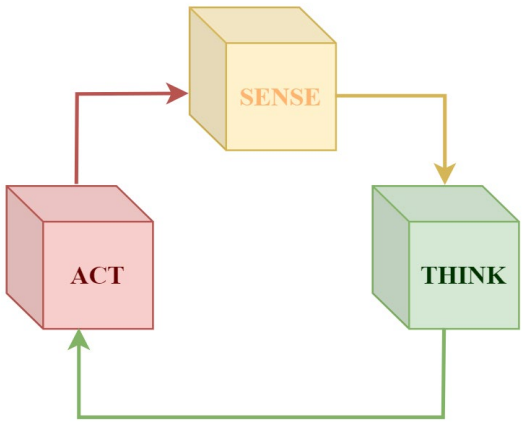


Рисунок 1 — Функционал робота по определению STA

SENSE – восприятие окружающего мира;

THINK – понимание окружающего физического мира и способность строить модели поведения, для выполнения предназначенных действий;

ACT – воздействие на физический мир действиями;

и отсутствие как минимум одного из данных функционалов, списывает устройство со списков робота. В свою очередь каждое из функционалов имеет сложную систему реализации.

Одной из актуальных задач в робототехнике на сегодняшний день является обработка информации, поступающие из различных источников и вырабатывающих алгоритмы реакций и ответных действий. При обработке данных, поступающих из сенсоров, камер и др., применение нейронных сетей в робототехнике показывает явные превосходства перед простыми алгоритмами. Особенно данное превосходство видно при работе с большими данными, требующие быструю обработку, в правильной выборке и анализе положения условных объектов, привлеченных в процессе на конкретный момент действий.

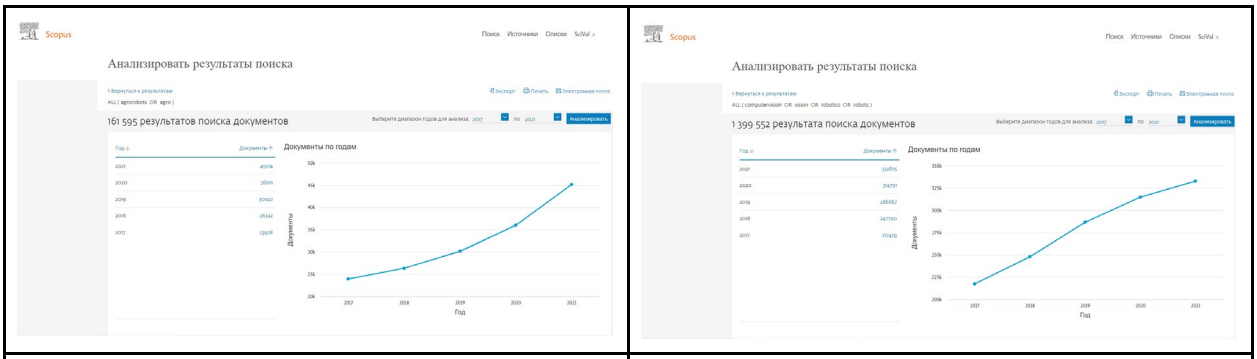
Внедрение роботов для частичной или полной автоматизации предприятий, множеств секторов разных отраслей также является актуальным проблемой на сегоднящний день. Областью применения роботов может быть как индустрияльный сектор, так и сектор оказания услуг, цифровизации и обработки информации. По исследованиям ученых и аналитиков по международной торговле и экономистов агросектор для Казахстана является перспективным сектором промышленности [6] для развития и внедрения роботов с искусственным интеллектом. Внедрение инновации в агропромышленную отрасль, также имеет экономическое обоснование, так как с ростом популяции или деградации земель, износом техники, с внедрением новых методов рано или поздно у государства встанет вопрос о модернизации технологии в области агропромышленности. Естественно, в зависимости от того, что, какие корпорации будут поставлять робототехнику в аграрный рынок Казахстана, могут быть предложены различные условия применения и ценообразования.

Как правило, ожидается, что продвинутая, еще и с искусственным интеллектом, сельскохозяйственная автоматизация позволяет увеличить производительность труда во многих случаях, повысить урожайность и качества возделывания и сбора, а также контролировать над экологическими последствиями. Несмотря на эти преимущества, многочисленные сельскохозяйственные задачи по-прежнему обрабатываются вручную, что негативно влияют на общую тенденцию развития как в техническом оснащении, так и в продуктивности сельскохозяйственных угодий и эффективности агропромышленного комплекса. Эти факторы обоснованы не готовностью агропромышленного комплекса к внедрению инновационных технологий, в том числе роботизированных комплексов, а также не совершенности самих роботов, ограниченности функционалов по применимости. Критическим аспектом успешных сельскохозяйственных роботов является их способность обрабатывать сенсорную информацию и, в

частности, их способность анализировать и интерпретировать визуальный ввод. Действительно, путем установления связи между визуальными данными и надлежащим принятием решения машинные алгоритмы могут облегчить многочисленные операции и продвигать сельскохозяйственную автоматизацию на новые уровни. Однако, проблемы, связанные с машинным зрением [7], встречающиеся в сельскохозяйственной среде очень много: объекты различных цветов, форм, размеров, текстур и отражательных свойств; постоянно меняющееся освещение и теневые условия; тяжелые окклюзии; являются лишь частью проблем, с которыми машинное зрение должно столкнуться. Неудивительно, что нынешний успех в создании роботов для сельского хозяйства все еще ограничен, тогда как именно модернизация и роботизация агропромышленного комплекса являются основой ее эффективности, как одной из важных направлении развития.

Одной из важных проблем ориентации роботов в пространстве является минимизация ошибок восприятия окружающей среды [8], максимизация функционалов распознавания объектов окружающей среды в математическом плане можно решить с помощью метаэвристических методов, позволяющие расширить функциональные возможности робототехнических систем с искусственными интеллектами [9]. Рост опубликованных работ по метаэвристическим подходам, основанных на локальном поиске с чередующимися окрестностями за последние годы постоянно растут. Например, по данным ScienceDirect (www.sciencedirect.com), можно заметить, что если 2003 г. число публикуемых каждый год работ не превышало 100 единиц, то 2019 году их количество стало более 2000 публикаций, и предполагается, что далее она будет только расти. Основные исследования в данной области принадлежат зарубежным ученым. В последние годы существенный вклад в развития теории и практики применения метаэвристических методов с чередующимся окрестностями внесли N.Mladenovic and P.Hansen (Variable Neighborhood Search) и их ученики. Развитие данного направления позволило разработать теоретические основы для многих оптимизационных задач в областях проектирования, планирования и интеллектуального анализа данных, машинного интеллекта и многих других. Тысячи значимых научных работ были опубликованы в журналах с высоким рейтингом, таких как Computers and Operations Research, European Journal of Operational Research, Electronic Notes in Discrete Mathematics, Information Sciences, Applied Mathematics and Computation, Applied Mathematical Modelling и другие. Исследования, проводимые исследователями разных стран, показывают, что внедрение искусственного интеллекта будет способствовать дальнейшему развитию автоматизации агропромышленного сектора. Статистика по опубликованным статьям за период с 2017 по 2021 год в базе данных Scopus (www.scopus.com) с ключевыми словами “agrorobots” или “agro”, которая представлена на рисунке 2 и показывает тенденцию роста на данную тему, так как всего за 5 лет было обпуликовано 161595 документов, в то время как по ключевым

словам “computervision” или “vision” или “robotics” или “robots” было опубликовано 1 399 552 документов. Исходя из того, что с каждым годом количество опубликованных статей увеличиваются, можно сделать вывод что уровень изучения данной темы актуальна по сей день.



а)	б)

Рисунок 2 — Статистика статьей (Scopus)

На сегодняшний день в Казахстане нет крупных предприятий, широкомасштабно внедряющие интеллектуальные системы в практику, тем не менее, в ходе обзорного исследования было определено, что данная проблема также актуальна и можно увидеть картину, где каждый исследователь, неважно в научных или коммерческих целях, пытается реализовать свой вклад в данное направление.

Естественно, в Казахстане также занимаются созданием робототехнических систем с элементами искусственного интеллекта коллективы научно-исследовательских институтов и университетов, такие как Институт информационных и вычислительных технологий, Институт механики машиноведения им. академика У.А. Джолдасбекова, Казахский национальный университет имени аль-Фараби, Казахский национальный исследовательский технический университет имени К.И. Сатпаева, Назарбаев Университет. Мы можем встретить полезные статьи отечественных ученых в базе зарубежных индексируемых и отечественных журналов, таких ученых как М.Н. Калимолдаева, А.К. Тулешова, Е.Н. Амиргалиева, Р.Р. Мусабаева, А.Н. Ешмухамедова, К. Ожикенова, а также ученых из Назарбаев Университет.

В то время, как машинное зрение в системах агропромышленных роботов еще не достигло полного потенциала, многие приложения уже разработаны для различных задач по обслуживанию садов и теплиц. Среди них автономная навигация и прохождение препятствия, прецизионное и селективное распыление, оценка урожайности, посадка саженцев и оценка качества применимости роботов [10-15]. Не только в заубежных странах, да и в нашей стране во многих важных документах и решениях Правительства, Министерства сельского хозяйства, а также потребителя установлены, что роботы в агропромышленности играют значительную роль для повышения

эффективности функционирования. Только в начале 2017 года около одной пятой всего салата, выращиваемого в США, было истончено с помощью LettuceBot. Данную разработку в США внедряют несколько объединенных производителей агротехнологий во главе разработчика Джордж Херауд (Jorge Heraud), который и сам открыл компанию Blue River Technology [16]. Это лишь один пример внедрения готовой продукции робототехники в агропромышленность. Что касается отечественной деятельности, на сегодняшний день широко распространяются внедрение технологии развитых стран в агропромышленность. Многие из них частично автоматизированы. Не секрет, что многие агропромышленные центры все еще используют человеческий труд, в решениях многих проблем, связанных с прямыми контактами с растениями, а также по уходу за сельскохозяйственными угодьями. В этом плане агророботы, оснащенные с алгоритмами анализа и обработки изображений и сенсорных данных, а также машинного обучения могут быть в последующем использованы, адаптированы и для других отраслей АПК.

Почти 90% оборудования, используемого в настоящее время в Казахстане, находится в конце своего жизненного цикла и нуждается в замене. Тракторы, используемые более 10 лет, составляют 94% всего парка, а уборочные комбайны в аналогичном состоянии составляют 77%. По состоянию на 1 января 2019 года сельхозпроизводители имеют 147 000 тракторов, 79 400 сеялок и 249 000 единиц для обработки почвы на предстоящий сезон посадки. Естественно, со стороны государства принимаются соответствующие решения по обновлению техники и оборудования. Так, импорт сельскохозяйственного оборудования в Казахстане субсидируется в размере 25% от стоимости, при этом финансовый лизинг предоставляется под ставку в 10%. Уровень обновления техники за последние 5 лет колебался от 3 до 4,9%, но этот показатель должен достигать 6–8% ежегодно [17, 18]. Учитывая приведенные цифры, становится ясным, что обновление парка машин либо введение новых технологий, позволяющих более разумное и экономичное использование расходных материалов, стоит в приоритете.

Основываясь на опыте других стран, можно сделать выводы, что введение автоматизированных или полуавтоматизированных систем позволит увеличить объем производства с использованием того же или меньшего количества человеческих ресурсов [19-22].

Внедрение в АПК разрабатываемых образцов агро-робота, оснащенного интеллектуальной системой компьютерного зрения, выполняющего сборку позволит обеспечить условия для быстрого и правильного сбора урожая томатов, обеспечит высокую производительность труда, даст возможность мониторингу и выявлению каких-либо изменений в процессе роста растений за счет поэтапного и хронологического самообучения, все это конечным итогом должно отразиться на новом уровне развития агропромышленной структуры страны в целом.

Полученные результаты дают стимул развивать науку и технику, как в научно образовательных целях, так и на уровне модернизации крупных и средних предприятий. Научные результаты внесут хороший вклад в локальную и глобальную науку, так как разработка подобных интеллектуальных агро-роботов на основе компьютерного зрения и машинного обучения, выполняющих сборку томатов является актуальной задачей в наши дни.

В Казахстане данное направление исследований находится на начальной стадии развития, нет отечественных образцов агро-роботов, выполняющих сборку томатов с применением компьютерного зрения и машинного обучения.

Существующие аналоги данной разработки зарубежных производителей: BoniRob от компании Deepfield Robotics (Германия), ecoRobotics, Ecorobotics, Швейцария; FarmWise, FarmWise, США; HortiBot, Дания; Ladybird, Сиднейский университет, Австралия; Naïo Technologies, Франция; Oz, Naio Technologies; RoboTrac, Hannes Zeeberg.

Интеллектуальный робот соответствует проектам Государственной программы развития АПК Казахстана до 2021 и 2027 года [23], а именно, приоритетам «Точное земледелие» и «SMART фермы» и будет следующим шагом по внедрению автоматизации в агропромышленность страны и значительно уменьшит материальные расходы химических средств, энергетических затрат, время по физическому труду, повысит внимательность и аккуратность по уходу за растениями, даст воздействовать более быстрому внедрению новых данных в базу и получению оттуда информации по отчетности продукции и трудовых этапов, уменьшит риск заболеваний растений при контакте с человеком, возможно приносящим их из за территорией комплекса.

Томат – это продукт, пользующийся большим спросом во всем мире [24], и его потребление постепенно увеличивается с каждым годом [25]. Уборка томатов вручную трудоемка, требует много времени и неэффективна, что делает ее нецелесообразной в крупных хозяйствах. Кроме того, помидор очень мягкий и склонен к появлению синяков, что затрудняет сбор урожая и процесс захвата. Одной из основных проблем в сельскохозяйственном секторе является рост стоимости рабочей силы и нехватка рабочей силы, поскольку сельскохозяйственные работы не пользуются популярностью среди молодежи. Таким образом, роботизация уборки томатов может стать одним из наиболее эффективных решений для устранения упомянутых выше проблем. Однако при рассмотрении вопроса о роботизации еще предстоит решить многие технологические проблемы. Это исследование предлагает один из подходов к роботизации уборки томатов.

В данной работе мы исследуем обработку визуальных данных, поступающих из стереокамер робота, так как использование камер в качестве распознавания окружающей среды, машинное зрение являются самыми

широкоиспользуемыми, а также дающими большее количество данных в информационном потоке методами.

Уровень изучения проблемы. Применение методов машинного обучения на основе нейронных сетей в робототехнике имеет очевидное приемущество по вопросам точности распознавания объектов и ориентации в пространстве роботов перед простыми алгоритмами, ориентированные на показания датчиков, но эти вопросы все еще формулируются для поиска оптимальных путей реализации и их внедрения. Последние труды ведущих специалистов в данной области подтверждают, что сегодняшние вычислительные ресурсы позволяют подобные изучения, однако мы не можем утверждать что тот или иной алгоритм всегда применим к одной и той же задаче, то есть множество решений являются эвристическими и не имеют однозначного решения.

Принимая во внимание данные условия мы делаем вывод, что изучение в данной области остается открытым.

Цель диссертационной работы. Разработка эффективной модели и технологий распознавания образов, компьютерного зрения и машинного обучения предназначенного для выполнения распознавания и сборки томатов.

Задачи исследования:

- Выполнить сравнительный анализ методов машинного обучения классификации объектов(томатов) с оценкой качества на основе вычислительного эксперимента;
- Разработать модифицированную архитектуру для сверточной нейронной сети с оценкой качества распознавания изображений;
- Разработать параллельный алгоритм для процессов обработки изображений с вычислением трехмерных координат объектов;
- Адаптировать разработанную систему, позволяющей определить локализацию в пространстве исследуемого объекта с целью внедрения в агроробот, предназначенного для сборки томатов.

Объектом исследования являются робототехнический комплекс с компьютерным зрением и теплица выращения томатов.

Методы и предметы исследования – нейронные сети, алгоритмы распознавания объектов в режиме реального времени, теория компьютерного зрения и искусственных нейронных сетей, технология разработки программного обеспечения.

Научная новизна. На основе архитектуры сверточных нейронных сетей была разработана новая модифицированная архитектура нейронной сети, с параллельной обработкой процессов распознавания томатов, позволяющая увеличить скорость обработки в два раза и улучшить точность на 3 %.

Теоретическая и практическа значимость. Теоретическая значимость полученных результатов заключается в модифицировании существующей архитектуы нейронной сети для распознования объектов и

раобработки паралельного алгоритма для обработки графической информации. Практическая ценность полученных результатов заключается в разработке программы распознования томатов и вычисления их трехмерных координат для робота сборщика на основе положений, вынесенных на защиту

Апробация работы. Основные результаты диссертационной работы доложены и обсуждены на международных и зарубежных научных конференциях, научных семинарах:

– 5th International Conference on Mechanics and Mechatronics Research (Tokyo, 2018);

– III Международная научно-практическая конференция «Информатика и прикладная математика» посвященная 80-летнему юбилею профессора Бияшева Р.Г. и 70-летию профессора Айдарханова М.Б. (Алматы, 2018);

– IV международная научно-практическая конференция «Информатика и прикладная математика», посвященная 70-летнему юбилею профессоров Биярова Т.Н., Вальдемара Вуйцика и 60-летию профессора Амиргалиева Е.Н. (Алматы, 2020);

– 2021 IEEE International Conference on Smart Information Systems and Technologies (Нур-Султан, 2021.г);

– Научном семинаре лаборатории искусственного интеллекта и робототехники Института информационных и вычислительных технологий КН МОН РК;

– Научном семинаре кафедры информатики факультета информационных технологий КазНУ имени аль-Фараби.

Результаты диссертационного исследования опубликованы в 12 работах. Из них 4 статьи в журналах, рекомендованных Комитетом по контролю в сфере образования и науки МОН РК, 4 статей в международных научных изданиях, входящих в базу данных Web of science и Scopus, 4 работы в материалах международных и республиканских конференций, 1 авторское свидетельство по результатам практического внедрения диссертационных исследований.

Связь темы с планами-исследовательскими программами. Предоставленные результаты получены при выполнении следующх проектов ИИВТ КН МОН РК (источник финансирования Комитет науки МОН РК):

– грантового финансирования (ГФ) КН МОН РК AP05132648 “Создание вербально-интерактивных роботов на основе современных речевых и мобильных технологий” в 2018-2020 годы;

– грантового финансирования (ГФ) КН МОН РК AP08857573 “Разработка интеллектуальных информационных технологий на основе машинного зрения и распознавания образов с построением мобильного робота по обслуживанию сельхоз угодий” в 2020-2022 годы;

Основные положения, выводимые на защиту. Предложенная система компьютерного зрения, реализованная на основе параллельной обработки изображений и усовершенственной архитектуры со сверточной

нейронной сетью с представлением трехмерных координат исследуемого объекта показал в результате вычислительных экспериментов высокую эффективность по скорости обработки изображении (количество кадров) и точности по сравнению первоначальной архитектурой нейронной сети по распознаванию объектов (томаты).

Личный вклад исследователя. Личный вклад исследователя заключается в обзоре и оценке результативности методов и технологий машинного зрения, модификации архитектуры сверточной нейронной сети, разработке параллельного алгоритма обработки изображений, в оснащений многозвенного робота машинным зрением, а также проведении вычислительных экспериментов.

Структура и объем дисертации. Общий объем диссертационной работы – 99 страниц. Работа состоит из введения, 4 разделов, заключения, списка использованной литературы и 3 приложений.

1. ПРИМЕНЕНИЕ МЕТОДОВ ЦИФРОВОЙ ОБРАБОТКИ ИЗОБРАЖЕНИИ ДЛЯ СЕГМЕНТАЦИЙ И РАСПОЗНОВАНИЯ ТОМАТОВ

С увеличением вычислительных ресурсов и мощностей обработки данных компьютерное зрение применяется во многих сферах, связанных со зрительными операциями человеческой деятельности, таких как, мониторинг, автономные средства, виртуальная реальность и др.[26–29], причем не только в научно-теоретических взглядах, а в конкретных практических применениях разного уровня и характера действий.

Внедрение нейронных сетей в компьютерное зрение также принесло свои плоды, особенно в методах обработки изображений. Во многих научных трудах, где проводился обзор, часто встречаются сравнительные анализы двух или более методов использования нейронных сетей при распознавании или обработке изображения, но очень сложно найти сравнения алгоритмов применяющих простой программный подход и подход с использованием нейронной сети, так как их сравнение хоть и допустимо, но считается не совсем корректным.

В данной диссертационной работе мы используем компьютерное зрение в качестве метода и аппаратной реализации для распознавания роботом агропродуктов – томатов, для манипуляционных действий с данными объектами.

Итак, мы приходим к проблеме, как обработка изображений, которая требует своего внимания в данной научной работе.

Обработка изображения – это форма обработки информации, предоставленной изображением на входе, например, фотография или видеокадр. Цифровая обработка это манипуляция изображениями с помощью цифровых вычислительных машин. Важно отметить, что в последние десятилетия цифровая обработка активно используется в медицине, компьютерных играх, киноискусстве, в геологии, биоинформатике, дистанционном зондировании и т. д. На сегодняшний день мультимедийные системы сильно зависят от цифровой обработки изображений [30].

С широким использованием в промышленности, искусстве, медицине, космосе, также в различных информационных технических системах, существуют различные методы машинной обработки изображений. В данном разделе представляется обзор методов фильтрации и сегментации изображений для решения таких задач, как разделение изображений на области, улучшенное представление и выделение контуров, границ и других отличительных признаков объектов.

1.1 Эксперименты по сегментации объектов в цветовой модели HSV

В ходе исследования был проведен сравнительный анализ цветового пространства (HSV), в котором меняются оттенок, насыщенность и интенсивность пикселя рассматриваемого изображения в цветовым пространстве RGB. Было использовано два основных фильтра для сегментации томата: первый фильтр определяет форму объекта, а второй фильтр использован для определения цвета. В этом исследовании мы выбрали цветовую модель HSV для сегментации томатов по цвету. После определения доминирующего свойства, основанных на указанных трех параметрах, извлекается конкретный пиксель, необходимый для сегментации изображения. Данная модель цветового пространства очень близка к человеческому восприятию по сравнению с другими моделями, такими как RGB или CMYK.

В работе [31] успешно применили данную модель HSV в качестве сравнительного образца, при благополучных условиях, когда снимки для обработки изображений были сделаны между 9:00-10:00 часами утра, в ясный солнечный день, и ее эффективность оказалась 81,6%, что в принципе показывает удовлетворительный результат. В данном исследовании мы постарались использовать снимки разного качества, сделанные в разные время суток и разных погодных условиях (солнечный, облачный).

HSV модель визуально можно представить в виде цилиндра, где интенсивность будет расположена на центральной вертикальной оси, а значение параметров оттенка определяются по углу наклона в диапазоне 0-360º. Для красного, зеленого и синего цветов соответствуют значения 0, 120º, 240º, а за определение глубины или чистоты цвета отвечает параметр насыщенности, который измеряется как радиальное расстояние со значением от центра до внешней поверхности, исчисляющийся от 0 до 1. Если записать в процентных значениях, то для значения 0 будет соответствовать черный, а для 100% белый цвет. С помощью понижения насыщенности каждый цвет можно преобразовать в оттенок серого. Для низких значении насыщенности цвет может соответствовать серому [32].

Ниже приведен расчет для преобразования кубической модели 3D RGB в цветовую модель HSV (см. рисунок 1.1) числовым способом (1.1).

$$h = \begin{cases} 0, & if\ max = min \\ \left(60^0 \times \dfrac{g-b}{max-min} + 0^0\right) mod 360^0, & if\ max = r \\ 60^0 \times \dfrac{b-r}{max-min} + 120^0, & if\ max = g \\ 60^0 \times \dfrac{r-g}{max-min} + 240^0, & if\ max = b \end{cases} \quad (1.1)$$

$$s = \begin{cases} 0, & if\ max = 0, \\ \dfrac{max-min}{max} = 1 - \dfrac{min}{max}, & otherwise \end{cases}$$

$$\vartheta = max$$

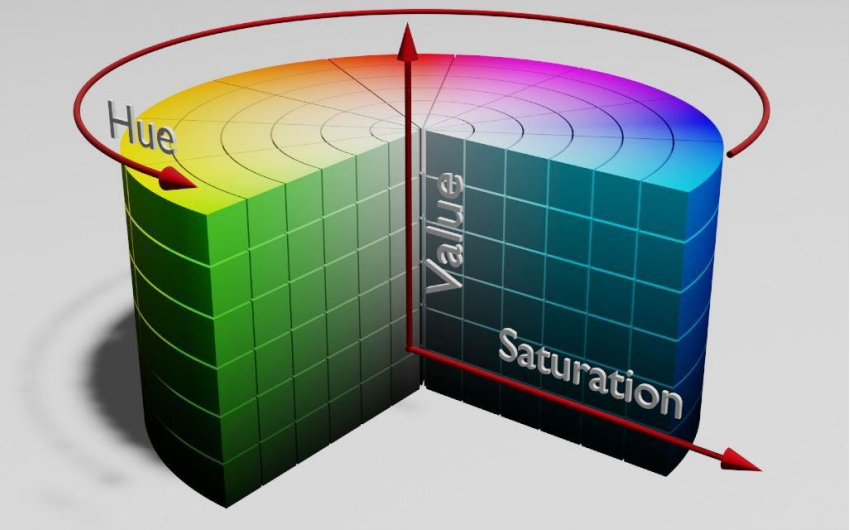


Рисунок 1.1 — Визуальное представление цветовой модели HSV (оттенок, насыщенность, интенсивность)

Для компьютерного зрения робота-сборщика томатов, в данной задаче цель – это сегментация томатов по цветам (зрелый плод – красный и незрелый – зеленый), где для этого изображение необходимо преобразовать в модель HSV. Преимущество данной модели в том, что оттенок цвета задаётся одной координатой hue. Он предоставляет возможность проще отсеивать нужные цвета с изображения. Saturation и value позволяют снизить и уменьшить влияние слабой освещённости в изображений. Это существенно повышает эффективность определения по цветам томата его спелости. На рисунке 1.2 можно увидеть разницу преобразования изображения из цветового пространства BGR на HSV:

Рисунок 1.2 — Преобразования изображения из цветового пространства BGR на HSV

Далее необходимо определить нижний и верхний диапазон значений для нужного цвета, которые в свою очередь состоят из трех значений: тона, насыщенности и интенсивности. После этого создаем маску изображения установив порог и добавив необходимый диапазон значений.

Разница распределения интенсивности: по оси абсциссы значения насыщенности пикселей, до 7000, а ось ординаты соответствует количеству пикселей, до 250 (см. рисунок 1.3). На первом рисунке(левая часть рисунка 1.3) значения интенсивности былы высокими на участке пикселей от 0 до 100, а после преобразования значения интенсивности от 0 до 50 были низкие. Самое максимальное значение можно заметить на участке 50. Также, если на первом изображении значения интенсивности планомерно уменьшаются начиная со значения 50, то на втором начиная от 100 до 250 ведут однообразно.

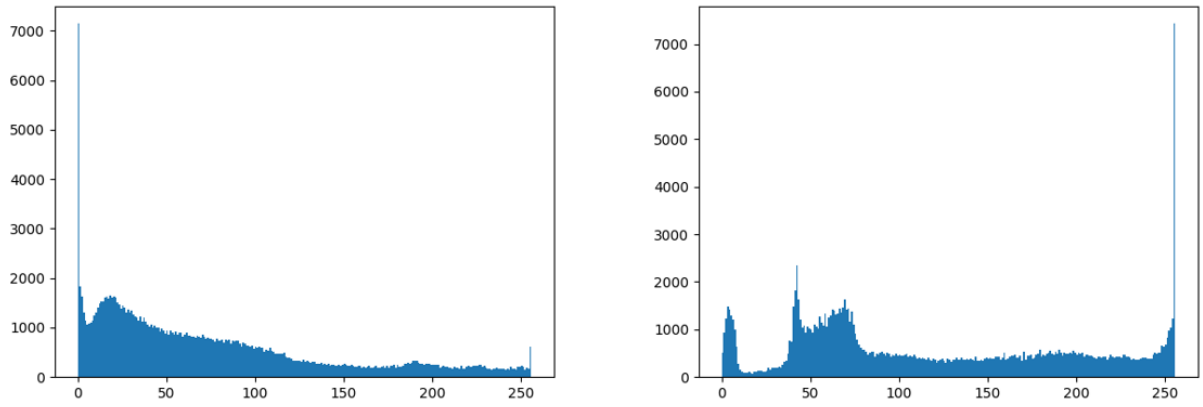


Рисунок 1.3 — Гистограмма изображении в формате RGB и HSV

Далее проводим морфологические преобразования с помощью библиотеки OpenCV. Данные операции необходимы для удаления белых шумов, отделения и установления границы двух связанных объектов. Морфологические операции выполняются с двоичными изображениями. Для этого нужно на входе дать исходное изображение и второе изображение,

которое называется структурным элементом. Структурный элемент или ядро проводит операции по сегментации объектов по отдельности. Существует несколько морфологических операторов, и чтобы выбрать подходящую, следует рассмотреть их все. Результаты морфологического преобразования исходного избражения приведены на рисунке 1.4.

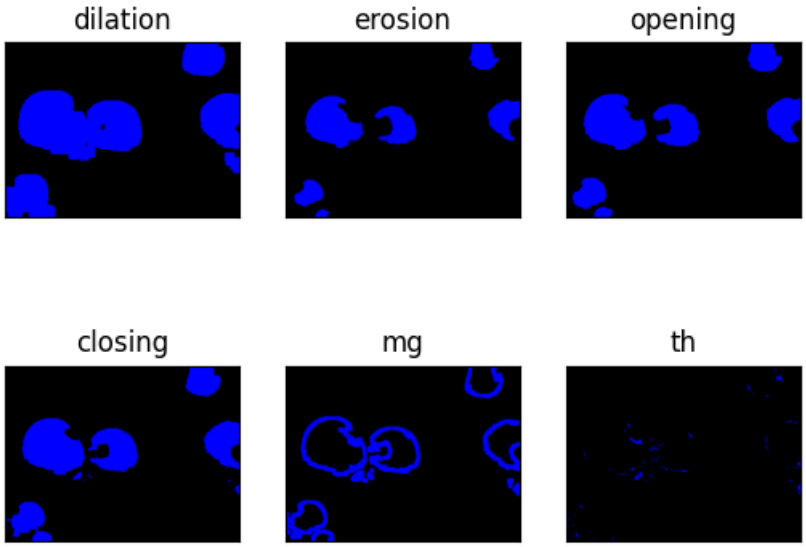


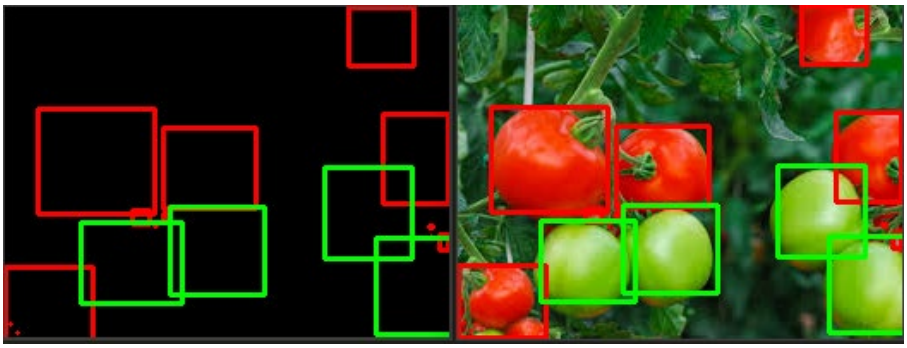
Рисунок 1.4 — Морфологическое преобразование исходного изображения

Первое изображение на рисунке 1.4 – результат работы операции Dilation (Расширение), которая увеличивает белую область на исходном изображении и увеличивает размер объекта переднего плана. Это также полезно для соединения разделенных частей объекта. В данном случае видно, что действительно размер объектов переднего плана значительно увеличился. Однако, нам это ненужно, так как наша главная задача – разделить границы объектов таким способом, чтобы сегментировать их по отдельности. Второе изображение – Erosion. Данная операция размывает границы объекта переднего плана, что является полной противоположностью первого. Главным преимуществом которого является отделение двух областей, что и необходимо. Все же трудно не заметить, что данная операция хоть и отделяет два объекта, она также уменьшает их размер. Становиться понятно, это тоже совсем не то, что нужно. Третье изображение – Opening (открытие), принцип работы которой также как у эрозии, но после этого увеличивает белую область в рассматриваемом изображении. Особенностью такой операции является удаление шумов. Четвертое – Closing (закрытие), что происходит в обратном порядке по отношению к closing. Главным преимуществом является удаление небольших отверстий внутри объектов переднего плана и ненужных пикселей. Пятое изображение Морфологический градиент – это результат работы разницы Dilation и Erosion. В исходном изображений можно увидеть, что получили только границу нашего объекта. Последнее изображение Tophat – результат работы

между исходным изображением и операцией Closing. На рисунке мы видим, что данная операция не совсем дает хороший результат.

Получив все результаты, было решено выбрать для нашей сегментации операцию Closing. После того, как мы могли сегментировать объекты по цветам, была написана функция для каждого необходимого нам цвета (красный и зеленый). После сегментации мы можем определить их контур и с помощью функции boundingRect ограничить их прямоугольником.

Одновременно определить на входном изображений два цвета невозможно, так как во время обнаружения одного цвета все остальные преобразуются в черный фон и переходят на задний план. Для решения данной проблемы, были созданы два изображения, одно из которых полностью состоит из черных пикселей, как показано на рисунке 1.5а, а другое является копией входного изображения, как показано на рисунке 1.5б. Оба они идентичны по размеру со входным (см. рисунок 1.5а). Первое изображение теряет свою необходимость, но ее наличие поможет для сравнения и визуализации. После этого, во время обнаружения каждого цвета, записываем координаты ограничивающего прямоугольника и также рисуем их на созданных двух изображениях (см. рисунок 1.5б).



а) б)

Рисунок 1.5 — Определение объектов по цветам и создание ограничивающего прямоугольника

Данное решение позволяет определить объекты по цветам без использования нейронных сетей с экономией вычислительных ресурсов.

1.2 Анализ и применение Гауссовской фильтрации для постобработки по уменьшению уровня шума в изображении

Метод, который интенсивно изучался в области обработки изображений – Гауссовская фильтрация [33]. Математическая модель Гауссова функции для фильтрации указано в формуле 1.2.

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}} \quad (1.2)$$

где x — расстояние от 0 по горизонтальной оси, y — расстояние 0 по вертикальной оси, а σ — стандартное отклонение распределения Гаусса.

Как оттенки серого, так и цветные изображения могут содержать много шума или случайных изменений яркости или оттенка между пикселями. Пиксели на этих изображениях имеют высокое стандартное отклонение, что просто означает, что в группах пикселей есть много вариаций. Поскольку фотография двумерна, размытие Гаусса использует две математические функции (одну для оси x и одну для y) для создания третьей функции, также известной как свертка. Эта третья функция создает нормальное распределение этих значений пикселей, сглаживая некоторую случайность. Величина сглаживания зависит от размера выбранного радиуса размытия. Каждый пиксель получит новое значение, установленное на средневзвешенное значение окружающих его пикселей, причем больший вес будет отдан более близким, чем тем, которые находятся дальше.

Данный поход эффективен при обработке изображений с несколькими разрешениями [34]. Использование фильтра Гаусса в этапе предварительной обработки приведет к смещению положения краев, исчезновению краев и фантомным краям. Поэтому авторы работы [35] рассматривая различные методы решения этих проблем, предложили алгоритм, адаптированный к характеристикам шума и к локальной дисперсии сигнала.

В статье [36] предложен алгоритм устранения помех смешанного изображения, основанный на гауссовом фильтре. В данной работе Гауссовская фильтрация была использована для фильтрации изображения шума и получения эталонного изображения, затем принимается как эталонное изображение, так и изображение шума в качестве входных данных для функции ядра диапазона двустороннего фильтра.

Поскольку процесс сегментации может быть очень дорогостоящим и трудоемким, важно выбрать правильный метод для правильной фильтрации изображения. В [37] исследуются качественные и количественные эффекты свертки функции Гаусса с изображением. Результаты показали, что метод размытия по Гауссу следует использовать на изображениях с высоким уровнем шума и с функцией Гаусса с малой дисперсией, тогда как функция Гаусса с большей дисперсией более уместна при сегментации изображений.

Итоги сглаживания изображения с применением 2D-фильтра Гаусса в [38] показали наилучшие результаты по сравнению с другими фильтрами.

Для получения сегментации и разделения изображения на передний и задний фоны мы избавились от шумов, используя Гауссову фильтрацию, которая является результатом работы ядра Гаусса, где предварительно указывается ширина и высота ядра. Она применяется после получения сегментации и до обработки сегментированного изображения морфологическим преобразованием. С помощью Гауссова фильтрации были убраны лишние шумы и получены только интересующие нас объекты (см. рисунок 1.6).

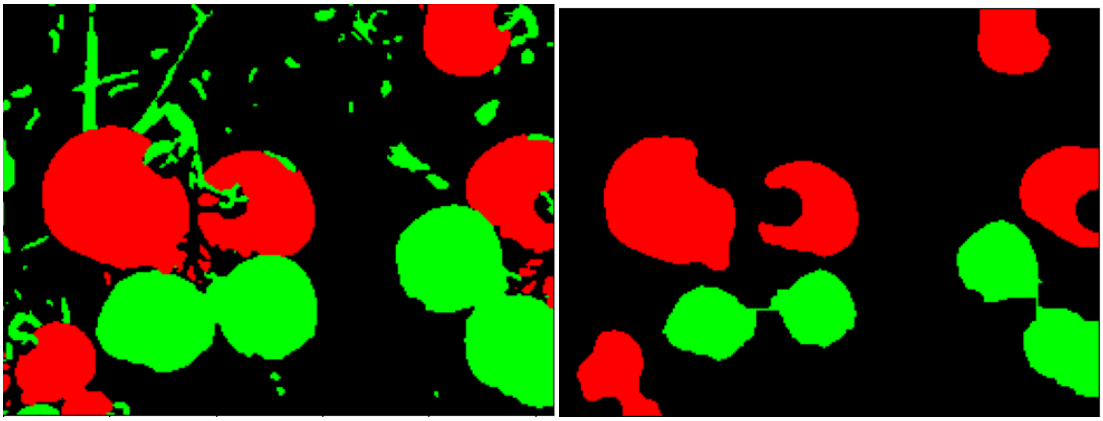


Рисунок 1.6 — Сегментация с использованием Гауссовой фильтрации

Однако, все же есть участки, на которых были соединены объекты. Для решения данной проблемы используется медианная фильтрация пикселей. Медианная фильтрация вычисляет медианное значение всех пикселей с помощью ядра и заменяет центральный пиксель этим медианным значением.

1.3 Анализ и применение медианной фильтрации для постобработки по уменьшению уровня шума в изображении

Медианная фильтрация (MF) — это каноническая операция обработки изображений, действительно полезная во многих практических приложениях. Наиболее привлекательной особенностью MF является его устойчивость к шуму и ошибкам в данных, но поскольку метод требует сортировки значений окна, он требует больших вычислительных затрат [39].

Принцип работы классической медианной фильтраций для N четного количества входных измерений можно описать следующей формулой 1.3.

$$MF = \frac{a(\frac{N}{2}) + a(\frac{N}{2} + 1)}{2} \tag{1.3}$$

где N четное число, то фильтр вычисляет среднее арифметическое между двумя центральными элементами отсортированного массива. Нужно отметить, что входные данные сортируются как в порядке возрастания, так и в порядке убывания.

Если N — нечетное число, то фильтр вычисляет центральный элемент отсортированного массива 1.4:

$$MF = \frac{a(\frac{N}{2} + 1)}{2} \tag{1.4}$$

В исследовательской работе [39] предложено новое понимание возможностей MF, основанное на оптимальном значении разбивки (BV) медианы. Результаты эксперимента объединили как реальные, так и синтетические данные, включая пример RGB и экспериментальное вычисление скорости усилия перехода. Таким образом, различия между стандартными и основанными на BV версиями анализируемых алгоритмов MF подчеркиваются и объективно оцениваются количественно. В целом, алгоритмы на основе BV заметно превосходят соответствующие стандартные версии как для одноканального, так и для многоканального изображения.

Медианная фильтрация обладает хорошими эффектами снижения шума. Алгоритм, предложенный в [40] использует корреляцию изображения для обработки особенностей фильтрующей маски по изображению. Он может адаптивно изменять размер маски в соответствии с уровнем шума маски. Результаты экспериментов показали, что алгоритм уменьшает шум и сохраняет детали изображения. Сложность алгоритма уменьшена до O(N), и производительность шумоподавления эффективно улучшилась.

Авторы работы [41] предложили два новых алгоритма для адаптивных медианных фильтров на основе двух типов моделей изображений, искаженных импульсным шумом. Первый, называемый адаптивным медианным фильтром на основе ранжированного порядка (RAMF), основан на тесте на наличие импульсов в самом центральном пикселе, за которым следует тест на наличие остаточных импульсов на выходе медианного фильтра. Второй, называемый адаптивным фильтром среды на основе размера импульса (SAMF), основан на определении размера импульсного шума.

Показано, что RAMF превосходит нелинейный средний фильтр L, устраняющий положительные и отрицательные импульсы при одновременном сохранении резкости; SAMF превосходит адаптивную схему Lin, поскольку он проще и эффективнее устраняет высокую плотность импульсного шума, а также неимпульсный шум и сохраняет мелкие детали. Моделирование на стандартных изображениях подтверждает, что эти алгоритмы превосходят стандартные медианные фильтры.

Основным преимуществом медианной фильтрации от гауссова фильтрации заключается в том, что значение центрального пикселя заменяется существующим значением пикселя в данном изображении. На рисунке 1.7 представлены результаты работы:.

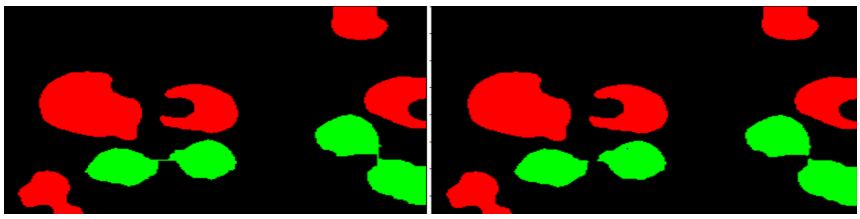


Рисунок 1.7 — Результат работы медианной фильтрации

Как видно, мы разделили объекты, которые до этого были соединены. Таким образом можно утверждать, что медианный фильтр работает эффективно и его можно использовать при распознавании томатов.

Выводы по первому разделу

В первом разделе представлены методы фильтрации, которые имеют широкое применение при обработке изображений. Согласно обзору, проведенному по исследованию фильтров цифровой обработки изображений, можно сделать следующие заключения:

- Модель цветового пространства HSV применяется для сегментации изображения по цвету. Сделанный анализ показывает, что с помощью данной сегментации возможно получать более точную идентификацию объектов по сравнению с другими алгоритмами.
- Фильтр Гаусса сглаживает неравномерные значения пикселей в изображении, вырезая крайние выбросы. Гауссовские фильтры могут быть реализованы очень эффективно, поскольку гауссовы функции разделимы. Двумерная свертка по Гауссу может быть выполнена путем свертки изображения с помощью одномерного гауссова, а затем свертки результата с помощью того же одномерного фильтра, ориентированного ортогонально гауссову, использованному на первом этапе. Таким образом, объем вычислений, необходимый для гауссова фильтра, растет линейно в начале ширины маски фильтра вместо квадратичного роста. Одна вещь, которую следует иметь в виду при применении фильтра Гаусса, заключается в том, что большая интенсивность размытия приводит к снижению резкости.
- Медианный фильтр сохраняет границы и контуры объектов без искажений, подавляет некоррелированные импульсные помехи и детали маленького размера. Свойство нелинейности является преимуществом медианной фильтрации, так как данный фильтр выдает наименьшее значение среднеквадратической ошибки, по сравнению с линейными фильтрами.

В следующих разделах, описываются другие подходы решения задач распознавания объектов с помощью алгоритмов машинного обучения и нейронных сетей.

2. РАСПОЗНАВАНИЕ ТОМАТОВ С ПРИМЕНЕНИЕМ АЛГОРИТМОВ МАШИННОГО ОБУЧЕНИЯ

2.1 Сравнительный анализ и оценка алгоритмов машинного обучения для распознавания объекта

В исследовании [17] был разработан алгоритм сегментации для управления манипулятором робота по сбору спелых помидоров с помощью системы машинного зрения. Исследователями использовалась система зрения для получения изображений с растения томата. Алгоритм распознавания адаптируется к условиям освещения теплицы. Использовались 110 цветных изображений томата в условиях тепличного освещения. Разработанный алгоритм работает в два этапа: (1) путем удаления фона в цветовом пространстве RGB и последующего извлечения созревшего помидора с использованием комбинации пространств RGB, HSI и YIQ и (2) локализации созревшего помидора с использованием морфологических особенностей изображения. По результатам общая точность предложенного алгоритма составила 96,36%.

Одной из уязвимых моментов на пути к надежному распознаванию продуктов в аграрии является снижение влияния основных помех: освещения и перекрытия. Чтобы распознать помидор в кроне растения с помощью недорогой камеры, в работе [18] был изучен надежный алгоритм распознавания помидоров, основанный на нескольких характерных изображениях и слиянии изображений. Два новых характерных изображения, a*-компонентное изображение и I-компонентное изображение были извлечены из цветового пространства L*a*b* и цветового пространства яркости, синфазного, квадратурного (YIQ). Также в работе было принято вейвлет-преобразование для слияния двух характерных изображений на уровне пикселей, которые объединили информацию об особенностях двух исходных изображений. Чтобы отделить целевой помидор от фона, был использован алгоритм адаптивного порога для получения оптимального порога. Окончательный результат сегментации был обработан операцией морфологии для уменьшения небольшого количества шума. В тестах на обнаружение было обнаружено 93% целевых томатов. Это указывает на то, что предлагаемый метод распознавания томатов мы могли бы использовать для роботизированной уборки томатов в неконтролируемой среде с низкой стоимостью.

Исследования, проводимые в [10], где взяты данные из 600 томатов на шести стадиях зрелости и были использованы для теста, коэффициенты поглощения и пониженного рассеяния для плодов томатов были извлечены с помощью пространственно-разрешенной спектроскопии, модели дискриминантного анализа частичных наименьших квадратов (PLSDA) были построены для оценки зрелости томатов. Более того, данная работа хорошо описывает определение каждого этапа развития плода, определяя цветовые

признаки по спектральным частотам. Спектры с пространственным разрешением (SR) для каждого образца томатов были получены с использованием спектроскопической системы с пространственным разрешением в спектральной области 550~1650 нм. Поскольку 30 волокон в зонде SR были расположены симметрично, каждая пара симметричных спектров была усреднена, что дало 15 спектров относительного отражения, покрывающих расстояния между источником света и детектором 1,5~36 мм. Из-за сильного фотопоглощения за пределами 1300 нм, только 550~1300 нм были выбраны для извлечения поглощения и пониженных коэффициентов рассеяния плодов томата. Кроме того, девять спектров SR на пространственных расстояниях от 1,5 до 12,5 мм фактически использовались для анализа свойств поглощения и пониженного рассеяния плодов томата в этом исследовании, поскольку сигнал за пределами 12,5 мм был слишком слабым, чтобы быть полезным. Значения для поглощения и приведенного коэффициента рассеяния были получены с помощью уравнения диффузионного приближения в сочетании с нелинейным обратным алгоритмом. Было определено, что содержание хлорофиллов уменьшается при длине волны 675 нм вместе с увеличением антоциана на длине волны 560 нм по мере того, как помидоры меняют цвет с зеленого на красный. Значения приведенного коэффициента рассеяния неуклонно уменьшались с увеличением длины волны для всех испытанных образцов томатов в спектральной области 550~1300 нм. Результаты классификации сравнивали с использованием μa и μ. Кроме того, стадия созревания томатов оценивалась по цвету поверхности и внутреннему цвету. Результаты показали, что комбинации μa и μ могут еще больше улучшить результаты классификации по сравнению с отдельными μa и μ-спектрами, особенно μa × μ (умножение длины волны двух параметров на длину волны), которые показали степень распознавания 78,5% и 85,5% для внутреннего цвета. и цвет поверхности соответственно. Лучшие результаты классификации были получены для трех стадий спелости с использованием μa и μ's и их комбинаций, а степень распознавания была аналогичной, около 94% для внутреннего и внешнего цвета. Авторы показали, что спектры оптического поглощения и рассеяния могут эффективно классифицировать стадии спелости томатов и предоставили новые средства неразрушающего обнаружения в сельскохозяйственных продуктах.

В другой работе тех же авторов [11] предлагается более новый подход, где частичные наималейшие квадраты дискриминантный анализ и опорные векторы дискриминантного анализа модель. PLSDA спектры SR 15 дали лучшие результаты классификации с точностью 81,3%, в то время как для моделей SVMDA спектры SR 10 имели лучшую точность 86,3%.

Также, авторы в [12] исследовали измерения оптических свойств плодов томатов в диапазоне длин волн с помощью метода пространственно-разрешенного диффузного отражения для оценки плотности, содержания растворимых твердых веществ (SSC) и pH. Спектры пространственно-

разрешенного диффузного отражения томатов, собранных на шести стадиях созревания, были получены с использованием недавно разработанной системы пространственно-разрешенной спектроскопии (SRS), на основе которой были оценены коэффициенты приведенного рассеяния (μ 's) и поглощения (μa). с использованием обратного алгоритма модели теории диффузии. Твердость томатов измеряли двумя эталонными методами, т. Е. Прессованием и проколом, а SSC и pH измеряли с помощью стандартной рефрактометрии и pH-метра. Модели частичных наименьших квадратов (PLS) были разработаны на основе μ′s, μa и их комбинаций для прогнозирования трех параметров качества. Хотя и μ′s, и μa коррелировали с твердостью томата, SSC и pH, лучшие результаты прогноза были получены для умножения μ′s и μa (т. е. μa×μ′s), за исключением максимальной силы прокола. Модели PLS дали хорошие прогнозы максимальной силы сжатия и максимальной силы прокола, наклона и плотности мякоти с коэффициентами корреляции 0,894, 0,915, 0,923, 0,835 соответственно, в то время как они имели плохие прогнозы SSC и pH томата с коэффициентами корреляции 0,623. и 0,769 соответственно. Рассматривая данное исследование, мы узнали, что метод SRS, наряду с коэффициентами поглощения и рассеяния, имеет потенциал для неразрушающего измерения характеристик качества, особенно твердости плодов томатов, которые мы могли бы использовать в дальнейшем для манипулятора сборщика.

В [13] обсуждается обнаружение спелых и переворачивающихся плодов помидоров с помощью компьютерного зрения и методов обработки изображений, где использовали Raspberry Pi и Pi Camera, программное обеспечение Raspbian с использованием Python3. Спелость определяется с помощью OpenCV и HSV. Исследователи собираются это оборудование подключить к Turtlebot, к роботу, который будет перемещаться по полю для обнаружения спелых растений.

В [14] предлагаемая система выполняет классификацию томатов в три этапа с использованием цифровых изображений образцов, снятых на экспериментальной установке, развернутой с использованием микроконтроллера. На первом этапе выполняется бинарная классификация, чтобы отличить томаты от других видов, используя вектор видов, построенный на основе этих изображений. На втором этапе томаты классифицируются на категории спелых и незрелых на основе признака цвета. Затем дефекты плодов идентифицируются с помощью вейвлет-преобразования Габора для сегментации инфицированных областей этих изображений. На третьем этапе определяются три типа дефектов, а именно черные пятна, язвы и меланоза, на основе вектора дефекта, построенного на основе дополнительных цветовых и геометрических элементов. Из-за сложности решения такой нелинейной задачи предлагаемая система реализована в виде каскада из двух машинных классификаторов опорных векторов. Производительность этой системы оценивается с помощью показателей точности, специфичности, чувствительности и точности.

Результаты экспериментов и сравнительный анализ аналогичных методов свидетельствуют об эффективности предлагаемой системы по сравнению с существующими системами сортировки и определения томатов. Результаты, полученные на каждом из трех этапов классификации, т.е. томатный/не томатный, хороший/дефектный и тип дефекта в случае дефекта, передаются в микроконтроллер для включения соответствующего двигателя, чтобы данный фрукт был классифицирован и собран. в соответствующей корзине.

Работа [42] рассматривает практику «зеленой» инженерии, сочетающая искусственный интеллект и подходы к обработке изображений, которая значительно повысила потенциал сельскохозяйственной цепочки поставок некоторых продуктов. Исследователи предложили систему классификации, использующую пару двоичных классификаторов SVM для сортировки спелости и дефектов томатов. Первый классификатор SVM является уникальным независимо от уровня спелости для разделения видов томатов. Уровень спелости томатов определяется на основе среднего зеленого компонента. После этого дефектные и здоровые помидоры различаются с помощью операции суперпиксельной сегментации. Затем используется второй классификатор SVM для обнаружения дефектов черного пятна и язвы отдельно у спелых и незрелых плодов. Хотя одни и те же векторы признаков используются для обнаружения дефектов независимо от спелости, эталонное изображение, используемое для создания остаточных изображений для выделения признаков, отличается для спелых и незрелых томатов. Обширные эксперименты с разработанными тестовыми примерами на наборах данных, созданных в рамках контролируемых экспериментов, демонстрируют точность классификации предлагаемой системы. Исследователи достигли точности классификации 95% для спелых фруктов и 92,5% для незрелых плодов при обнаружении черных пятен и язв.

Использование модели Fuzzy Mask R-CNN для автоматического определения спелости томатов описана в работе [16]. Исследователи предложили нечеткую модель Mask R-CNN для автоматического определения уровней спелости томатов черри. Для автоматического аннотирования изображений использовалась нечеткая модель c-средних для сохранения пространственной информации различных элементов переднего и заднего плана изображения. Применялся метод преобразования Хафа, чтобы определить положение геометрических краев помидоров. Каждая точка данных в пространстве изображений была аннотирована в файл нотации объектов JavaScript. Аннотированные изображения были обучены с помощью Mask R-CNN для точной идентификации каждого помидора. Чтобы предотвратить опадение томатов перед сбором урожая, для прогнозирования спелости томатов использовалась цветовая модель «оттенок-насыщенность-значение» и правила нечеткого вывода. Тригонометрическая функция с евклидовым расстоянием была рассчитана от начала чашечки и стебля до нижней части томата, чтобы получить положение головки ножки и своевременно разрезать плод. Для обнаружения 100 изображений помидоров

маска R-CNN достигла точности 98,00%. Классификация томатов по спелости достигла общей взвешенной точности и степени отзыва 0,9614 и 0,9591 соответственно.

В работе [19] описывается система классификации спелости томатов в реальном времени, основанная на модели глубокого обучения одноэтапного обнаружения. Прикладная модель глубокого обучения следует архитектуре сверточной нейронной сети из одного потока, которая демонстрирует самую быструю и наиболее точную производительность в области обнаружения объектов. Для проверки метода использовался набор данных из 4 182 изображений с высоким разрешением, который применялся для обучения и тестирования модели глубокого обучения. Классификация спелости проводилась при максимальной скорости вычислений 97 кадров в секунду и средней точности 91,3%.

В работе [20] предлагается метод определения спелости томатов путем использования нескольких потоков сверточной нейронной сети (ConvNet) и их методологии стохастического объединения решений (SDF). Авторы назвали общий конвейер SDF-ConvNets, который может правильно определять спелость томатов, выполняя последовательные этапы: (1) определение начальной спелости томатов для многовидовых изображений на основе модели глубокого обучения и (2) стохастическое объединение этих начальных результатов для получения окончательного результата классификации. Общий конвейер определения спелости состоял из двух основных этапов: начального этапа определения спелости томатов на основе потоков ConvNet и этапа объединения стохастических решений для получения более точного результата классификации спелости. Даже если исходная классификация спелости не удалась для изображения томатов на концах стебля или цветков, предложенная фаза слияния решений компенсирует ошибочно классифицированную стадию на правильную стадию. Пятикратная перекрестная проверка использовалась для надежной оценки эффективности предложенного метода, где средняя точность определения пяти стадий спелости образцов томатов достигла 96%. В данной работе не проводилось последующее исследование для разработки интегрированной системы, которая может определять подходящее время сбора урожая и контролировать состояние роста урожая путем распознавания и оценки стадии созревания в режиме реального времени посредством наблюдения за томатами перед сбором урожая.

В [23] система машинного зрения для робота-уборщика яблок была разработана на основе алгоритмов YOLOv3 и YOLOv5 со специальной предварительной и пост-обработкой, где YOLOv3, позволяет достичь доли необнаруженных яблок (FNR) - 9,2% для всего набора изображений, 6,7% для общих изображений и 16,3% для изображений крупным планом. Доля ошибочно принимаемых за яблоки предметов (FPR) составила 7,8%. YOLOv5 определяло яблоки без каких-либо дополнительных методов, показывая FNR на уровне 2,8% и FPR на уровне 3,5%. Предлагаемая методика позволила

адаптировать YOLOv3 для робота-сборщика яблок, разделив среднее время обнаружения яблок 19 мс с FPR на 7,8% и FNR на 9,2%. Авторы утверждают, что доля ошибок были меньше, чем во всех известных аналогичных системах. Также YOLOv5 обнаруживало яблоки без каких-либо дополнительных методов: 2,8% яблок не были обнаружены, и 3,5% объектов, обнаруженных как яблоки, относились к фону.

Робот-манипулятор привлекает большое внимания исследователей, поскольку они широко используются в научных исследованиях и промышленных, инженерных сферах деятельности, таких как исследование космоса, подводная съемка, промышленная и военная промышленность, сварка, покраска и сборка, а также медицинские приложения. Последние исследования также подтверждают и направления, связанные с аргопромышленным сектором. Одним таким примером служит и уборка томатов, так как делать это вручную трудоемкая работа, которая требует время и посильный физический труд, что делает ее нецелесообразной, расходной в крупных хозяйствах и даже может нарушать некоторые санитарные нормы в плане физического носителя вирусов и бактерий, передаваемые растениям от человека и обратно – пыльцы, вызывающие у человека аллергические реакции. При сборке следует учитывать и физические качества, к примеру помидор очень мягкий и склонен к появлению синяков, что затрудняет сбор урожая и процесс захвата для манипуляторов. Еще одной из основных проблем в сельскохозяйственном секторе является рост стоимости рабочей силы. Таким образом, роботизация уборки томатов может стать одним из наиболее эффективных решений для устранения упомянутых выше проблем. Однако при рассмотрении вопроса о роботизации еще предстоит решить многие технологические проблемы. Это исследование предлагает один подход к роботизации уборки томатов.

Перспективы в этой отрасли были связаны с робототехникой, и поэтому были разработаны и исследованы различные типы роботов-манипуляторов, например, последовательные манипуляторы состоят из дублирующих манипуляторов [43] и мобильных манипуляторов [44], параллельных манипуляторов [45] и манипуляторов с тросовым приводом [46].

Резервный манипулятор часто проектируется как серия звеньев, соединенных приводными в действие соединениями, которые простираются от фиксированного основания до рабочего органа, в то время как мобильный манипулятор часто проектируется как роботизированное устройство, состоящее из мобильной платформы и резервного манипулятора. закреплен на платформе [47]. В отличие от этих последовательных манипуляторов, параллельный манипулятор - это механическая система, которая обычно использует несколько последовательных цепей для поддержки одной платформы или рабочего органа. Кроме того, тросовый манипулятор - это специальный параллельный манипулятор, в котором движущаяся платформа приводится в движение тросами, а не жесткими звеньями [48].

Использование этих манипуляторов для экономии труда и повышения точности становится обычной практикой в различных промышленных областях. Как следствие, многие подходы были предложены, исследованы и использованы для управления роботами-манипуляторами [49].

2.2 Оценка качества алгоритмов машинного обучения для классификации томатов

После предварительной обработки изображения, идет процесс классификации для выявления к какому классу объект относится. В данной работе были использованы алгоритмы Random Forest, Support Vector Machine и Extreme Gradient Boosting.

Как показано на рисунке 2.1, набор данных содержит 3 класса томатов с 15 примерами: Yellow, Red и Green. Каждый рассматриваемый класс был сохранен в наборе данных в виде массива.

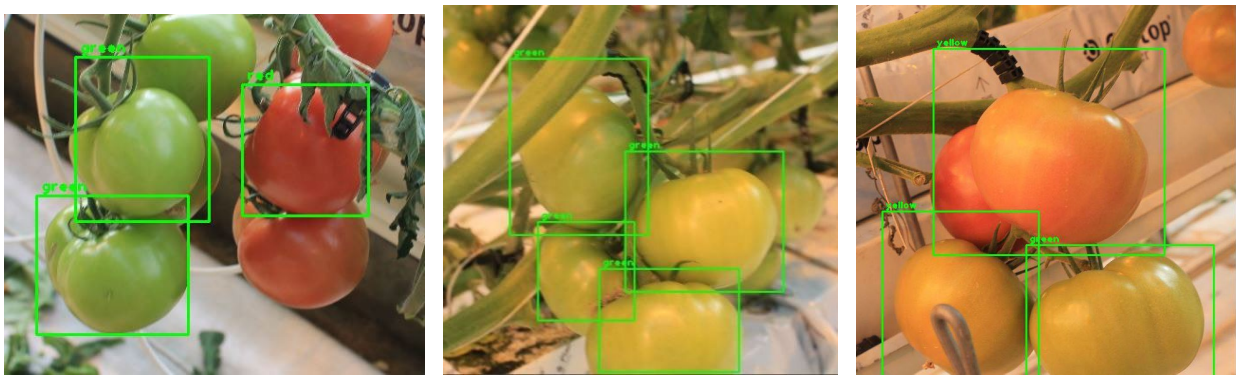


Рисунок 2.1 — Формирование датасета

Поскольку классификатор должен использовать обучающий набор, сбалансированный по классам, чтобы быть эффективным [50], был реализован процесс случайной выборки, чтобы выбрать одинаковое количество объектов для каждого класса во всем изображении.

Для оценки качества работы алгоритмов машинного обучения было разработано множество метрик. Вычисление метрик осуществляется с помощью confusion matrix, которая содержит в себе 4 комбинаций: True Negative – объекты, которые являются верно-отрицательными; True Positive – объекты, которые являются верно-положительными; False Positive – объекты, которые классифицированы как положительные, но в действительности являются отрицательными; False Negative – количество решений, которые классификатор предсказал как отрицательные объекты, но в действительности являющиеся положительными. Столбцы confusion matrix резервируются за фактическими решениями, а строки – за решениями, предсказанных классификатором. Функции FPR, FNR, recall, precision, accuracy, индекс Жаккарда используют два или три комбинаций матрицы неточностей и не дают объективной оценки результатов классификации [51]. А метрики F1, каппа Коэна, коэффициент корреляции Мэтьюса, используя

все элементы матрицы ошибок, оценивают результаты классификатора при несбалансированных данных. Ниже представлены формулы нахождения этих метрик:

1. Долю ошибок, сделанных классификатором при отнесении одного или другого объекта к выбранному классу демонстрирует метрика FPR, ложно-положительный показатель, который вычисляется по формуле 2.1. Значение данной метрики зависит от количества ложно-положительных и истинно-отрицательных решений.

$$FPR = \frac{FP}{FP + TN} \tag{2.1}$$

2. Ложно-отрицательный показатель FNR демонстрирует ошибку второго рода, когда модель машинного обучения предсказывает отрицательное решение, но на самом деле он является объектом выбранного класса. Для вычисления данной метрики нужно использовать формулу 2.2.

$$FNR = \frac{FN}{FN + TP} \tag{2.2}$$

3. Метрика recall определяется с помощью верно-положительных результатов, но вместо ложно-положительных решений (FP) учитывает количество объектов, классифицированных как отрицательные, но фактически являющиеся положительными (FN). Данная метрика оценки демонстрирует способность обнаруживать определенный класс, и показывает сколько примеров из положительных было потеряно в результате классификации. Чем выше значение метрики recall, тем меньше и значение потери правильных предсказаний.

$$Recall = \frac{TP}{TP + FN} \tag{2.3}$$

4. Precision отвечает за способность отличать заданный класс от всех остальных классов, но в отличие от recall, зависит только от положительных результатов, то есть precision – соотношение между верно-положительными результатами (TP) и всеми положительно классифицированными объектами (TP и FP).

$$Precision = \frac{TP}{TP + FP} \tag{2.4}$$

5. Accuracy рассчитывает долю правильных классификаций, и определяется как соотношение всех истинных результатов и суммы всех комбинаций матрицы ошибок.

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{2.5}$$

6. F1-measure является метрикой, которая сводит к одному числу две основных метрики оценки: precision и recall. Она нужна для сбалансирования, когда максимальное значение precision и recall не достижимы одновременно.

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} \tag{2.6}$$

7. Индекс Жаккарда используется для обнаружения граней с изображения, так как он способен количественно оценить сходство между идентификацией граней компьютера с идентификацией тренировочных данных. Поэтому, этот индекс является важным при семантической сегментации изображения.

$$Jaccard\ index = \frac{TP}{TP + FN + FP} \tag{2.7}$$

8. Каппа Коэна, как и другие показатели оценки, рассчитывается на основе матрицы ошибок. В отличие от расчета общей точности, каппа Коэна учитывает дисбаланс в распределении классов. Поэтому, как показано в формуле, он исключает возможность совпадения классификатора и случайного предположения и измеряет количество предсказаний, который он делает.

$$Cohen's\ kappa = \frac{(TP+FP)(TP+FN)+(FN+TN)(FP+TN)}{(TP+FP+FN+TN)^2} \tag{2.8}$$

9. Коэффициент корреляции Мэтьюса используется в тех случаях, где размеры классов сильно различаются, и набор данных отрицательных и положительных являются несбалансированными. Показатель этой метрики зависит от произведения истинных и ложных решений классификатора по отдельности, так же от вычитания этих двух чисел.

$$MCC = \frac{(TP*TN-FP*FN)}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \tag{2.9}$$

10. Специфичность или истинно-отрицательный показатель отвечает за вероятность того, насколько классификатор правильно не относит объекты к выбранному классу, игнорируя ошибку второго рода, то есть количество ложно-отрицательных решений.

$$Spc = \frac{TN}{TN+FP} \tag{2.10}$$

11. Индекс Юдена определяется разницей между долей истинно-положительных результатов и долей ложноположительных решений.

$$Youden's\ index = Recall + Specificity - 1 \tag{2.11}$$

2.3 Применение и оценка алгоритма машинного обучения Random Forest (RF) для классификации томатов

Первым методом, выбранный для классификации томатов является алгоритм Random Forest [52]. На рисунке 2.2 показан процесс работы алгоритма Random Forest.

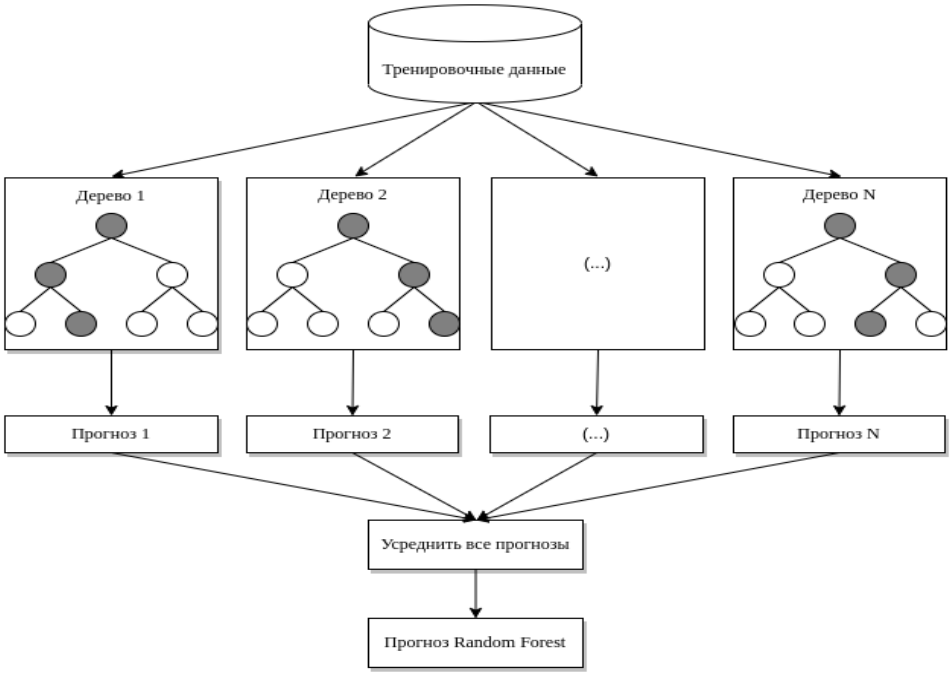


Рисунок 2.2 — Процесс работы классификатора Random Forest

Представляя совокупность деревьев решений, данный алгоритм объединяет их в целях получения более точного результата. Обучающая выборка делится на подвыборки заданного размера, после по этим выборкам

строятся деревья решений. Для разветвления в дереве, выбирается максимальное значение из случайных признаков, где каждое новое разветвление дерева имеет случайных признаков. Выбирается наилучший признак и строение дерева продолжается до окончания выборки. Но новые реализации данного алгоритма имеют параметры, ограничивающие высоту деревьев и количество объектов в подвыборке.

Random Forest для задач классификации часто используете индекс $Gini$ или формулу 2.12, используемую для определения того, как разветвляются узлы в дереве решений.

$$Gini = 1 - \sum_{i=1}^{C}(p_i)^2 \tag{2.12}$$

Эта формула использует класс и вероятность для определения $Gini$ каждой ветви на узле, определяя, какая из ветвей произойдет с большей вероятностью. Здесь p_i представляет относительную частоту класса, который вы наблюдаете в наборе данных, а C представляет количество классов.

Также можно использовать энтропию, чтобы определить, как узлы разветвляются в дереве решений (2.13).

$$Entropy = \sum_{i=1}^{C} -p_i * (p_i) \tag{2.13}$$

Правило машинного обучения заключается в том, что чем больше функций, тем больше модель пострадает от переобучения. А классификатор Random Forest способен измерить относительную важность каждой функции для стабильного прогноза. Таким образом, можно исключить функции, не являющимися полезными для классификации.

Для ускорения работы и улучшения качества классификатора используются гиперпараметры, такие как количество деревьев, необходимые для посторения, перед тем как принять максимальное голосование прогнозов и максимальное количество функций, учитывающиеся для разделения узла. Следует отметить, что использование деревьев большого количества приводит к замедлению работы классификатора. В процессе обучения было использовано 400 деревьев решений, так как это значение оказалось приемлемым при использовании RF-классификатора.

На рисунке 2.3 показана общая матрица ошибок алгоритма RF для всех классов томатов. Количество истинно-положительных решений (TP), предсказанных классификатором RF – 22, а число истинно-отрицательных комбинаций(TN), которые не принадлежат к выбранному классу и были классифицированы как отрицательные правильно – 46. Ложно-отрицательная комбинация отвечает за количество ошибочно предсказанных решений, то есть классификатор предсказал их как отрицательные объекты, но в

действительности они являются положительными, принадлежащими к выбранному классу объектами, количество FN - 2. С помощью этих комбинаций матрицы ошибок вычисляются метрики, которые определяют качество работы классификатора. Численные показатели оценок алгоритма для каждого класса представлены в таблице 2.1.

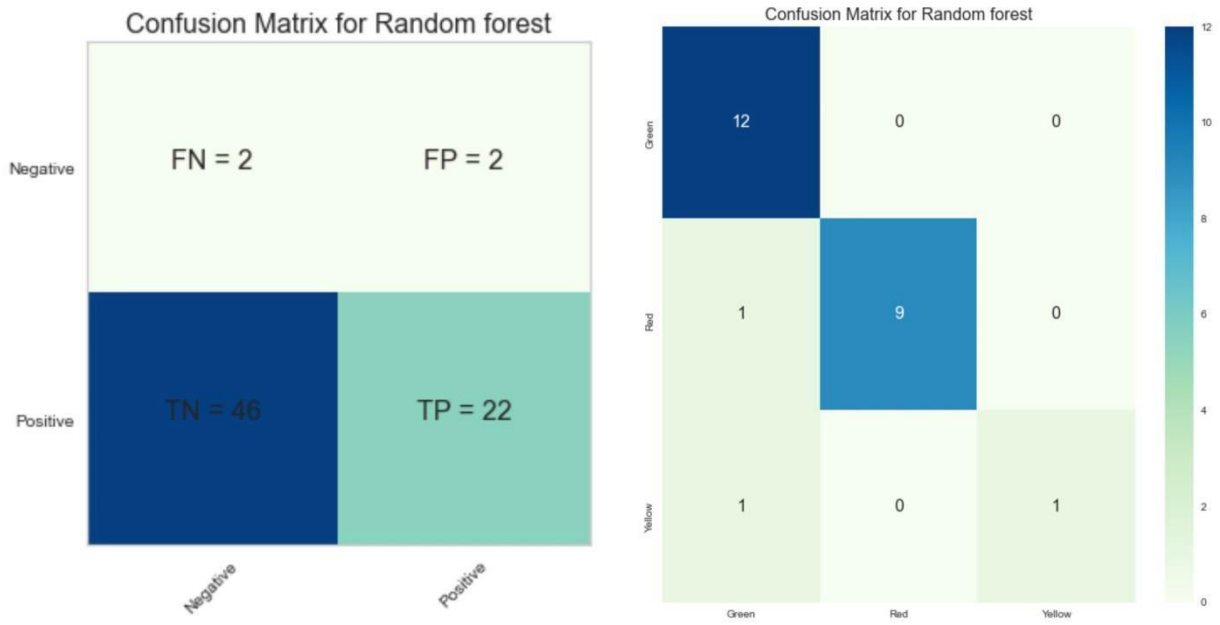


Рисунок 2.3 — Матрица ошибок Random Forest

Таблица 2.1 — Оценки для каждого класса алгоритма Random Forest

Класс/Метрика	*Precision*	*Recall*	*F1*	*Accuracy*
Yellow	*1.00*	*0.50*	*0.67*	*0.96*
Red	*1.00*	*0.90*	*0.95*	*0.96*
Green	*0.86*	*1.00*	*0.92*	*0.92*

Согласно формуле (2.4), precision класса томатов «Green» равна 0.86, классов «Red» и «Yellow» равны 1, что показывают точную классификацию объектов этих классов. Среднее значение precision составило 0.95, так как классификатор ошибочно предсказал некоторые объекты класса «Green» как объекты других классов, и это повлияло на общее качество. Следующая метрика, recall вычисляется по формуле (2.5). Для объектов класса «Yellow» значение данной метрики составляет 0.5, для сорных растений вида «Red» - 0.90, для класса «Green» оценка recall достигла максимума, то есть 1.0. Общая оценка метрики recall равна 0.79. Значение метрики F1, которая является гармоническим средним этих двух метрик, будет равно 0.87. По результатам проведенной оценки, можно сделать вывод о том, что алгоритм Random Forest выполнил классификацию хорошо, так как показатели метрик оценки имеют высокие результаты.

2.4 Применение и оценка алгоритма машинного обучения Support Vector Machine (SVM) для классификации томатов

Следующий метод исследования – классификатор **support vector machine (SVM) [53]**. Процесс работы данного алгоритма можно разбить на две части, которые отвечают за обучение классификатора и за распознавание подаваемых на вход символов.

На первом этапе разрабатывается программная реализация математического аппарата SVM для создания модели классификатора. Модель SVM представляет разные классы на гиперплоскости в многомерном пространстве. Эта гиперплоскость сгенерируется итеративным способом, чтобы минимизировать ошибку. Целью работы данного классификатора является разделить наборы данных на классы для нахождения максимальную предельную гиперплоскость. Один из важных понятий в SVM – опорные векторы или векторы поддержки, совокупность точек данных, расположенные ближе всего к гиперплоскости, и с помощью этих точек определяется разделительная линия. Гиперплоскостью называется плоскость принятий решения, то есть пространство, разделенное между набором объектов, имеющих разные классы.

Во втором этапе, реализуется процесс распознавания и классификации. Выбирается гиперплоскость, которая правильно разделяет классы.

SVM имеет несколько ядер, среди которых наиболее распространенными являются: линейное, полиномиальное и радиальное. В данной работе была использована линейное ядро. Линейное ядро которая представлена на рисунке 2.4 можно описать следующей формулой (2.14):

$$K(w, x) = \langle w, x \rangle \tag{2.14}$$

где w — вектор весов, x — вектор признаков.

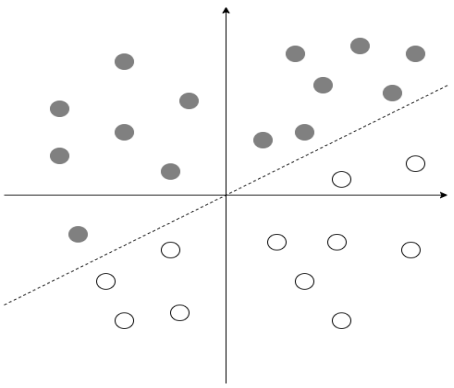


Рисунок 2.4 — Линейное ядро

Главными преимуществами SVM являются способность показать высокую точность в пространстве большого размера. SVM классификаторы

больше всего используют подмножество точек, что способствует использованию меньшей памяти.

Стоить отметить, что SVM требует больше времени для обучения, связи с этим, на практике он не подходит для больших данных. Следующим недостатком SVM является то, что он плохо работают с перекрывающимися классами. Матрица неточностей классификационной модели Support Vector Machine показана на рисунке 2.5. Количество корректно предсказанных объектов – 23, а количество истинно-отрицательных комбинаций составило 47. А количество ошибок, которые классификатор сделал при распознавании сорных растений равно 1. По сравнению с алгоритмом RF,

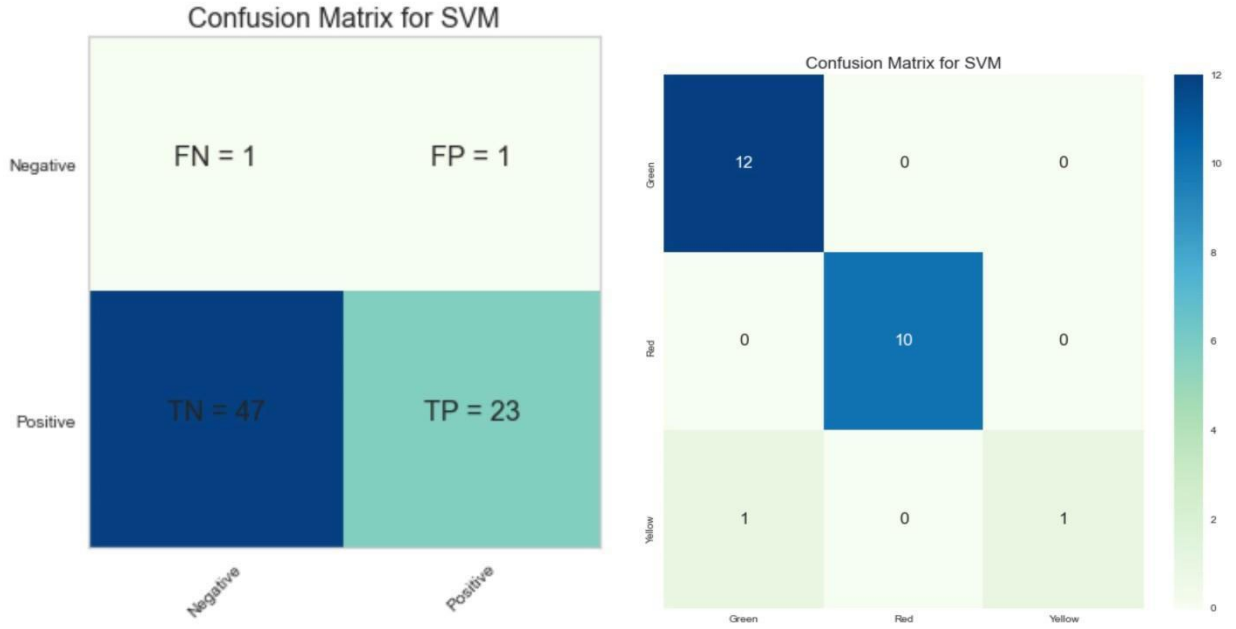


Рисунок 2.5 — Матрица ошибок Support Vector Machine

Значение метрики Precision для объектов «Red» и «Yellow» равно 1, для объектов класса «Green» - 0.92, а общая оценка precision составила 0.97. А показатель recall классов «Red» и «Green» является высоким, чем значение этой метрики класса «Yellow», так как при классификации объектов второго класса было потеряно больше долей правильных предсказаний. Это связано с тем что, множество объектов вида «Yellow» были ошибочно классифицировано как объекты остальных классов и снизило среднее значение данной метрики. Accuracy классификатора Support Vector Machine по двум классам, кроме «Red» составила 0.96, из-за низкой оценки recall класса «Yellow» и precision класса «Green». Численные показатели представлены в таблице 2.2.

Таблица 2.2 — Оценки для каждого класса алгоритма Support Vector Machine

Класс/Метрика	*Precision*	*Recall*	*F1*	*Accuracy*
Yellow	*1.00*	*0.50*	*0.67*	*0.96*
Red	*1.00*	*1.00*	*1.00*	*1.00*
Green	*0.92*	*1.00*	*0.96*	*0.96*

2.4 Применение и оценка алгоритма машинного обучения Extreme Gradient Boosting (XGBoost) для классификации томатов

Третьим алгоритмом, использованный в данном исследовании является классификатор Extreme Gradient Boosting (XGBoost) [54]. Данный алгоритм работает на основе градиентного бустинга деревьев решений. В начале строится ансамбль из слабо обученных моделей, далее обучение производится последовательно. После каждой итерации вычисляются ошибка предсказаний. Добавляя прогнозы нового дерева к прогнозам обученного ансамбля, уменьшается средняя ошибка модели. Новые деревья будут добавляться в ансамблевую модель, пока ошибка будет уменьшаться.

Алгоритм XGBoost разработан для классификации структурированных и табличных данных. С помощью архитектуры градиентного спуска, алгоритм усиливает работу слабых классификаторов. Основными параметрами алгоритма являются число деревьев, размер шага для предотвращения переобучения, изменение значения функции потерь для разделения листа на поддеревья, максимальная глубина дерева и коэффициент регуляризации. Для поддержки распараллеливания процесса построения дерева используется блочная структура, и имеется возможность продолжить обучение для дообучения на новых данных. Параллелизация алгоритма осуществляется с помощью циклов: внутренний цикл вычисляет признаки, внешний цикл перебирает листья деревьев. При этом внешний цикл начнет свою работу, только тогда если внутренний цикл закончил работу. Это мешает параллелизации алгоритма. Чтобы улучшить время работы порядок циклов меняется: инициализация выполняется при считывании данных, после применятся сортировка, которая использует параллельные потоки.

По матрице ошибок алгоритма XGBoost (см. рисунок 2.6), можно увидеть, что количество истинно-положительных решений классификатора равно 20. Число TN-комбинаций составляет 44, а количество ошибок первого и второго рода, сделанных моделью машинного обучения – 4. Так как правильно предсказанных объектов меньше чем, у SVM и RF, Accuracy имеет низкий показатель по сравнению с другими классификаторами.

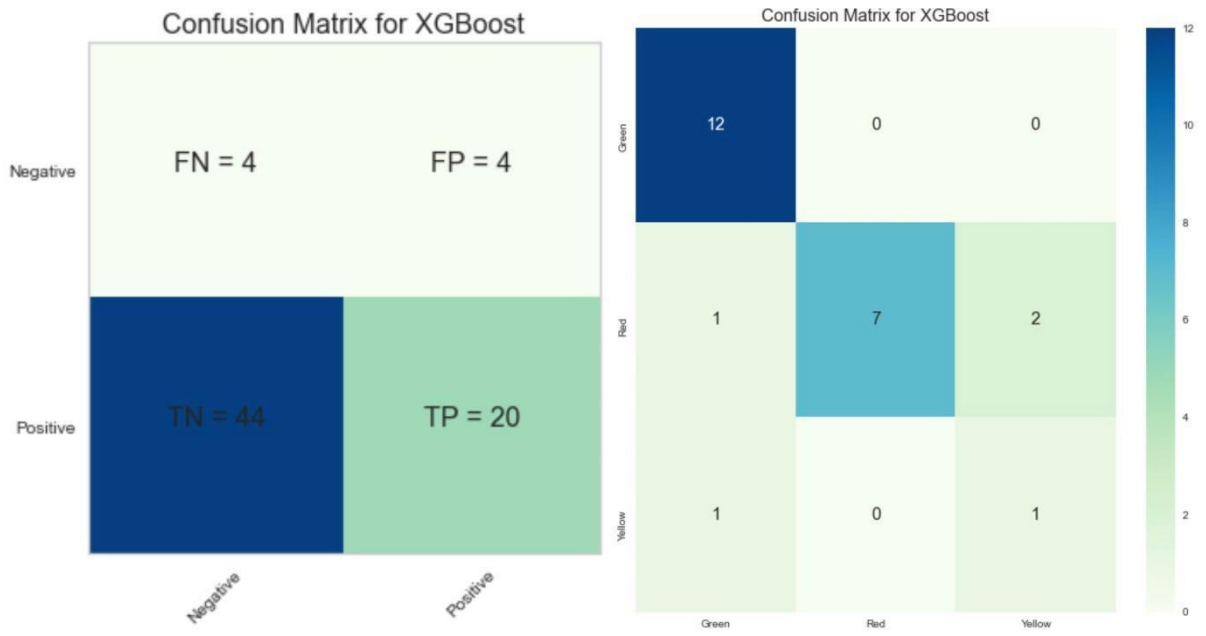


Рисунок 2.6 — Матрица ошибок XGBoost

Больше всего ошибочных предсказаний сделаны при классификации томатов вида «Yellow», поэтому Precision, Recall и Accuracy этого класса имеют значение ниже, чем остальные классы. Численные показатели представлены в таблице 2.3. С использованием метрики F1, уравновешивается баланс между precision и recall, таким образом, для классификатора XGBoost, их значение составило 0.40, 0.82 и 0.92 соответственно для «Yellow», «Red» и «Green».

Таблица 2.3 — Оценки для каждого класса алгоритма XGBoost

Класс/Метрика	*Precision*	*Recall*	*F1*	*Accuracy*
Yellow	*0.33*	*0.50*	*0.40*	*0.88*
Red	*1.00*	*0.70*	*0.82*	*0.88*
Green	*0.86*	*1.00*	*0.92*	*0.92*

Выводы по второму разделу

В представленном разделе описывается процесс различения томатов, а также проведена оценка алгоритмов, которые применялись для классификации. Сформирована собственная база с изображениями красных, желтых и зеленых помидоров. По итогам проведенной оценки, точность обнаружения томатов классификаторами RF, SVM и XGBoost составила 94.4%, 97.2%, и 88.8%.

Количественные результаты, полученные на реальных данных, демонстрируют, что предлагаемый подход может обеспечить хорошие результаты классификации изображений томатов разной размерности.

Помимо этого, качество работы классификатора оценивается скоростью выполнения и производительностью алгоритма.

По времени обучения RF оказался быстрее, чем SVM и XGBoost. Для проверки точности была проведена кросс-валидация, где данные были поделены на 5 блоков. Что же касается скорости предиктирования в задаче реального времени, то RF хоть и выигрывал в скорости обучения, но уступает в скорости выполнения, так как начинаются просадки FPS.

Таким образом точность предсказания и в трех методах является высоким, но SVM и XGBoost показали себя лучше за счёт скорости выполнения работы при обнаружении объекта.

3. РАЗРАБОТКА АЛГОРИТМОВ НЕЙРОННОЙ СЕТИ ДЛЯ ОБНАРУЖЕНИЯ ТОМАТОВ И ОЦЕНКА ИХ ЭФФЕКТИВНОСТИ

3.1 Обзор архитектур нейронных сетей для решения задач распознавания объектов

Нейронные сети, которые обеспечивают высокоскоростную параллельную распределенную обработку и могут быть легко реализованы аппаратно, были признаны мощным инструментом обработки в реальном времени и успешно применяются в различных системах управления. В частности, большое внимание привлекло использование нейронных сетей для управления роботами-манипуляторами, и были предложены и исследованы различные связанные схемы и методы. Обзор прогресса исследований в области управления манипуляторами с помощью нейронных сетей, анализ проблемных основ управления манипуляторами и теоретические идеи по использованию нейронной сети для решения проблемы, подробно описывается и анализирется в отношении практических приложений и указаны некоторые потенциальные направления, которые, возможно, заслуживают исследования в управлении манипуляторами с помощью нейронных сетей в работе [55]. Ссылаясь на разработки и исследования более ранних ученых, где практиковались простые, основанные на линейных, дифференциальных и т.д. алгоритмы [56-58], можно сделать вывод что алгоритмы управления, основанные на нейронных сетях, превосходят по производительности и точности.

В [59] разрабатываются методы обнаружения болезней томатов на основе Faster R-CNN и Mask R-CNN, где Faster R-CNN используется для идентификации типов болезней томатов, а Mask R-CNN используется для обнаружения и сегментации местоположений и форм зараженных участков плода томата, и чтобы выбрать модель, которая наилучшим образом соответствует задаче обнаружения болезней томатов, четыре различных глубоких сверточных нейронных сети комбинируются с двумя моделями обнаружения объектов.

В [60] использовали на слабо контролируемую и облегченную модель общей архитектуры сегментации экземпляров - Mask R-CNN для реализации сегментации экземпляров плодов томатов на основе обучения со слабым учителем, а затем распознавание болезней листьев томатов реализуется с помощью многомасштабной свертки с разделением по глубине.

В работе [61] представлены результаты обнаружения помидоров на снимках, сделанных в теплице, с использованием алгоритма MaskRCNN, который обнаруживает объекты, а также пиксели, соответствующие каждому объекту. Экспериментальные результаты данного исследования по обнаружению помидоров по изображениям, сделанным в теплицах с помощью камеры RealSense лучше, чем показатели, о которых сообщались в более ранних работах, даже если они были получены в лабораторных

условиях или с использованием изображений с более высоким разрешением. Результаты также показывают, что Mask R-CNN может неявно изучать глубину объекта, что необходимо для устранения фона.

Большинство методов обнаружения и подсчета плодов, включая помидоры, использовали преобразования цветового пространства, в которых выделяются интересующие объекты, и извлечение таких признаков, как форма и текстура [62].

Сверточные нейронные сети (CNN) все чаще используются для сегментации и классификации изображений из-за их способности изучать надежные дискриминирующие признаки и справляться с большими вариациями.

Региональные сверточные нейронные сети (R-CNN) объединяют метод селективного поиска [63] для обнаружения предложений регионов [64] и были основой для методов Fast-RCNN [65] и Faster-RCNN [66]. You Only Look Once (YOLO) [67] применяет только одну сеть, разделяя изображение на области и предсказывает ограничительные рамки и вероятности для каждой области. YOLO быстрее, чем Fast-RCNN, который применяет модель к изображению в нескольких местах и масштабах. Эти методы обеспечивают сегментацию интересующих объектов ограничительной рамкой, которая напрямую не передает, какие пиксели принадлежат какому экземпляру объекта, особенно когда есть перекрывающиеся или закрывающие объекты одного класса.

Распознавание зрелых плодов томата (далее просто томат) является одной из основных задач робота и в данной работе для обнаружения томата и определения степени его созревания были реализованы два независимых метода. Первый метод заключается в использовании сверточной нейронной сети Mask R-CNN. А для второго метода на основе работы [68] мы выбрали YOLO.

3.2 Обучение нейронных сетей Mask R-CNN для обнаружения томатов

Mask R-CNN является архитектурой нейронной сети для распознавания объектов и их сегментации. Данный вид нейронной сети имеет свою историю развития связанной с главными задачами компьютерного зрения, так как прародителем данной архитектуры является Faster R-CNN [69].

Для полного понимания необходимо рассмотреть архитектуру сети Fast R-CNN (см. рисунок 3.1). В самом начале на вход подается исходное изображение произвольного размера и пропускается через сверточные слои. В этапе сверточного слоя происходит объединение значения расположенных близкостоящих пикселей и рассчитать общие признаки изображения. Для этого происходит процесс, в котором каждая часть изображения умножается на матрицу (ядро) свёртки поэлементно.

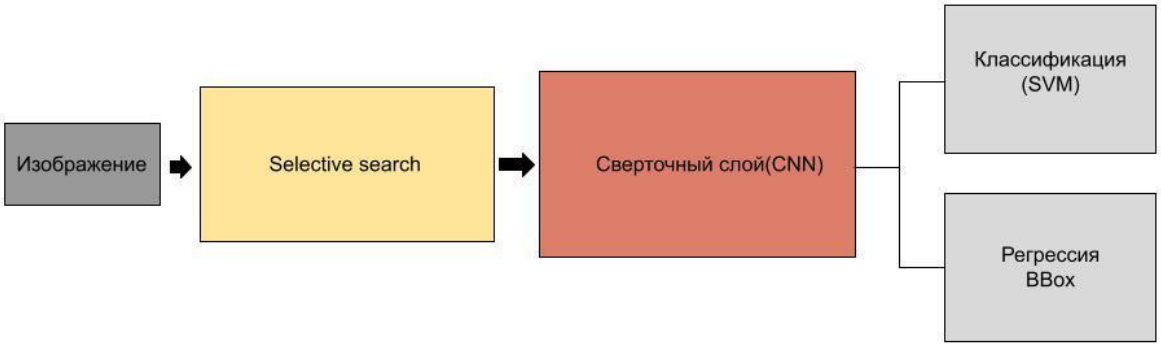


Рисунок 3.1 — Архитектура Fast R-CNN

После сверточного слоя получаем набор характеристик фиксированного размера и количество этих характеристик будет зависеть какой сеть используется. Если это будет AlexNet, то будет 256 карт слоев, а если VGG, то 512. После из этих карт характеристик выбирается только те характеристики, которые идентичны в selective search и передаются в полносвязные слои. После полносвязного слоя они передаются в полносвязный слой softmax для классфикации и в регрессию для уточнения координат углов Bounding box. Данный алгоритм работает значительно быстрее чем R-CNN, так как все выполняется за один проход.

Также создатели заменили SVM (Support Vector Machine) на полносвязанные слои.

В улучшенной версии под названием Faster R-CNN была изменена генерация гипотез. Если в Fast R-CNN гипотезы генерировались с помощью selective search, то в Faster R-CNN данную функцию выполняет Region proposal network [66]. На вход поступает набор признаков из сверточной сети и подается в Region proposal network. Дальше на полученные признаки применяется операция свертки. После этого полученные характеристики для вероятности принадлежности, то есть для определения в данной boundig box содержится объект. После передается информация в полносвязные слои и пропускается только то информация, которая передалась от Region Proposal Network. То есть, если есть вероятность того, что рассматриваемый объект находится в конкретной области выше порога, тогда данные характеристики выбираются из сверточной сети и пропускаются через полносвязные слои. В конце с помощью полученной информации происходит классификация и определение ограничивающей рамки.

На основе Mask R-CNN лежит архитектура Fast R-CNN, которая улучшает его с помощью построения ветви для прогнозирования масок сегментации каждой области интереса [69]. Mask R-CNN по отдельности определяет пиксели принадлежащие каждому объекту для каждого класса. Маска это прямоугольная матрица, где 0 — пиксель объекту не принадлежит, 1 — пиксель объекту принадлежит.

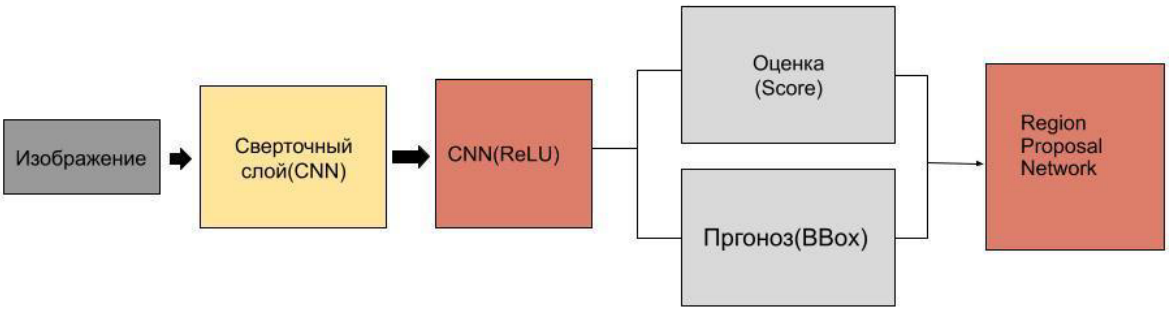


Рисунок 3.2 — Архитектура Faster R-CNN

В Faster R-CNN используется ROI Pooling смысл, которого является объединение соответствующие область на карте признаков в карту признаков фиксированного размера в соответствии с координатами положения предварительно выбранного блока, чтобы выполнить последующие операции классификации и регрессии ограничивающей рамки. Однако, в данном методе были обнаружены недостатки, которые существенно влияли на точность определения. Таким образом, вместо него в Mask R-CNN с целью устранения таких недостатков используется ROI Align.

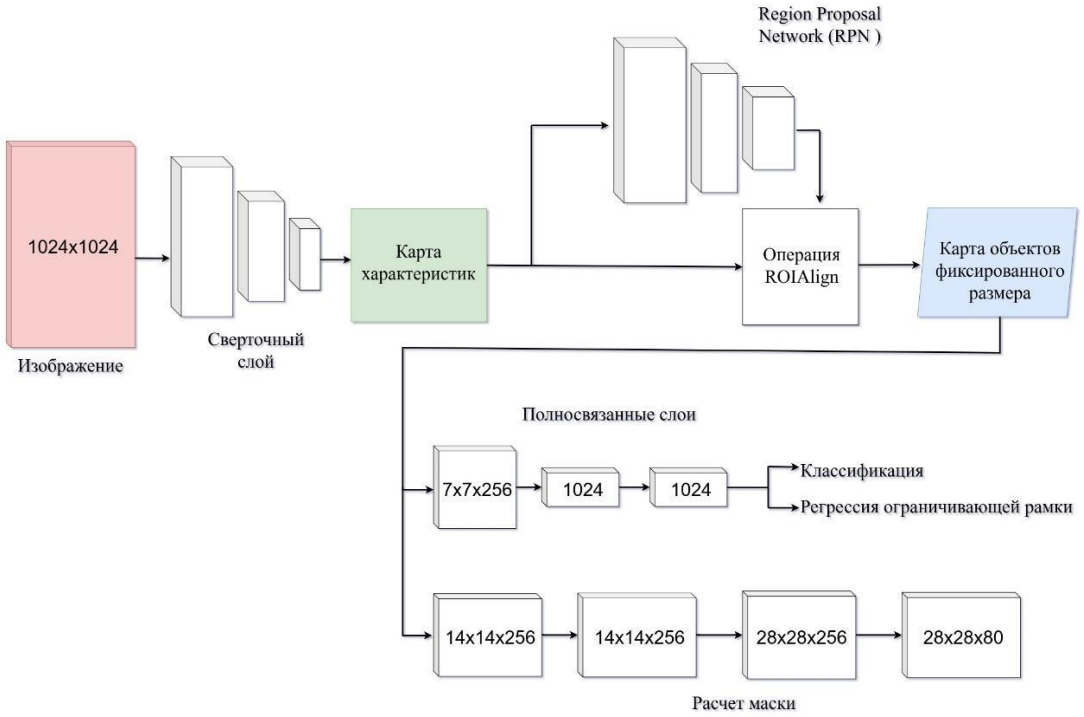


Рисунок 3.3 — Архитектура Mask R-CNN

Главным преимуществом данной архитектуры от Faster R-CNN является предсказания объекта маски с помощью вычисление матрицы признаков для региона интереса. В процессе свертки карты признаков имеет меньший размер, по сравнению с исходным изображением. Поэтому регион пикселей, нельзя отобразить в пропорциональный регион признаков.

В данной работе было необходимо обучить на архитектуре Mask R-CNN три вида помидора по уровню спелости и сделать сравнительный анализ полученных результатов и оценок с архитектурой глубокой нейронной сети YOLOv5. В качестве набора данных были использованы те же данные, что для YOLOv5. Количество классов также 3: спелый (красный) – 0, неспелый (зеленый)-1, полуспелый (желтый) представлены на рисунке 3.4.

Рисунок 3.4 — Три вида классов помидора из набора данных

Обучающие данные были разделены на тренировочные и валидационные в соотношений 80% на 20% соответственно. Количество итерации при обучениях было 100: десять эпох и по 10 шагов по каждому из них. При обучениях параметр confidence (пороговая вероятность) была поставлена на 0.9.

На рисунке 3.5 показан результат работы нейронной сети. Обнаружение и распознавание объектов в данной картинке хоть и показывает неплохой результат, можно заметить, что все же есть объекты, которых алгоритм не смог обнаружить или обнаружил неправильно. Кроме того, сегментация объектов также показывает неплохой результат.,

Рисунок 3.5 — Три вида классов помидора из набора данных

Также на тестовых данных визуализация показывает, что алгоритм хорошо сегментирует объекты. Однако, можно заметить, что хоть и обнаружение происходит неплохо, но есть несколько объектов, которые относятся к другому классу (см. рисунок 3.6).

Рисунок 3.6 — Три вида классов помидора из набора данных

Для оценки качества алгоритма были вычислены среднее значение точности детектора объектов(mAP). С помощью вычисления средней точности для нашего предсказания. В нашем случае алгоритм показал значение 0.132, которое является очень слабым. Однако, среднее значение плотности(mAR) по всем классам составил 0.818.

Наша выборка была несбалансированная, так как количество данных для класса под номером 2 были относительно меньше, чем для двух остальных. Поэтому для оценки качества алгоритма была вычислена совместная оценка точности и полноты (F1 score). Данная метрика позволяет получить более сбалансированную характеристику модели, чем те метрики, которые были упомянуты выше. Значение F-меры показал значение 0.23023. Таким образом, можно утверждать, что алгоритм показал плохой результат при обучениях, так как значении mAP = 0.132 и F1 score = 0.23023 являются низкими.

3.3 Модификация и обучение архитектуры YOLOv5 для обнаружения томатов

Модель YOLO – модель быстрого обнаружения компактных объектов, которая очень эффективна со своим размером по сравнению с аналогами и постоянно совершенствуется [67]. Модификации YOLOv4 и YOLOv5 в первую очередь заключается в интеграции достижений в разных областях компьютерного зрения и доказательстве того, что в совокупности они улучшают обнаружение объектов YOLO.

YOLOv5 выполняет обработку данных со скоростью около 140 кадров в секунду. Он занимает меньше объема чем предыдущие версии, и считается более удобным, что позволяет применяться чаще в производстве, и данное преимущество также уместно для нашей работы. При обучении графический процессор позволяет ускорить время обучения. Также данную версию можно обучать с использованием более чем одного графического процессора, и данный подход ускоряет вычисления.

YOLO имеет высокую скорость распознавания объектов в режиме реального времени, что отличает его от остальных алгоритмов. В YOLO одновременно вводится все изображения. При этом они проходят через сеть только один раз. Сетевая архитектура YOLOv5 состоит из трех частей: Backbone, Neck, Head (Output).

На этапе Backbone выполняется извлечение информативных признаков (см. рисунок 3.7). Модуль focus разрезает входное изображение (image) на части. После для исходного изображения создаются и объединяются четыре слоя карт объектов разных измерений, чтобы уменьшить потерю данных.

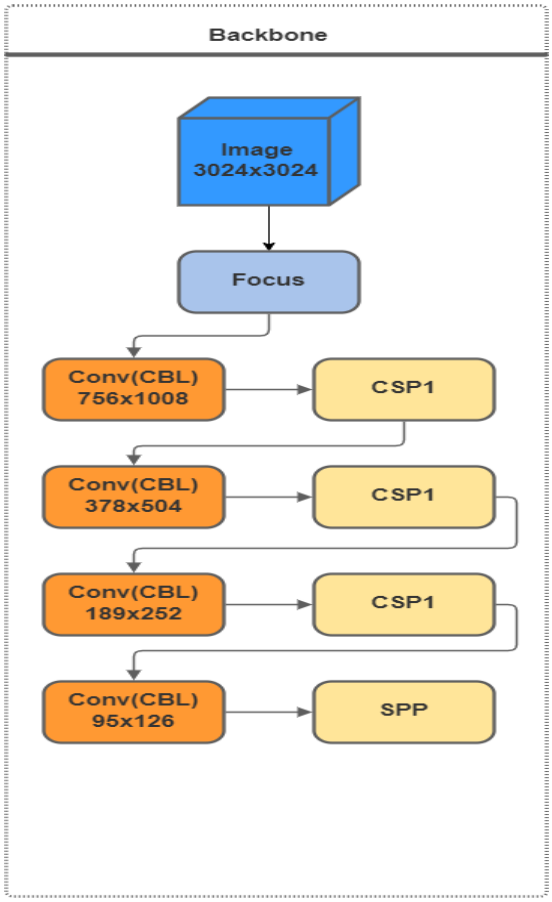


Рисунок 3.7 — Этап Backbone

Полученные результаты с модуля focus последовательно будет использована в модуле CBL: Convolution + Batch Normalization + Leaky—ReLU. Данный модуль состоит из смеси операции свертки, нормализации данных и функции активации Leaky—ReLU.

Нормализация является широко применяемым методом предварительной обработки целью, которой является стандартизация данных. Данный метод позволяет иметь место разных источников данных в одном диапазоне. Отсутствие нормализации данных перед обучением может вызвать проблемы в сети, значительно усложнив обучение и снизив скорость обучения.

В основном для нормализации данных используют формулу с помощью, которой точки данных будут иметь значение между 0 и 1:

$$x_{normalized} = \frac{x-m}{s},$$

где x — точка данных для нормализации, m — среднее значение набора данных и s — является стандартным отклонением набора данных. Таким образом каждая точка данных будет принимать стандартное нормальное распределение. Благодоря такому подходу мы избавимся от смещения признаков и обучение будет происходить лучше.

В нейронных сетях Batch Normalization является методом, который выполняется между слоями с помощью мини-пакетов вместо полного набора данных. Формула нормализации Batch Normalization будет иметь следующий вид:

$$z^N = (\frac{z - m_z}{s_z})$$

где m_z — среднее значением выходных данных нейронов и s_z — стандартное отклонение выходных данных нейронов.

Batch Normalization выполняется перед применением функции активации. Если обычное вычисление нейрона происходит следующим образом:

$$z = g(w, x) + b;$$

$$a = f(z)$$

где $g()$ — линейное преобразование нейрона, w — вес нейрона, b — смещение, а $f()$ — функциz активации, с добавлением Batch Norm будет иметь следующий вид:

$$z = g(w, x);$$

$$z^N = (\frac{z - m_z}{s_z}) \cdot \gamma + \beta; a = f(z^N),$$

где z^N будет иметь выход из Batch Normalization , m_z среднее значениее вывода нейронов, s_z стандартным отклонением вывода нейронов, а γ и β параметры обучения. При вычитании среднего значения m_z любая константа по значениям z будет вычтена сама по себе как смещение (b) .С помощью параметров γ и β можно корректировать среднее значение и стандартное распределений . Таким образом, выходные данные Batch Normalization по слою приводят к распределению со средним значением β и стандартным отклонением γ.

Batch Normalization работает очень похожим образом в сверточных нейронных сетях. Хотя мы могли бы сделать это так же, как и раньше, мы должны следовать свойству свертки.

В сверточных нейронных сетях существуют фильтры, которые проходят по картам объектов ввода и являются одинаковыми на каждой карте объектов. С помощью разделения данных фильтров по картам объектов можно сделать нормализацию выходных данных. Из этого следует, что параметры, используемые для нормализации, рассчитываются вместе с каждой полной картой объектов.

Leaky Rectified Linear Unit (Leaky ReLU) — это тип функции активации, основанный на ReLU. Однако, данная функция для

отрицательных значений по сравнению с ReLU имеет небольшой наклон. Такие функции активации позволяют решать проблему разреженных градиентов.

А при использовании CSP скорость вывода увеличивается, а сложность вычислений уменьшается. Принцип работы сети Cross Stage Partial Network (CSPNet) заключается в разделении и объединении изображений без потери оптимальной скорости и точности. Для обнаружения объектов использовалась сеть Cross Stage Partial Network (CSPNet) с применением фреймворка DarkNet (распечатывает обнаруженные объекты, их достоверность и время, которое ушло на их поиск). Эта сеть разделяет карту объектов базового уровня на две части: первая часть будет проходить через плотный слой и переходной слой; вторая часть затем объединяется с переданной картой признаков на следующий этап.

Из архитектуры видно, что последний слой заменен слоем Spatial Pyramid Pooling (SPP) (объединение пространственных пирамид). Применение SPP обусловлено тем, что для входа в полносвязанный слой, размер входа должен быть фиксированной длины. Для того, чтобы получить вывод фиксированного размера необходимо сделать окно обьединения и шаг пропорциональным входному изображению. Слой SPP может применять несколько операции объединения выходных данных разного размера и объединять их. В данном слое для каждой N операции будут вычислены N чисел с размером сетки nxn.

Таким образом, такая операция будет применяется к каждой карте объектов с значением M, которые были получены в результате предыдущей операции свертки. Они выравниваются и отправляются на следующий слой. Слой SPP позволяет модели CNN выдавать результаты независимо от размера входного изображения.

На этапе Neck (см. рисунок 3.8) берутся результаты последних трех слоев карты объектов. Результаты CSP этапа Backbone передаются в функцию Concat, которая отвечает за операцию объединения тензоров. И блок SPP, заменивший последний слой CSP, переносится в межступенчатую частичную сеть второй ступени. Также на этапе Neck используется модуль Upsample (метод nn.Upsample из фреймворка PyTorch), которая используется для операции повышения частоты дискретизации изображения. В цифровой обработке главной задачей повышения частоты дискретизации является получение одинаковых размеров данных изображении на входе и на выходе.

В данном модуле есть несколько функции, которые позволяют пользователям определять размер выходных данных. С помощью параметра mode выбирается один из 5 типов алгоритмов повышения дискретизации: «ближайший», «линейный», «билинейный», «бикубический» и «трилинейный» (по умолчанию будет выбран алгоритм «ближайший»). При использований алгоритмов «линейный», «билинейный» или «трилинейный» с помощью выходных тензоров можно выровнять входной угол для того, чтобы сохранить значения этих пикселей.

После операции Concat сеть CSP применяется еще раз для сохранения точности и уменьшения размера модели.

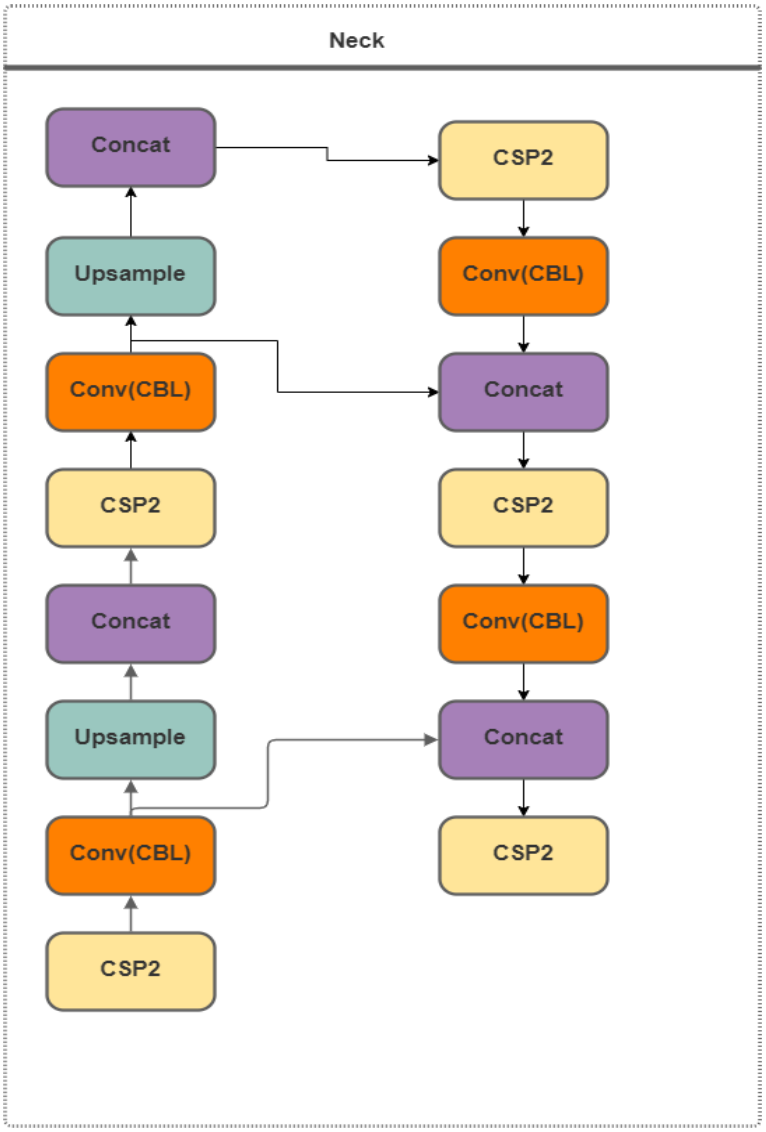


Рисунок 3.8 — Этап Neck

На последнем этапе Head выполняет заключительную часть обнаружения (см. рисунок 3.9). Он применяет блоки привязки к объектам и генерирует конечные выходные векторы, содержащие предсказанную ограничивающую рамку, координаты (центр, высота, ширина), оценку достоверности прогноза и классы вероятности.

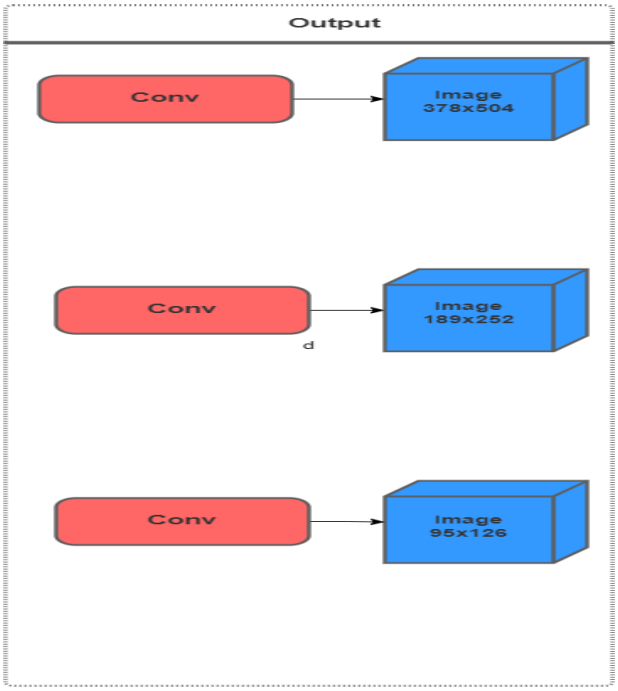


Рисунок 3.9 — Этап Head

В нашей исследовательской работе представлена улучшенная архитектура YOLOv5, которая показана на рисунке 3.10. Ее отличие заключается в изменении архитектуры путем добавления модуля внимания на основе ECA-Net для достижения лучшей производительности нейронной сети. ECA-Net - это механизм внимания, предназначенный для балансировки параметров сложности и производительности, который ранее применялся к архитектурам нейронных сетей, таким как ResNet и ImageNet [62].

В ECA-Net подается данные в качестве входного четырехмерного тензора. Каждый тензор будет состоять из четырех параметров: размер батча, количество каналов и пространственный размер (высота и ширина) картинки каждой карты объектов. Обозначим их как b,c,h,w. При выходе из сети также будет четырехмерный тензор (b,c,h,w).

ECA-Net состоит из 3 операции:

1. Глобальный дескриптор функции;
2. Адаптивное взаимодействие с соседями;
3. Широковещательное масштабирование.

Известно, что для эффективной работы сети необходимо рассмотреть каждый канал. Однако, каждый канал состоит из матрицы размером hxw, что значительно усложняет данный процесс. Глобальный дескриптор функции используется для уменьшения каждой карты функций до одного пикселя, взяв среднее значение всех пикселей в этой карте функций. Таким образом, размер информационного пространства сокращается до единичного пиксельного пространства с CxHxW до Cx1x1. Данная операция позволяет захватывать глобальную информацию в каждом канале.

После тензор размера Cx1x1 или вектор длины C подвергается одномерной шаговой сверточной операции, размер ядра которой равен k, определяемым нелинейным отображением, адаптируется к глобальному пространству канала C в соответствии с функцией ψ (C) и определяется по следующей формуле:

$$k = \psi(C) = \left| \frac{log_2(C)}{\gamma} + \frac{b}{\gamma} \right|_{odd},$$

k обеспечивает размер локального соседнего пространства, которое будет использоваться для захвата межканального взаимодействия при отображении весов внимания для каждого канала. Эта функция ψ (C) по существу аппроксимирует ближайшее нечетное число в качестве размера ядра для одномерной свертки на основе формулы. Данная операция называется адаптивное взаимодействие с соседями.

Полученный взвешенный тензор формы Cx1x1 проходит через сигмовидный слой активации, который устанавливает пороговые значения весов каждого канала в диапазоне от 0 до 1. Затем этот вектор внимания канала передается на вход. размер тензора (CxHxW), а затем поэлементно умножается на него. Данная операция называется широковещательное масштабирование.

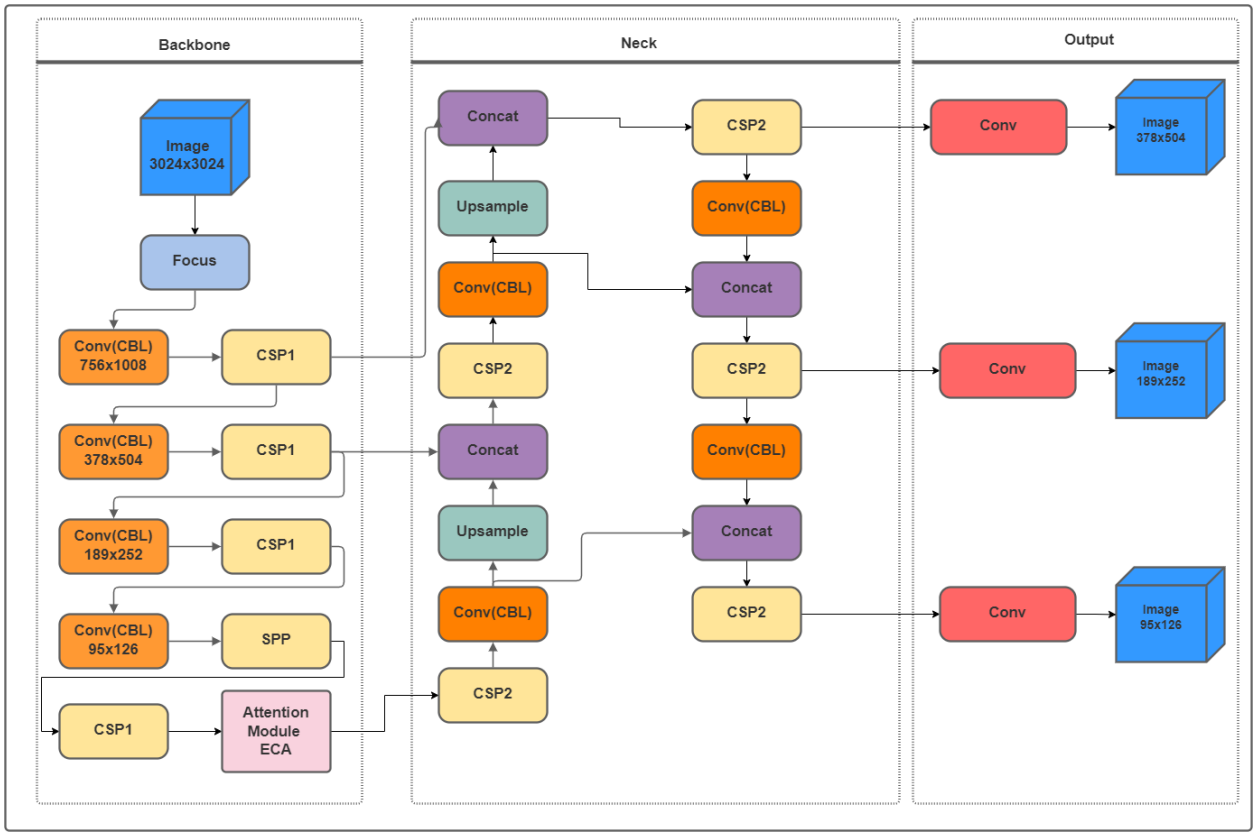


Рисунок 3.10 — Архитектура YOLOv5 с добавлением модуля внимания

Модуль внимания применяется к результатам CSP1 этапа Backbone перед этапом Neck. Важным фактором в изучении модуля внимания является уменьшение размерности. На этапе Backbone результаты CSP1 первого этапа последовательно передаются в модуль дополнительного внимания, таким образом, производительность модели повышается, а также улучшается способность обнаружения. Как показано в представленной архитектуре, на вход было подано изображение томата размером 3024x4032 пикселей. С помощью модуля focus были извлечены более информативные функции. После использования модуля CBL, который включает в себя функции свертки, нормализации и активации, размер уменьшился до 756x1008 пикселей. В результате очередной свертки было получено изображение размером 378x504. Использование третьего слоя уменьшило размер входных данных до 189x252 пикселей. В результате на этапе Neck было передано изображение размером 95x126 пикселей.

После свертки для вывода были получены три карты объектов разного масштаба, которые составляют 378x504, 189x252 и 95x126 пикселей соответственно для каждого слоя.

После получения результата обнаружения объекта оценивается представленная архитектура.

Для обучения нейронной сети YOLO было собрано набор данных из 800 фотографии, которые были сделаны в теплице Almaty Tomato Greenhouse, Казахстан. Следует учитывать, что данные фотографии были сделаны в разное время суток, с разных ракурсов и при разном освещении. При разметке данных томаты были разделены на 3 класса: 0-красный, 1-зеленый, 2- желтый. Само обучение проводилось в 100 эпохах.

Для демонстрирации эффективности улучшенной архитектуры YOLOv5 мы сранили его с YOLOv5.

На рисунке 3.11 показана confusion matrix YOLOv5 и улучшенной архитектуры YOLOv5 с нормализацией для мультиклассовой (3 класса) классификации. По диагонали представлено количество TP-комбинаций для каждого класса: от всех объектов класса 0 с применением YOLOv5 92%, с улучшенной 95%; от всех объектов класса 1 с применением YOLOv5 89%, с улучшенной 93%; от всех объектов класса 2 с применением YOLOv5 82%, с улучшенной 92% были классифицированы правильно. Так же 1% от всех объектов класса 0 ошибочно предсказаны как объекты первого класса в обеих архитектурах, и еще 2% в YOLOv5, а в улучшенной 1% - как объекты второго класса. Остальные 6% с применением YOLOv5 классификатор не отнес ни к какому классу, в то время как для улучшенная архитектура не смогла отнести 4%. Для класса 1 остальные 7%, кроме TP-решений также являются false negative показателями. И для класса 2 2% от всех объектов ошибочно классифицированы как объекты нулевого класса, и еще 4% классификатор предсказал как объекты первого класса. Доля false negative решений от всех объектов класса 2 – 7%. Из результатов видно что, YOLOv5 без улучшения ошибается больше.

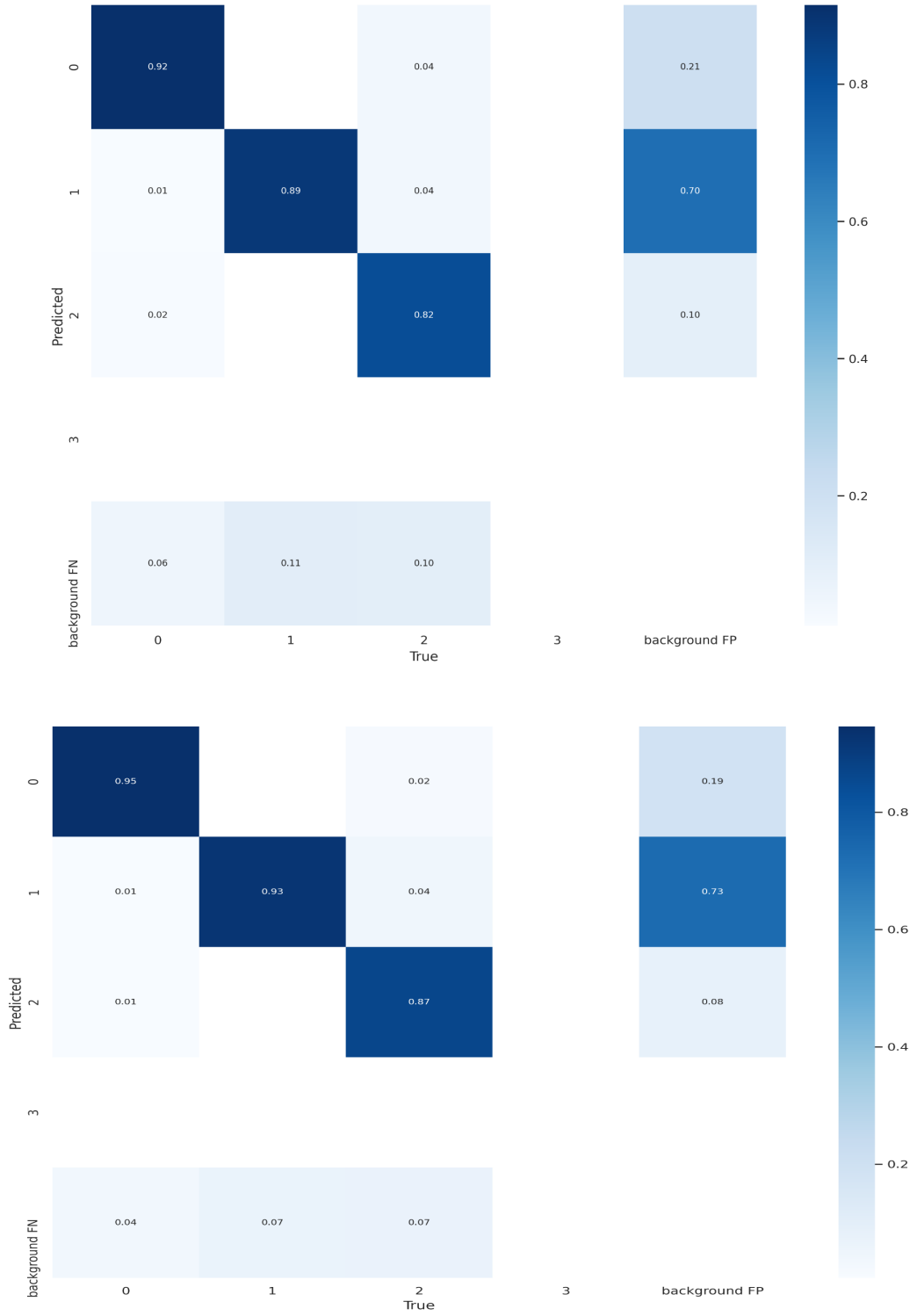


Рисунок 3.11 — Матрица ошибок

На основе этих комбинаций, рассчитываются основные метрики классификационной способности алгоритма, такие как Precision, Recall, Accuracy и f1-measure. По графику, приведенному на рисунке 3.12 можно увидеть, что значение метрики precision увеличивается после каждой итерации, что доказывает высокую точность распознавания.

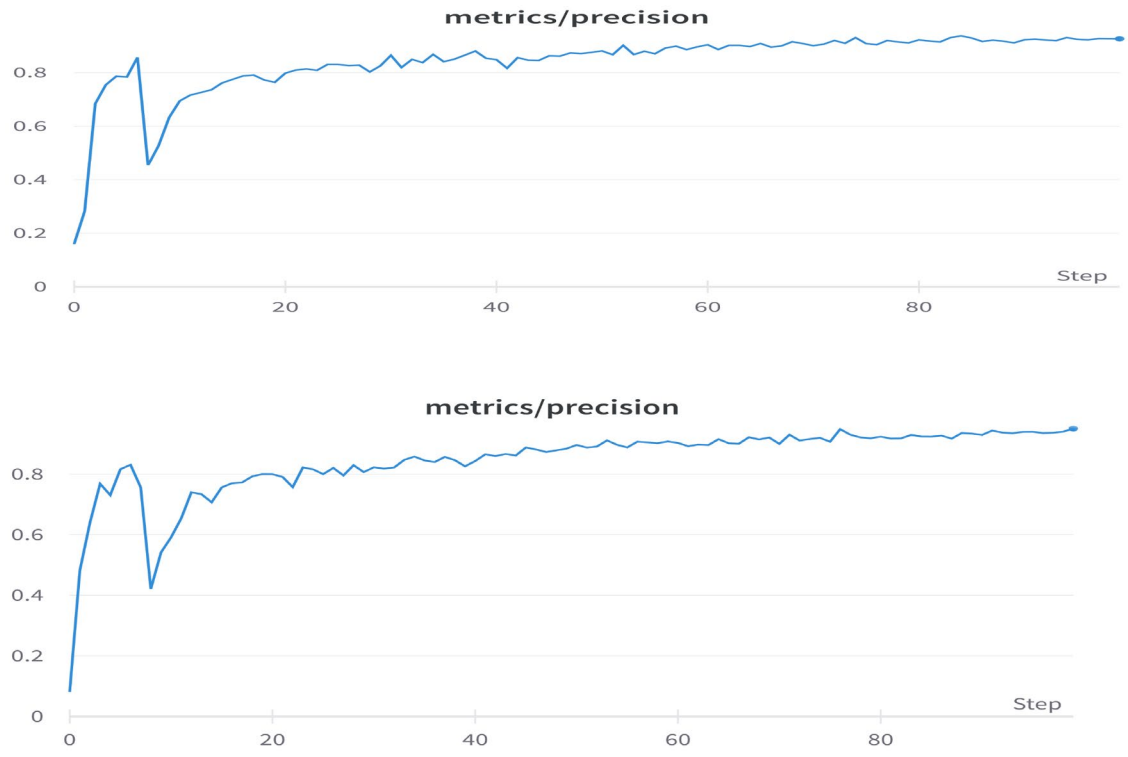
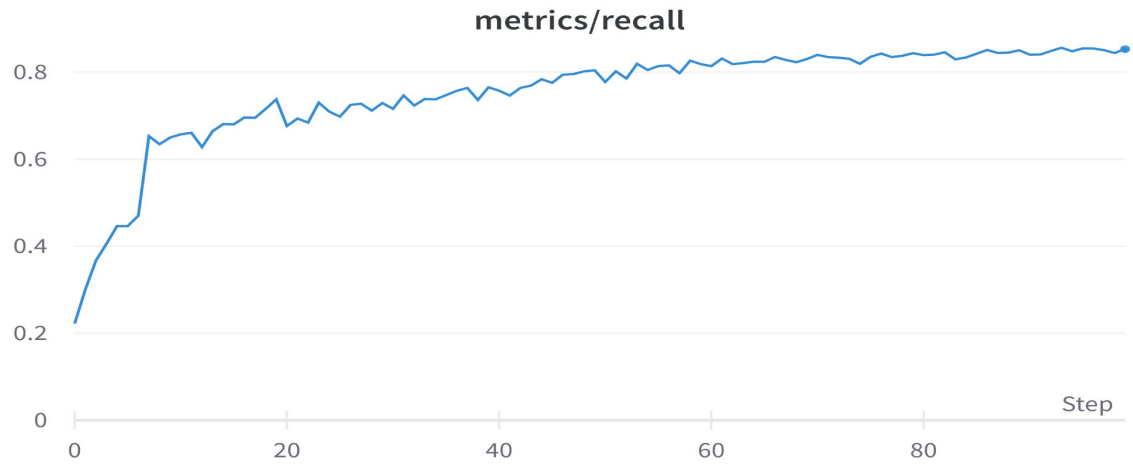


Рисунок 3.12 — Графическая интерпретация метрики precision

По рисунку 3.13 можно увидеть, что значение recall в начальных итерациях было меньше 0.5, что сильно влияет на качество работы алгоритма, но потом с каждой итерацией возрастало почти до единицы.



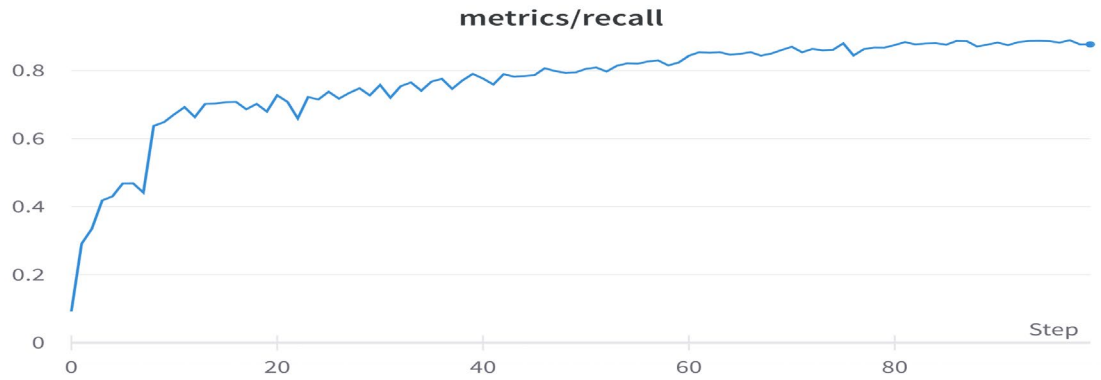
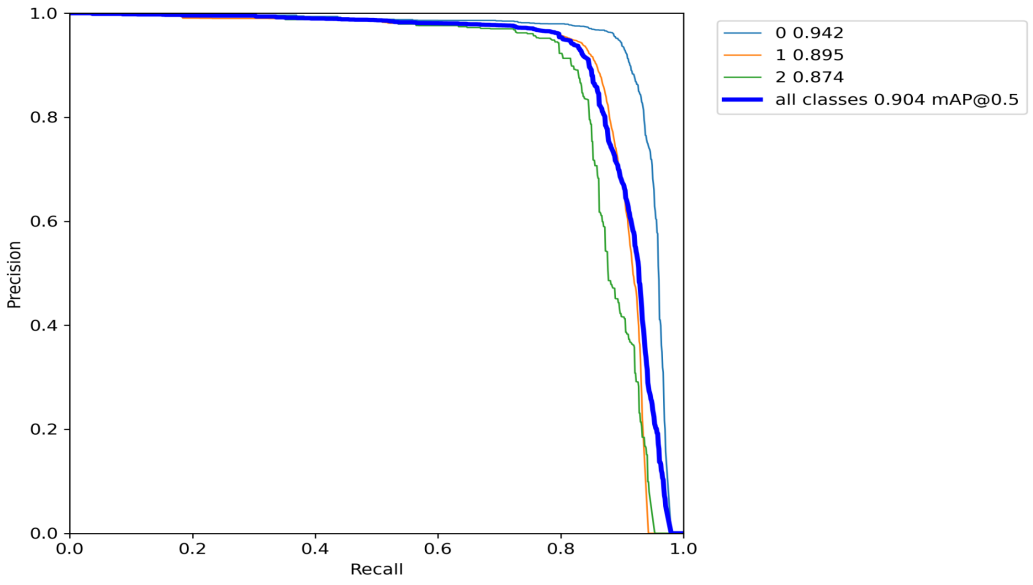


Рисунок 3.13 — Графическая интерпретация метрики recall

С помощью precision и recall демонстрируется precision-recall curve, которая является важной метрикой при работе с несбалансированными данными. Высокие показатели под кривой для обоих метрик показывает, что классификатор делает правильные прогнозы. В качестве порогового значения было выбрано 0.5. На рисунке 3.14 показано, что площадь под кривой PR с применением YOLO для класса 0 равна 0.942, для класса 1 – 0.895, а для класса 2 равна 0.874. Среднее значение данной кривой для всех классов составляет 0.904. С улучшенной архитектурой площадь под кривой PR с для класса 0 равна 0.961, для класса 1 – 0.921, а для класса 2 равна 0.927. Среднее значение данной кривой для всех классов составляет 0.937.



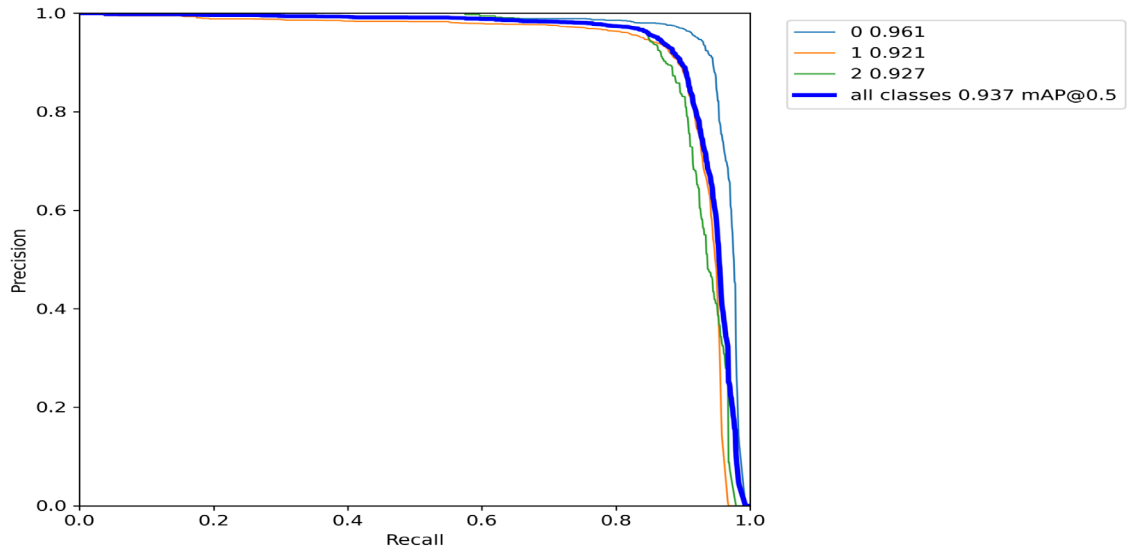
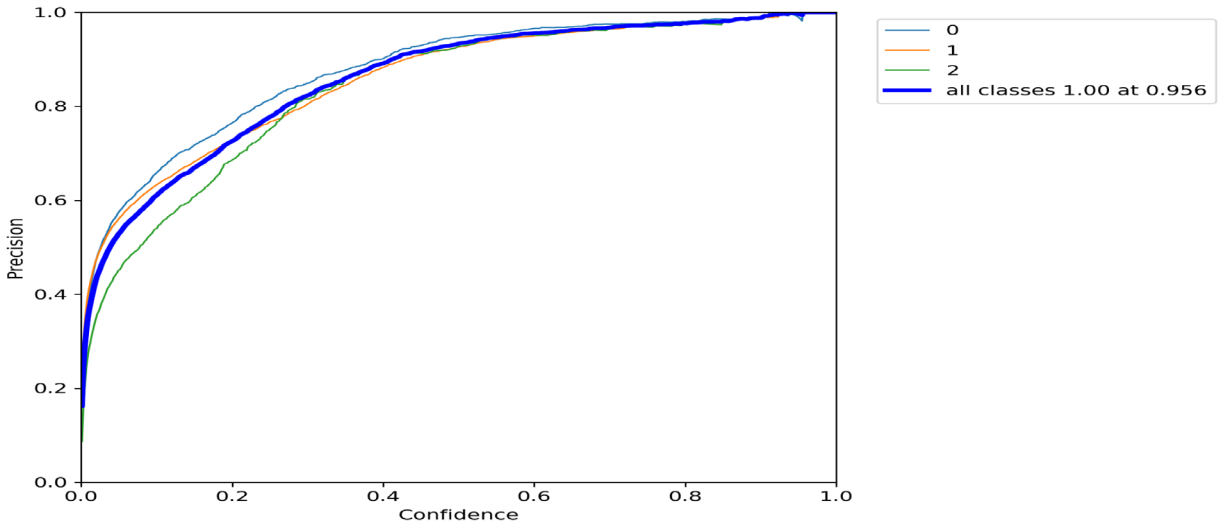


Рисунок 3.14 — Precision-recall curve

В YOLO одной из значительных метрик является confidence, которая предоставляет информацию о надежности прогнозов классификатора. Если увеличить порог confidence, повышается значение метрики precision, а recall будет снижаться. Рисунок 3.15 показывает, что в YOLO когда порог надежности был равен 0.956, все классы достигли идеальной точности, то есть значении 1.00. А в улучшенной архитектуре порог надежности был равен 0.945.



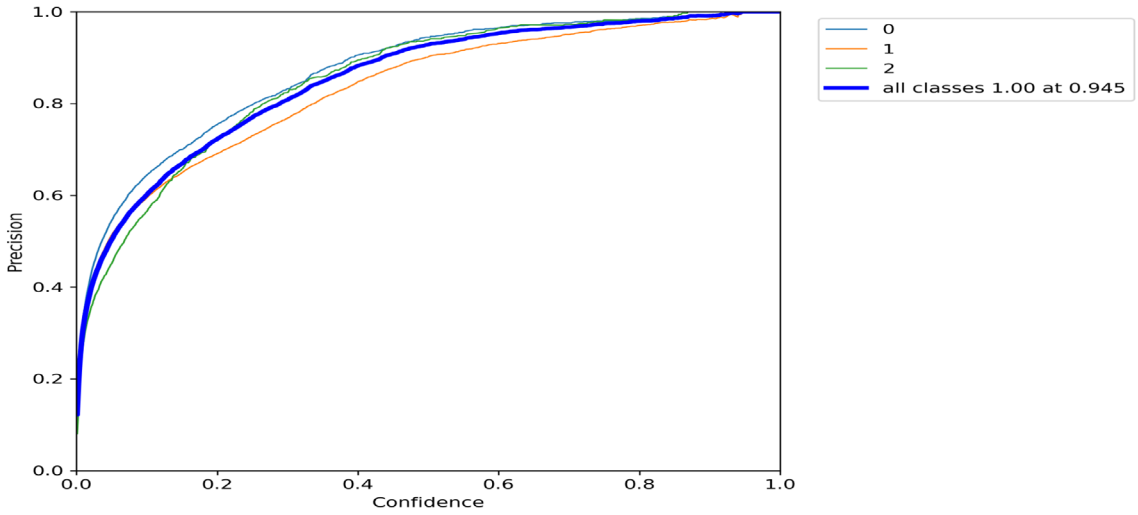
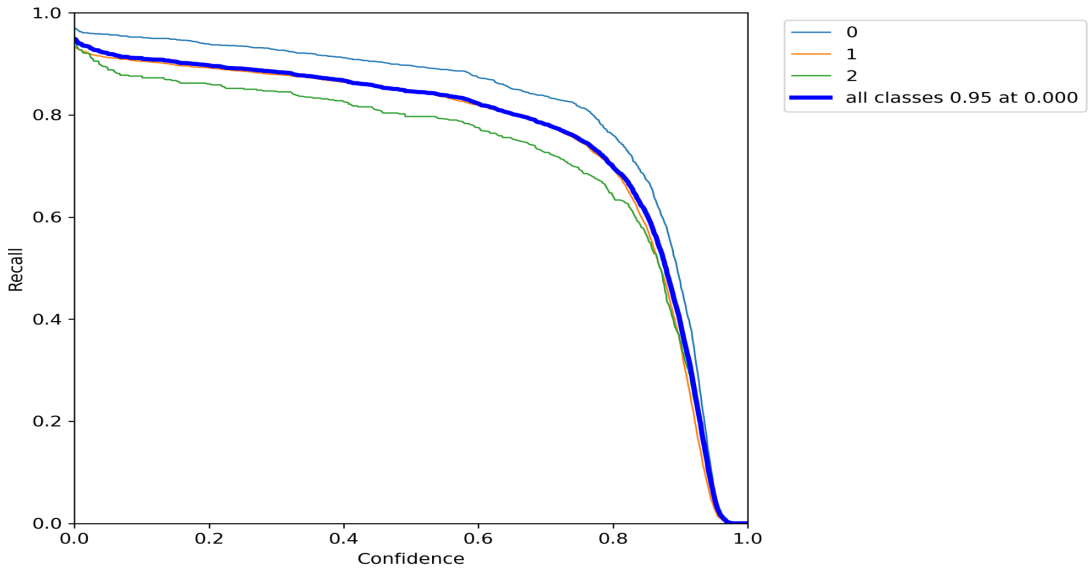


Рисунок 3.15 — Оценка precision трех классов для модели YOLO v5

На рисунке 3.16 показано как снижается значение оценки recall с увеличением порога надежности. С применением YOLO максимальный результат с показателем 0.95 был достигнут при пороге 0.00, тогда как в улучшенной архитектуре максимальный результат с показателем 0.95 был достигнут при пороге 0.00. В обеих архитектурах если порог confidence выбрать 1.00, recall будет равна 0.00, что является самым низким показателем метрики.



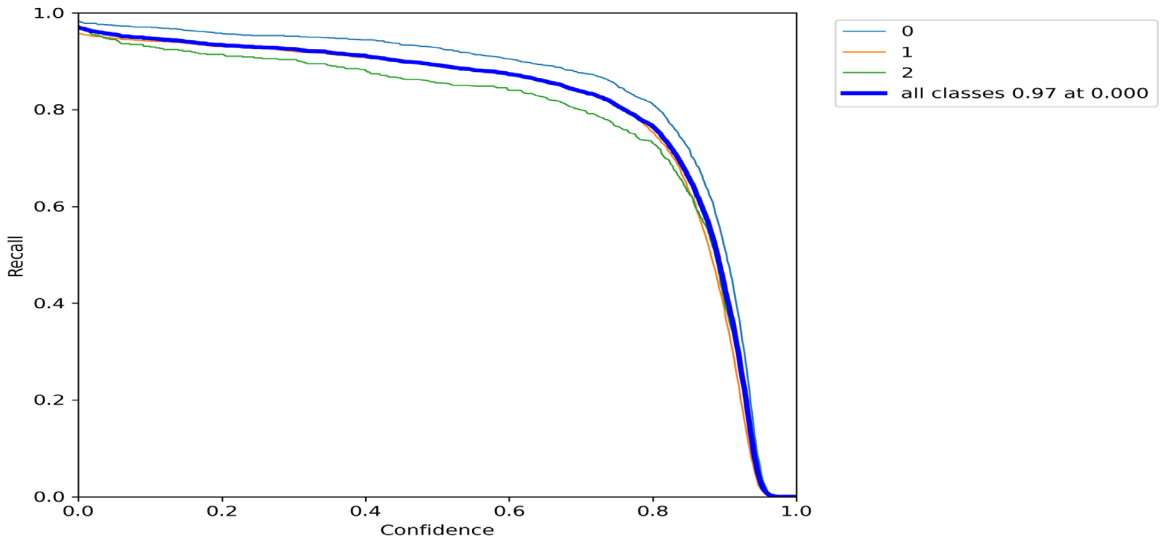
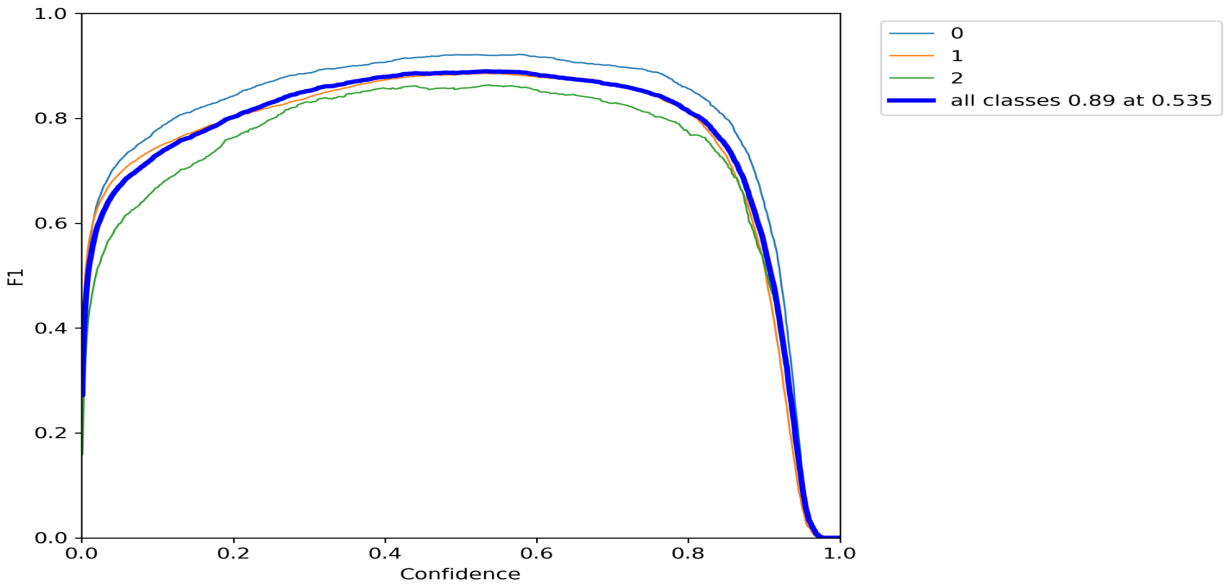


Рисунок 3.16 — Оценка recall трех классов для модели YOLO v5

Поэтому для уравновешивания баланса между precision и recall, понадобится метрика f1-measure, которая объединяет в себе информацию об этих метрик, и определяется как их среднее гармоническое значение. Как показано на рисунке 3.17, значение confidence из кривой f1, которое балансирует precision и recall составляет 0,573. При таком значении confidence, по графику можно заметить, что f1-measure находится между интервалами 0.90-0.95.



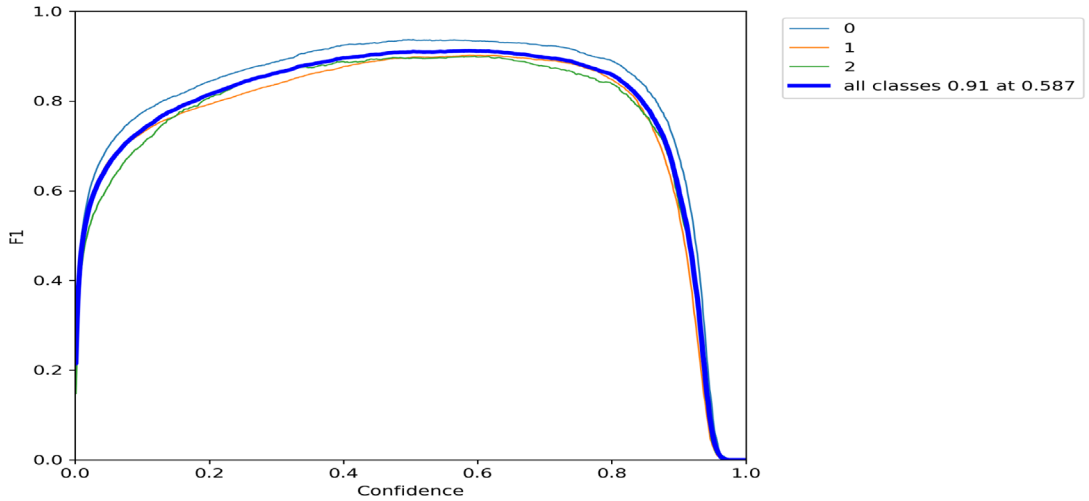
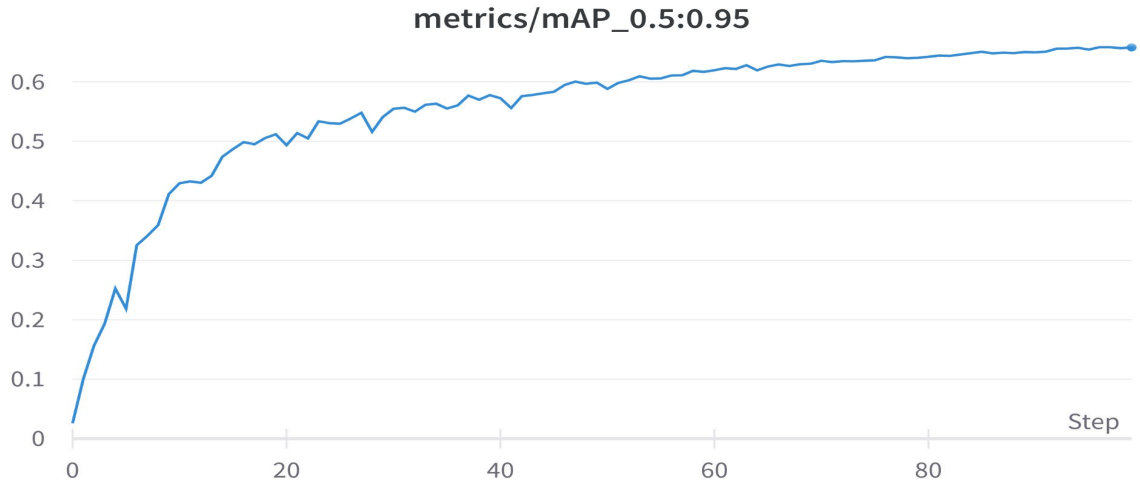


Рисунок 3.17 — Графическая интерпретация оценки F1

Для оценки моделей обнаружения объектов, таких как YOLO, часто применяется mean average precision. Так как наша модель имеет объектов, распределенных по 3 классам, сначала вычисляется средняя точность(AP) для каждого класса. Потом определяется среднее значение AP для всех классов, которое называется mAP. На рисунке 3.18 показаны результаты оценки mAP с шагом 0.5 для обеих архитектур.



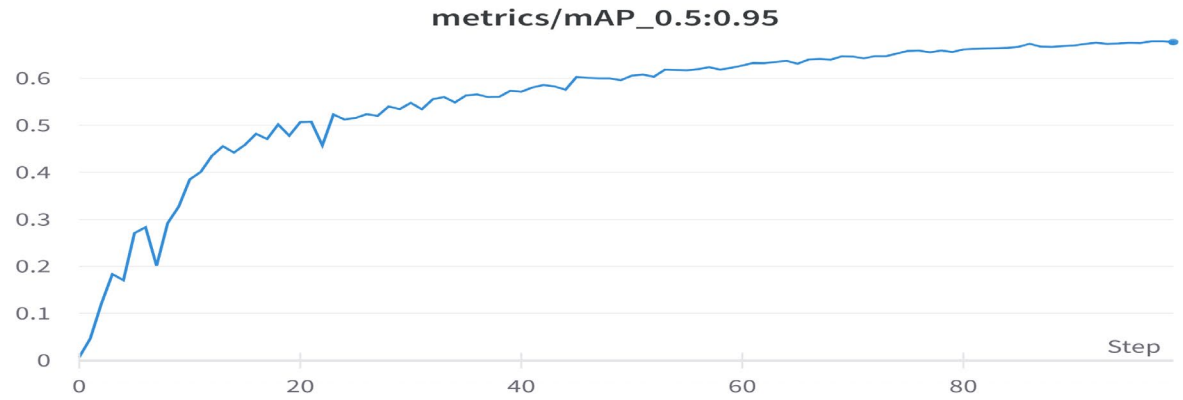


Рисунок 3.18 — Mean average precision

Результаты распознавания алгоритма показан на рисунке 3.19.

Рисунок 3.19 — Результаты распознавания томатов YOLO 5

Были подобраны изображения с разными освещенностями, с разных ракурсов и с разными уровнями шумов заднего фона.

Выводы по третьему разделу

Модель представления цветов имеют значительные важные недостатки и очень часто данные недоставки являются серьезными проблемами в области компьютерного зрения. К примеру, слабая освещённость и влияние теней могут снизить эффективность распознавания цвета помидора. Довольно сложно задать нужный цветовой тон, так как тон определяет степень отличия цвета. Исходя из этих двух недостатков, решено было использовать нейронную сеть Yolov5.

Использование YOLOv5 для обнаружения помидоров по нашим данным превышает метрики оценок качества, о которых сообщались в предыдущих исследованиях.

Анализ производительности при работе с небольшими объектами и изображениями с различным разрешением дополнительно подтверждает, что улучшенная сеть YOLOv5 обладает высокой надежностью для обнаружения объектов различных размеров и изображений с различным разрешением, что может удовлетворить потребности обнаружения томатов в сложных условиях.

Итоги проведенной оценки подтверждают, что улучшенный алгоритм YOLOv5 имеет высокую точность при обнаружении объектов.

4. РАЗРАБОТКА ПАРАЛЛЕЛЬНЫХ АЛГОРИТМОВ МАШИННОГО ОБУЧЕНИЯ ДЛЯ СИСТЕМЫ ОРИЕНТАЦИИ РОБОТА В ПРОСТРАНСТВЕ

В первом разделе были описаны методы и подходы по распознаванию объекта, сегментации, вычислению координат распознанного объекта. Во втором разделе реализовано применение классических алгоритмов для решения задач обнаружения и классификации объектов. Все эти решения необходимы для подачи данных на систему управления манипулятора. Как известно, для ряда манипуляционных задач целью управления является одновременное отслеживание положения и силы манипуляционных механизмов того самого робота-манипулятора. Исследователи разработали различные стратегии для достижения управления движением и силой для роботов при наличии ограничений окружающей среды. То, какое положение для сближения и обхвата объекта необходимо задаются именно при помощи положения и силы. В этом разделе рассматривается параллельное программирование на вычислительных ресурсах для робототехнических устройств.

4.1 Разработка параллельных алгоритмов обработки данных для робототехнического комплекса

В исследовании [22] рассматривается решение задачи вычисления трехмерных (3D) координат материальной точки. Для этой цели используются два плоских изображения (стереопара), которые соответствуют левой и правой точкам обзора 3D-сцены. Стереопара получается с помощью двух камер с параллельными оптическими осями. Для проверки теоретических результатов была проведена серия экспериментальных исследований. Во время этих экспериментов незначительное несоответствие было вызвано пространственным искажением камеры (искажением) оптической системы, а также ее несоответствием. Использование высококачественной стереоскопической системы, существующий расчет расхождений позволяет применять этот метод для широкого круга практических задач.

В [71] анализируются ошибки вычисления расстояния до объекта с использованием стереоскопической системы. Было установлено, что процентная погрешность при расчете расстояния обратно пропорциональна количеству пикселей, используемых при переключении между двумя изображениями, и прямо пропорциональна расстоянию до объекта.

Основываясь на обзоре проделанной работы, мы выделяем следующие выводы:

– Определение расстояния с помощью монокамеры не имеет широкого применения и исследований. Хотя использование монокамеры должно снизить затраты на производство технических устройств.

– Уменьшение погрешности определения расстояния исследования в области стереовидения недостаточно изучены и имеют перспективы для более детального изучения.

В исследовании мы попытались решить проблемы, полученные из обзоров литературы. В этом исследовании описывается эксперимент со стереокамерой, вычисление ошибки и определение параметров, повлиявших на эту ошибку. Использованы 2 метода для сравнения эффективности распознавания томатов: линейный метод и разработанный нами параллельный метод.

Принцип работы линейного метода заключается в том, что все алгоритмы будет реализованы последовательно. В начале процесса будут поступать данные с левой камеры ($camera_{left}$), после с правой камеры ($camera_{right}$). А в параллельном методе процессы будут распараллеливаны в начале, то есть обнаружение с двух камер будут происходить параллельно. После работа двух методов будут идентичны.

У нас имеется: label_left_queue, label_right_queue - лист очереди, где содержатся данные о признаках томатов, box_left_queue, box_right_queue - массив, в которых хранятся данные о рамках томатов, imgl, imgr - изображения взятые с левой и правой камеры. На рисунке изображена схема работы линейного метода.

Принцип работы работы паралельного алгоритма указана ниже:

Шаг 1: Примененить современный детектор компьютерного зрения YOLO, использующий триангуляцию для получения данных о координатах и границах ограничивающего прямоугольника с каждого распознанного объекта на изображении полученных с двух стереокамер ($camera_{left}$, $camera_{right}$) одновременно. Координаты помещаются в очередь, которая называется “queue”, при этом для каждой камеры имеется своя очередь. Изображение с ограничивающим прямоугольникам выводится на экран, а оригинальное изображение (необработанное) помещается в очередь. В очереди помещаются данные одного объекта. Если очередь оказывается не пустой, то снова проверяется очередь на наличие элементов. В данном исследовании мы ориентировались данным взятых с $camera_{left}$. Стоить отметить что обнаруженные объекты с $camera_{left}$ возможно не будут обнаружены с $camera_{right}$;

Шаг 2: Запустить цикл для перебора объектов обнаруженных с $camera_{left}$ (for i in range(0, len(label_left))), после внутри цикла запускаем еще один цикл для перебора объектов обнаруженных с $camera_{right}$ (for j in range(0, len(label_right))). Для того, чтобы идентифицировать объекты взятых с $camera_{left}$ на $camera_{right}$ задаем условие: если классы объектов равны между собой (if(label_left[i]==label_right[j])) и объекты находятся на одной высоте с разницей ± 20 пикселей по оси y. Бывает случаи что при данных условиях можно найти более одного объекта.

Шаг 3: Найти матрицу сходства. Матрица сходства вычисляется следующей формулой:

$$A = \begin{pmatrix} a_{00} & a_{01} & \cdots & a_{0m} \\ a_{10} & a_{11} & \cdots & a_{1m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n0} & a_{n1} & \cdots & a_{nm} \end{pmatrix}$$

$$B_k = \begin{pmatrix} b_{00} & b_{01} & \cdots & b_{0m} \\ b_{10} & b_{11} & \cdots & b_{1m} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n0} & b_{n1} & \cdots & b_{nm} \end{pmatrix}$$

$$C_k = |A - B_k|$$

$$c_{i,j} = |a_{i,j} - b_{i,j}|$$

$$\left| \begin{pmatrix} a_{00} & a_{01} & \cdots & a_{0m} \\ a_{10} & a_{11} & \cdots & a_{1m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n0} & a_{n1} & \cdots & a_{nm} \end{pmatrix} - \begin{pmatrix} b_{00} & b_{01} & \cdots & b_{0m} \\ b_{10} & b_{11} & \cdots & b_{1m} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n0} & b_{n1} & \cdots & b_{nm} \end{pmatrix} \right| = \begin{pmatrix} c_{00} & c_{01} & \cdots & c_{0m} \\ c_{10} & c_{11} & \cdots & c_{1m} \\ \vdots & \vdots & \ddots & \vdots \\ c_{n0} & c_{n1} & \cdots & c_{nm} \end{pmatrix}$$

Матрица A – это окно с центром в точке совпадающей с центром ограничивающего прямоугольника обнаруженного объекта на левом изображении. Размер матрицы A - n*m, то есть высоту и ширину окна мы вводим сами.

Матрицы B_k, k – количество матриц.

$$k = \sum_{i=0}^{n} \omega_i$$

где n - найденные объекты соответсвующие данным условиям. ω_i - ширина i обекта соответсвующего данным условиям. Центр матриц B, находятся на той же строке на правом изображении, что и центр матрицы A.

Матрица C – матрица сходства или разница по модулю между матрицами A и B.

Чтобы понять какая из матриц C_i наиболее точно отражает сходство между матрицами A и B_i, мы считаем количество наименьших элементов, путем сравнения соответсвующих элементов матриц C_i. После нахождения самой подходящей матрицы C_i, мы проверяем ее на подлинность с помощью порогового значения. Для этого сумма элементов матрицы C_i должна быть меньше порогового значения P. P зависит от размера матрицы C_i.

Шаг 4: После нахождения точной отражающей матрицы C_i, мы знаем центральные координаты найденного объекта.

При калибровке камеры мы получаем матрицу камеры размером 3x3.

$$\begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix}$$

где f_x – фокусное расстояние по оси X, f_y– фокусное расстояние по оси Y, (c_x, c_y) – координаты оптического центра. В нашем случае матрица выглядит так:

$$\begin{pmatrix} 816.04 & 0 & 288.74 \\ 0 & 807.93 & 238.68 \\ 0 & 0 & 1 \end{pmatrix}$$

Для вычисления расстояния до объекта берем среднее фокусное расстояние:

$$f = \frac{f_x + f_y}{2}$$

$$f = \frac{816.04 + 807.93}{2} \approx 812$$

T_x – расстояние между камерами равное 0.15 метров.

Так как камеры откалиброваны достаточно относительно друг от друга по высоте, глубине и наклону, то мы считаем что пиксель с координатами x1, y1 в левом изображении имеет координаты x1-d1, y1 в правом изображении, где d1 это значение смещения для этого пикселя.

Находим все смещения для каждого пикселя и строим карту смещений или несоответствий.

После нужно вычислить разницу координат по оси x между левым и правым изображениями или несоответсвие. Несоответствие вычисляется следующей формулой:

$$disparity = x - x'$$

где x – центр координат ограничивающего прямоугольника обнаруженного объекта по оси X в левом изображении, x' – в правом.

Шаг 5: Для того чтобы описать объекты в пространстве нужно найти мировые координаты.

Мировые координаты:

$$Q = \begin{pmatrix} 1 & 0 & 0 & -c_x \\ 0 & 1 & 0 & -c_y \\ 0 & 0 & 0 & f \\ 0 & 0 & -\frac{1}{T_x} & \frac{c_x - c'_x}{T_x} \end{pmatrix}$$

Матрица Q, где (c_x, c_y) – координаты оптического центра левой камеры, c'_x – координата оптического центра по оси X правой камеры $c'_x = c_x$, f – фокусное расстояние левой камеры, T_x – расстояние между камерами:

$$Q = \begin{pmatrix} 1 & 0 & 0 & -288 \\ 0 & 1 & 0 & -238 \\ 0 & 0 & 0 & 812 \\ 0 & 0 & -\frac{1}{15} & 0 \end{pmatrix}$$

$$Q * \begin{pmatrix} x \\ y \\ disparity \\ 1 \end{pmatrix} = \begin{pmatrix} X \\ Y \\ Z \\ W \end{pmatrix}$$

(x, y) – координаты, *disparity* – несоответствие. *X, Y, Z, W* – однородные координаты.

$$\begin{pmatrix} X/W \\ Y/W \\ Z/W \\ W/W \end{pmatrix} = \begin{pmatrix} X/W \\ Y/W \\ Z/W \\ 1 \end{pmatrix}$$

X/W, Y/W, Z/W – мировые координаты в реальном мире.

Шаг 6: Последним шагом является нахождение глубины для проверки точности вычисления мировых координат в реальном мире. Глубина вычисляется формулой указанной ниже:

$$depth = \frac{f * T_x}{disparity}$$

Основываясь на разработанном алгоритме, можно повысить точность и скорость вычисления расстояния за счет использования алгоритмов параллельных вычислений.

Измерение расстояния между роботом и объектом необходимо для контроля действий робота, например, захвата объекта или даже избегания препятствий [72,73]. Существует множество методов оценки расстояния,

таких как ультразвуковые, лазерные и (видео, фото) камеры. Техника, основанная на видеокамере, имеет преимущество в ее низкой стоимости, поэтому в этой работе применяется метод измерения расстояния между стереокамерой и объектом (томатом).

В статье [74] сравнивается производительность двух стратегий распараллеливания для нейронной сети обратного распространения на кластерном компьютере: примерной параллельной и узловой параллельной стратегий. Уравнения для расчета теоретических затрат этих двух стратегий предложены на основе реализации, представленной в статье. Результаты производительности сопоставляются в соответствии с различными размерами нейронной сети, различными размерами набора данных и количеством процессоров. Результаты работы показывают преимущества и недостатки этих двух стратегий. Было обнаружено, что экспериментальные результаты очень хорошо согласуются с теоретическими затратами. Поэтому представленные уравнения затрат можно использовать для прогнозирования того, какая стратегия будет лучше, учитывая размер сети, размер набора данных и количество процессоров.

4.2 Применение результатов к роботу-манипулятору в задачах сбора урожая

В этом подразделе предлагается один из подходов к роботизации процедур сбора урожая томатов. Ручная уборка томатов является трудоемкой и неэффективной, типичной для интенсивной работы, что делает ее непрактичной, поскольку их растения становятся крупномасштабными полями. Более того, помидор очень мягкий и склонен к образованию синяков, что также затрудняет сбор урожая и процесс захвата для внедрения автоматической системы тестирования. Кроме того, одной из главных проблем в сельскохозяйственном секторе является рост затрат на рабочую силу и нехватка рабочей силы, поскольку сельскохозяйственный труд не пользуется популярностью среди молодежи. Поэтому, многие технологические проблемы еще предстоит решить, когда мы примем их во внимание при практической реализации. Роботизация и автономизация уборки томатов может быть одним из наиболее эффективных перспективных решений для устранения вышеупомянутых проблем.

Основными компонентами при проектировании уборочных роботов являются движущаяся платформа, достигающий манипулятор, захватный инструмент и алгоритм распознавания помидоров. В этой части диссертационной работы предложена манипулирующая рука робота, захватный инструмент.

Основной компонент, который принимается во внимание, - механическая рука, поскольку наиболее важной проблемой на уборочном манипуляторе является рука. Многие исследователи и инженеры предложили различные типы роботизированных манипуляторов для сбора урожая. Во многих примерах используются коммерчески доступные манипуляторы с

жесткими связями, такие как KUKA и универсальные роботы [75-77]. Аналогично, в [78] использовался последовательный манипулятор типа SCARA, установленный на верхней части мобильного робота. Однако в работе [79] предложен манипулятор континуума харды с гибкой структурой, который оказался безопасным с широкой доступностью, но обладающим низкой грузоподъемностью.

Второй сложной проблемой при сборе плодов является конструкция захвата. Помидор - хрупкий и мягкий фрукт, поэтому хватать помидор следует осторожно, чтобы избежать ушибов и ударов. Кроме того, отсоединение помидора также является сложной задачей. Например, в [80] предложен робот SCARA с двумя руками, в котором одна рука держит помидор, а вторая рука срезает стебель. Это требует дополнительной подготовки для обнаружения стеблей томатов, и два рычага должны работать синхронно с высокой точностью. Кроме того, другие исследователи предложили захват для выщипывания с бесконечным поворотным соединением для автоматического отсоединения помидоров [81], но вероятность успеха отсоединения в реальном применении составила всего 60%. Аналогичная конструкция захвата была также предложена компанией Root Ai и ее роботом Virgo, имеющим рычаг типа SCARA, который отделяет помидоры, скручивая их после захвата. [82]. Компания Panasonic также представила коммерчески доступный прототип робота-сборщика помидоров. Робот тратит на это всего 2-3 секунды на один помидор, и теперь он выбирает цикл, который будет самой быстрой машиной прототипа [83]. Однако вышеупомянутые прототипы не могут отделять помидоры вместе с их чашелистиками, что может вызвать проблемы с точки зрения долговечности томатов во время транспортировки.

В этом исследовании мы представляем новую роботизированную систему для уборки томатов с манипулятором и системой обнаружения томатов. В качестве манипулятора мы используем дискретную управляемую проводом непрерывную роботизированную руку TakoBot [84-86].

В соответствии с предполагаемым применением робот должен обладать следующими функциями:

- Гибкая структура для работы в ограниченных рабочих пространствах;
- Жесткая конструкция и приличная точность движения;
- Грузоподъемность более 100 г;
- Хватание помидоров без причинения вреда;
- Способность различать зрелость томатов;
- Способность отделять помидоры.

TakoBot - это дискретный гипер-резервированный манипулятор непрерывного действия с кабелем с двумя исполнительными секциями. Он состоит из трех основных частей: рычага непрерывного действия, блока предварительного натяжения и блока управления. Непрерывная часть делится на две секции: первая секция расположена в дистальной части, а

вторая секция расположена в проксимальной части, как показано на рисунке 4.1.

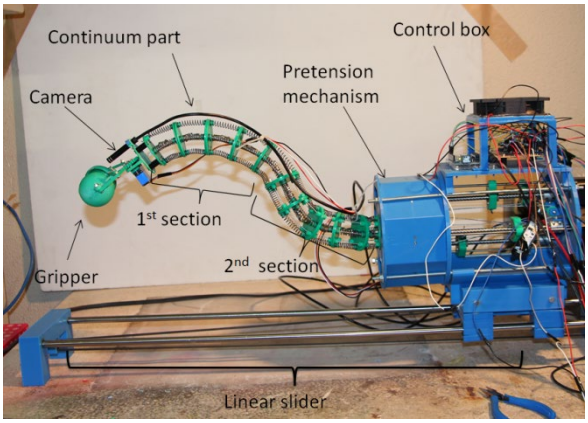


Рисунок 4.1 — TakoBot

Каждая секция содержит последовательно соединенные пять сегментов, и каждый из них управляется четырьмя проводами, которые заключены в отдельные пружины сжатия. Таким образом, всего восемь проводов управляют TakoBot-ом. Один сегмент состоит из двух распорных дисков, соединенных между собой универсальным шарниром (см. рисунок 4.2 (а)) и четырьмя пружинами сжатия для жесткости (см. рисунок 4.2 (б)).

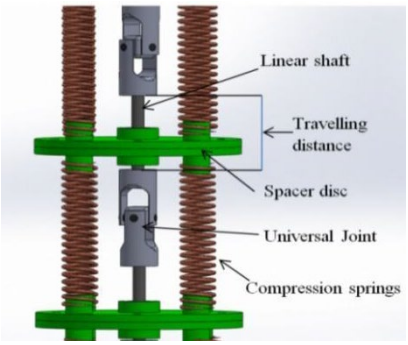
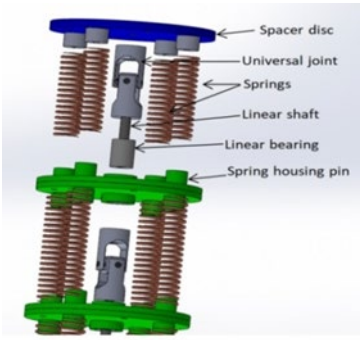
	
а)	б)

Рисунок 4.2 — Визуализация одного сегмента

Четыре провода, управляющие движениями манипулятора, заключены в четыре пружины сжатия, как показано на рисунке 4.3, расположение которых позволяет сжимать пружины с минимальной вероятностью непредсказуемого изгиба.

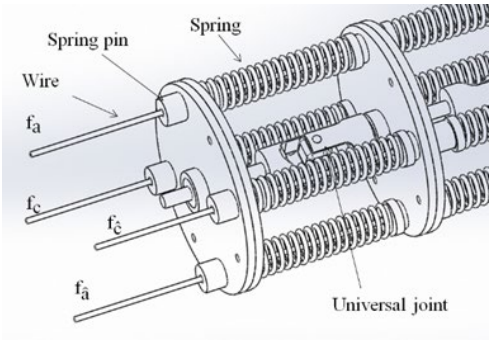


Рисунок 4.3 — Визуализация пружины сжатия

В центре диска установлены линейные подшипники, которые позволяют скользить диску вдоль линейного вала. Кроме того, часть предварительного натяжения, предназначенная для компенсации натяжения кабеля и обеспечения жесткости конструкции робота. Последняя часть представляет собой блок управления, который приводит в действие провода (см. рисунок 4.4).

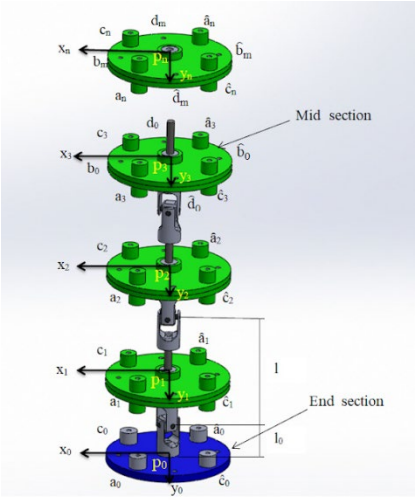


Рисунок 4.4— Визуализация распорных дисков

В данном рисунке $H_{0,1}$ первая (нижняя) секция

Как показано на рисунке 4.4, распорный диск может скользить вдоль направляющего вала. Механически диск может перемещаться на расстояние (всего 10 мм, но такое скольжение не влияет на общую длину манипулятора, такая конструкция способствует увеличению длины) между соседними универсальными шарнирами. Этот механизм скользящего диска позволяет избежать локальной чрезмерной концентрации сжатия пружины за счет равномерного распределения силы пружины между сегментами, что способствует распределению внутреннего напряжения вдоль тонкой детали. Аналогичным образом, конструкция обеспечивает разумное распределение напряжений при изгибе для ближайших сегментов пружин сжатия. Кроме того, такая конструкция обеспечивает более острый угол изгиба и

компенсирует натяжение проволоки во время движения стабилизирующих манипуляторов и предотвращает срыв проволоки со шкивов в результате.

При разработка захватного инструмента нужно учитывать, что захват помидора должен быть мягким, чтобы предотвратить любые сбои, такие как избыточное давление. Кроме того, конструкция инструмента должна предусматривать отсоединение томата от стебля помидора (см. рисунок 4.5).

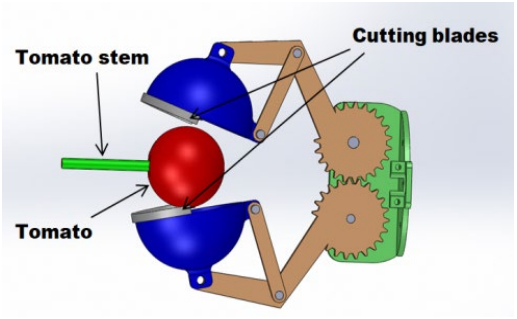


Рисунок 4.5 — Конструкция захватного инструмента

На рисунке представлен разработанный нами захватный инструмент полусферической формы для захвата сферических объектов, таких как помидор. Для отделения помидоров мы добавили режущие лезвия по краям чашки. Такая конструкция позволяет отделять помидоры в последовательной процедуре, и это увеличивает время сбора урожая (см. рисунок 4.6).

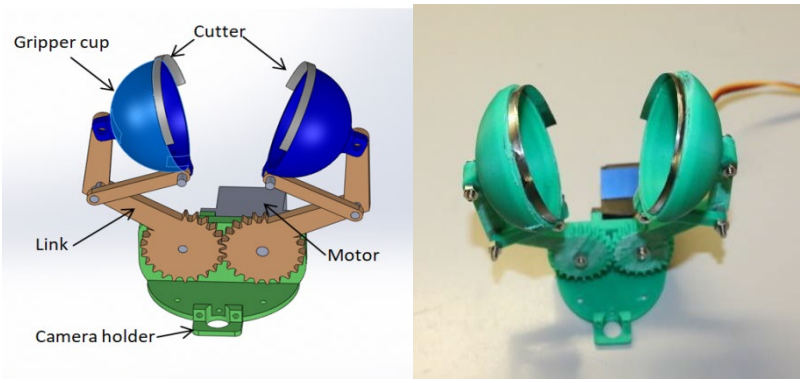


Рисунок 4.6 — Разработанный захватный инструмент

По сравнению с другими захватами этот прототип может разделяться последовательно и не требует датчиков для управления. Отсутствие электроники позволяет роботу работать во влажной среде. Кроме того, предлагаемый захват может разрезать помидор с его уплотнением, что помогает увеличить время хранения собранных помидоров, в то время как другие прототипы оставляют чашелистик на стебле**.**

Архитектура управления TakoBot состоит из двух основных частей: программного и аппаратного обеспечения (см. рисунок 4.7). Рабочий процесс начинается с программного обеспечения, которая описана в предыдущем

подразделе. После измеренная информация помогает рассчитать обратную кинематику робота. Рассчитанная обратная кинематика отправляет информацию и координаты помидора на плату Arduino. Таким образом, плата Arduino отправляет данные в двигатели для управления TakoBot, чтобы заставить захват занять нужное положение.

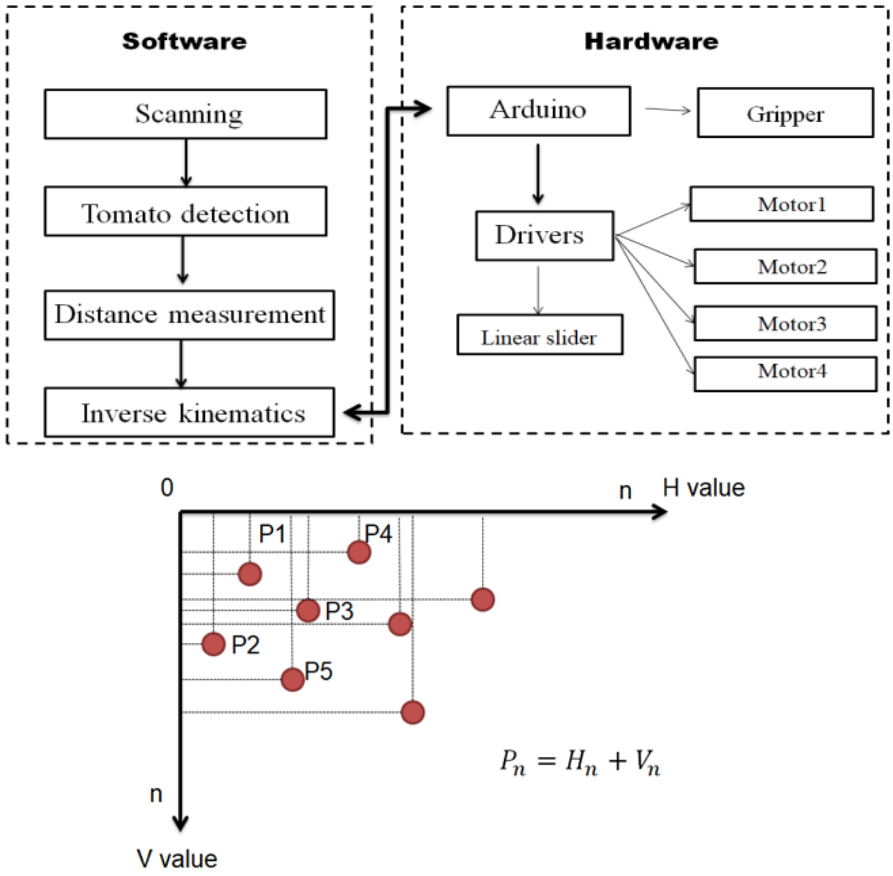
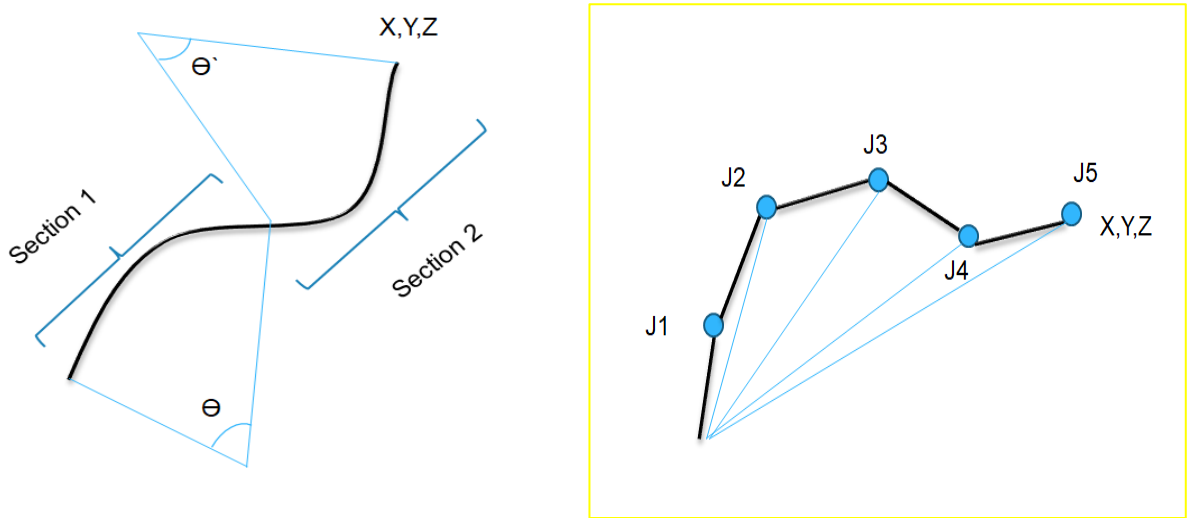


Рисунок 4.7 — Архитектура TakoBot

В начале уборки томатов нужно определить позицию робота от нулевой координаты базы до конечного эффектора (захватный инструмент). Для этого вычисляется прямая кинематика. Имеется 2 способа вычисления прямой кинематики, которая показана на рисунке 4.8. В данной работе была применена секционная система.



а) Постоянная кривая б) Секционная система

Рисунок 4.8 — Виды вычисления прямой кинематики

Стоит отметить что системы координат задаются на каждом универсальном шарнире. Для прямой кинематики вычисляется гомогенное преобразование.

$$H_{0,i} = H_{0,1}H_{1,2}\cdots H_{i-1,i} = \begin{pmatrix} i_i & j_i & k_i & u_i \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\sum_0 \to \sum_1 : H_{0,1} = \begin{pmatrix} R_y(\theta_{y1}) & R_x(\theta_{x1}) & 0 & u_{0,1} \\ 0 & 0 & 0 & 1 \end{pmatrix}, u_{0,1} = \begin{pmatrix} 0 \\ 0 \\ l_1 \end{pmatrix}$$

$$\sum_{i-1} \to \sum_i : H_{i-1,i} = \begin{pmatrix} R_y(\theta_{yi}) & R_x(\theta_{xi}) & 0 & u_{i-1,i} \\ 0 & 0 & 0 & 1 \end{pmatrix}, u_{i-1,i} = \begin{pmatrix} 0 \\ 0 \\ 2l \end{pmatrix}$$

$H_{0,i}$ - Гомогенное преобразование
u_i- матрица трансляция поступательного движения
l_1-растояние между дисками

Гомогенное преобразование состоит из матрицы вращения (rotation) и матрицы трансляции (translation). Две матрицы вращения показаны ниже:

$$R_x(\theta_{xi}) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_{xi} & -\sin\theta_{xi} \\ 0 & \sin\theta_{xi} & \cos\theta_{xi} \end{pmatrix}$$

$$R_y(\theta_{yi}) = \begin{pmatrix} \cos\theta_{yi} & 0 & \sin\theta_{yi} \\ 0 & 1 & 0 \\ -\sin\theta_{yi} & 0 & \cos\theta_{yi} \end{pmatrix}$$

R_x, R_y - матрица вращения

После получения матрицы с помощью прямой кинематики мы используем метод Якобиан. Якобиан применяется для того, чтобы описать взаимосвязь между скоростями суставов и скоростями конечного эффектора. Следовательно Якобиан используется для определения скоростей суставов для перемещения конечного эффектора вдоль заданного вектора. Якобиан вычисляется следующей формулой:

$$\Delta p_n = \frac{\partial p_n}{\partial v}\Delta v + \frac{\partial p_n}{\partial \varphi}\Delta\varphi$$

где v линейная скорость, φ угловая скорость.

Кинетика это когда во время движения учитывается динамика. В данной работе она была применена чтобы вычислить упругость пружины.

$$S_{\sigma,i} = k(L_s - |\sigma_i - \sigma_{i-1}|), S_{\hat{\sigma},i} = k(L_s - |\hat{\sigma}_i - \hat{\sigma}_{i-1}|)$$

$S_{\sigma,i}$ – упругость пружины, k – коэффициент жесткости пружины.

Так как в каждой секции 4 пружины, то статистический эквилибриум одной секции вычисляется следующим образом:

$$\begin{aligned}
&\left(-S_{a,i+1}(\overline{a_{i+1} - a_i}) + S_{a,i}(\overline{a_i - a_{i-1}})\right) \times (a_i - u_i) \\
&+ \left(-S_{\hat{a},i+1}(\overline{\hat{a}_{i+1} - \hat{a}_i}) + S_{\hat{a},i}(\overline{\hat{a}_i - \hat{a}_{i-1}})\right) \times (\hat{a}_i - u_i) \\
&+ \left(-S_{c,i+1}(\overline{c_{i+1} - c_i}) + S_{c,i}(\overline{c_i - c_{i-1}})\right) \times (c_i - u_i) \\
&+ \left(-S_{\hat{c},i+1}(\overline{\hat{c}_{i+1} - \hat{c}_i}) + S_{\hat{c},i}(\overline{c_i - \hat{c}_{i-1}})\right) \times (\hat{c}_i - u_i) \\
&+ \left(m_w(p_n - u_i) + m_p \sum_{k=i+1}^{n-1}(p_k - u_i)\right) \times g = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}
\end{aligned}$$

^ данный знак означает противоположность.
Данная формула основана на законе Гука. Закон Гука:

$$F_x = -kx$$

Механиз компенсирования упругости пружины:

$$f_\sigma = \frac{1}{2}k_p\left(2\bar{u}_{p\sigma} + \frac{\lambda\varphi_\sigma}{4\pi} + \frac{1}{2}\left(\sum_{i=1}^{n}|\sigma_i - \sigma_{i-1}| - nL\right)\right)$$

$$f_{\hat{\sigma}} = \frac{1}{2}k_p\left(2\bar{u}_{p\hat{\sigma}} - \frac{\lambda\varphi_\sigma}{4\pi} + \frac{1}{2}\left(\sum_{i=1}^{n}|\hat{\sigma}_i - \hat{\sigma}_{i-1}| - nL\right)\right)$$

$$2f_\sigma = k_p u_{p\sigma}, \sigma = a, b, c, d$$
$$2f_{\hat{\sigma}} = k_p u_{p\hat{\sigma}}, \hat{\sigma} = \hat{a}, \hat{b}, \hat{c}, \hat{d}$$

$$2u_{p\sigma} = 2\bar{u}_{p\sigma} + \frac{\lambda\varphi_\sigma}{2\pi} + \sum_{i=1}^{n} |\sigma_i - \sigma_{i-1}| - nL$$

$$2u_{p\hat{\sigma}} = 2\bar{u}_{p\hat{\sigma}} - \frac{\lambda\varphi_\sigma}{2\pi} + \sum_{i=1}^{n} |\hat{\sigma}_i - \hat{\sigma}_{i-1}| - nL$$

где $u_{p\sigma}, u_{p\hat{\sigma}}$ – длина упругости пружины предварительного натяжения, жесткость которой равна k_p

φ_p- угол поворота двигателя для создания предварительного натяжения

λ- вывод винтового стержня

k_p - жесткость пружины предварительного натяжения

После вычисляется обратная кинематика. Обратная кинематика способствует изменению углов двигателя начиная с конечного эффектора до объекта. Она вычиляется следующим образом:

$$\begin{pmatrix} p_n \\ 1 \end{pmatrix} = H_{0,n} \begin{pmatrix} 0 \\ 0 \\ l_n \\ 1 \end{pmatrix} = \begin{pmatrix} i_n & j_n & k_n & r_n \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ l_n \\ 1 \end{pmatrix} = \begin{pmatrix} k_n l_n + r_n \\ 1 \end{pmatrix}$$

В качестве контроллера была использована плата Arduino Uno и драйверы двигателей TMC2208 для шаговых двигателей. Для непрерывной работы мы также оборудовали вентилятор для охлаждения электронных деталей с помощью кулера. TakoBot имеет четыре двигателя и один шаговый двигатель для линейного перемещения вдоль линейного ползунка, а также один микро-сервомотор в конечном эффекторе.

Для эксперимента помидоры повесили перед манипулятором. Задача состояла в том, чтобы дотянуться и схватить помидор, отделить его от стебля и положить в корзину. Поэтому в ходе эксперимента проверили управляемость робота, например, возможность дотягиваться до объекта под разными углами (см. рисунок 4.9).

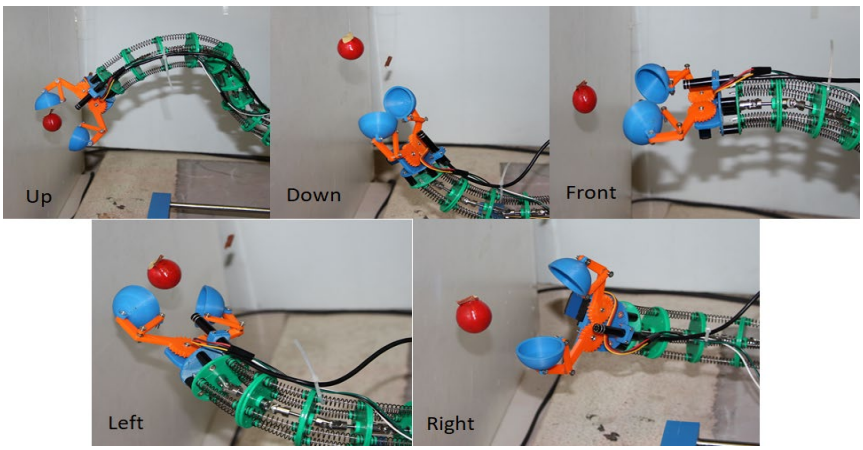


Рисунок 4.9 — Манипуляция сбора урожая под разными углами

Общая длина манипулятора робота составляет 800 мм, тонкая часть или подвижная часть робота 400 мм, остальные 400 мм - это блок управления и исполнительные механизмы. Длина ползуна платформы составляет 1000 мм, но доступно только 600 мм, потому что стационарная часть манипулятора занимает 400 мм.

Согласно проведенному эксперименту, TakoBot продемонстрировал высокую работоспособность при работе в ограниченном рабочем пространстве, а также высокую достижимость. Для выполнения поставленной задачи было достаточно одной единственной руки. В качестве реального эксперимента на рабочем пространстве были установлены дополнительные препятствия, чтобы проверить достижимость робота и способность робота-манипулятора обходить препятствия. В этом эксперименте мы использовали твердую белую бумагу, чтобы имитировать ограниченное рабочее пространство.

В ходе исследования мы обнаружили, что при существующих препятствиях время сбора урожая увеличивается. В таком случае тонкая часть робота потратила больше времени на то, чтобы приспособиться к новой ограниченной среде и добраться до объекта (см. рисунок 4.10). Кроме того, процесс захвата помидоров также занимает больше времени, занимая почти половину времени всего процесса сбора урожая. Однако вероятность успеха помидора была достаточно высока, манипулятор смог захватить все обнаруженные помидоры, но потратил больше времени на выполнение задачи.

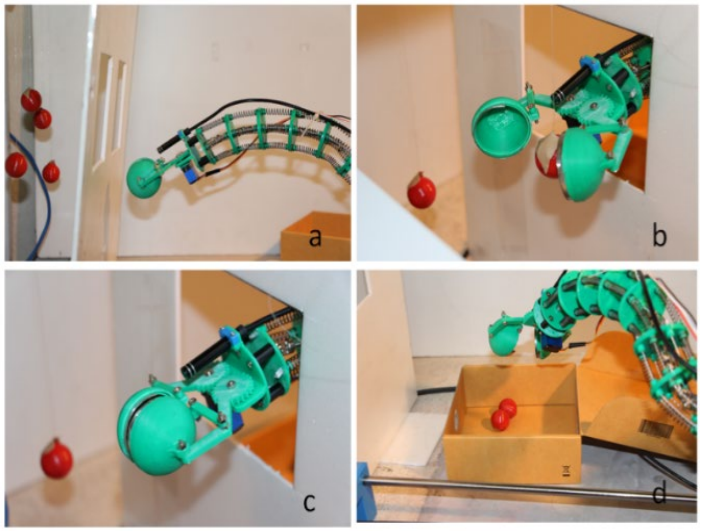


Рисунок 4.10 — Манипуляция сбора урожая с препятствиями

На рисунке 4.11 показана экспериментальная установка, в этом эксперименте трекер использовался для отслеживания пути конечного эффектора и времени сбора урожая. Система координат установлена в базовой части манипулятора робота continuum. Исходя из полученного результата, основное время сбора урожая уходит на то, чтобы избежать препятствий и захватить помидоры.

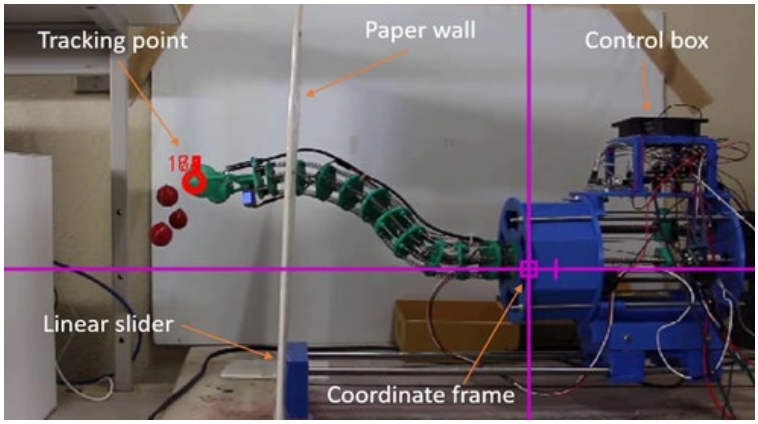


Рисунок 4.11 — Экспериментальная установка

Проведенные эксперименты показали, что предложенная конструкция захвата для захвата томатов эффективна с точки зрения отделения томатов и их размещения в корзине. Предложенная конструкция полусферической захватной чашки продемонстрировала хорошую способность захвата, не повреждая помидоры даже в случае неправильного положения.

Выводы по четвертому разделу

Разработанная параллельная модель программирования на вычислительных ресурсах для робототехнических устройств показал наилучший результат, по сравнению с линейным методом. Данное утверждение подтверждает, что за счет использования алгоритмов параллельных вычислений, можно повысить точность и скорость вычисления расстояния.

Разработана интеллектуальный-роботизированный комплекс Takobot реализованный на основе новой параллельной модели распознавания томатов для выполнения автономной работы по сборке томатов.

ЗАКЛЮЧЕНИЕ

В результате разработки эффективных параллельных алгоритмов для системы ориентации робота в пространстве, использующих методы машинного обучения были получены следующие результаты.

– Проанализировано современное состояние проблемы распознавания объектов при сборке томатов, реализованы несколько методов для сегментации объектов, а также приведены сравнение результатов данных методов между собой. Проанализированы существующие методы машинного обучения, используемые для классификации томатов. Выполнен сравнительный анализ методов машинного обучения классификации объектов (томатов) с оценкой качества на основе вычислительного эксперимента.

– Разработана модифицированная архитектура YOLOv5 для сверочной нейронной сети с оценкой качества распознавания изображений. Использование модели позволяет точно распознать томаты.

– Разработан алгоритм распараллеливания процессов обработки изображений с вычислением трехмерных координат объектов. Используя данную идею, можно повысить скорость распознавания томатов.

– Разработана архитектура многозвенного робота с машинным зрением, позволяет определять локализацию в пространстве исследуемого объекта.

– Сконструирован агроробот, предназначенный для сбора томатов, проведением экспериментальных работ в реальном масштабе времени.

Задачи исследования диссертационной работы полностью решены. Разработанные интеллектуальный роботизированный комплекс Takobot основанный на новой параллельной модели распознавания томатов показывает хорошие результаты.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Thrun S. A lifelong learning perspective for mobile robot control [Electronic resource] / S. Thrun // IEEE International Conference on Intelligent Robots and Systems : IROS 1994, Мюнхен, 12-16 сентября 1994. — Institute of Electrical and Electronics Engineers Inc., 1994. — Режим доступа: https://doi.org/10.1109/IROS.1994.407413.

2. Raj M., Seamans R. Primer on artificial intelligence and robotics [Электронный ресурс] // Journal of Organization Design. — 2019. — Т. 8, № 1. — Режим доступа: https://doi.org/10.1186/s41469-019-0050-0.

3. Ren S., Yang G. The innovative design of the robot picking fruits / S. Ren, G. Yang // TEIN 2011 - 2011 2nd ETP/IITA Conference on Telecommunication and Information : 2011 2nd ETP/IITA Conference on Telecommunication and Information, TEIN 2011, 2011, 3-4 апреля 2011. — Engineering Technology Press, 2011.

4. Banjanović-Mehmedović L. Service Robots: Advances in Research and Application. — New York : Nova Science Publishers, 2021. — 55-75 с. — ISBN 978-153619573-6.

5. Ступина Е.Е., Ступин А.А., Чупин Д.Ю., Каменев Р.В. Основы робототехники: учебное пособие. — Новосибирск: Агентство «Сибпринт», 2019. — 160 с. ISBN 978-5-94301-769-8.

6. Kazakhstan - Agricultural Sector // https://www.privacyshield.gov/article?id=Kazakhstan-Agricultural-Sector / [Электронный ресурс]: 25.07.19.

7. Nefti-Meziani S., Manzoor U., Davis S., Pupala S.K. 3D perception from binocular vision for a low cost humanoid robot NAO [Электронный ресурс] // Robotics and Autonomous Systems. — 2015. — Т. 68. — С. 129-139. — Режим доступа: https://doi.org/10.1016/j.robot.2014.12.016.

8. Panigrahi P.K., Bisoy S.K. Localization strategies for autonomous mobile robots: A review [Электронный ресурс] // Journal of King Saud University - Computer and Information Sciences. — 2021. — ISSN 13191578. — Режим доступа: https://doi.org/10.1016/j.jksuci.2021.02.015.

9. Fong S., Deb S., Chaudhary A. A review of metaheuristics in robotics [Электронный ресурс] // Computers & Electrical Engineering. — 2015. — Т. 43. — С. 278-291. — Режим доступа: https://doi.org/10.1016/j.compeleceng.2015.01.009.

10. Huang Y.P., Wang D.Z., Zhou H.Y., Yang Y.T., Chen K.J. Ripeness Assessment of Tomato Fruit by Optical Absorption and Scattering Coefficient Spectra // Guang Pu Xue Yu Guang Pu Fen Xi/Spectroscopy and Spectral Analysis. — 2020. — Т. 40, № 11. — С. 3556 - 3561.

11. Huang Y.P., Lu R., Qi C., Chen K.J. Tomato Maturity Classification Based on Spatially Resolved Spectra // Guang Pu Xue Yu Guang Pu Fen Xi/Spectroscopy and Spectral Analysis. — 2018. — Т. 38, № 7. — С. 2183 - 2188.

12. Huang Y., Lu R., Hu D., Chen K. Quality assessment of tomato fruit by optical absorption and scattering properties [Электронный ресурс] // Postharvest Biology and Technology. — 2018. — Т. 143. — С. 78-85. — Режим доступа: https://doi.org/10.1016/j.postharvbio.2018.04.016.

13. Al-Mashhadani Z., Chandrasekaran B. Autonomous Ripeness Detection Using Image Processing for an Agricultural Robotic System [Электронный ресурс] // 2020 11th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON). — 2020. — Режим доступа: https://doi.org/10.1109/uemcon51285.2020.9298168.

14. Dhakshina Kumar S., Esakkirajan S., Bama S., Keerthiveena B. A microcontroller based machine vision approach for tomato grading and sorting using SVM classifier [Электронный ресурс] // Microprocessors and Microsystems. — 2020. — Т. 76. — С. 103090. — Режим доступа: https://doi.org/10.1016/j.micpro.2020.103090.

15. Dhakshina K. S., Esakkirajan S., Bama S. A nondestructive tomato grading and sorting mechanism using binary SVM pair-based classifier for sustainable agricultural practices // Journal of Green Engineering. — 2020. — Т. 10, № 7. — С. 3637-3654.

16. Huang Y.P., Wang T.H., Basanta H. Using Fuzzy Mask R-CNN Model to Automatically Identify Tomato Ripeness [Электронный ресурс] // IEEE Access. — 2020. — Т. 8. — С. 207672-207682. — Режим доступа: https://doi.org/10.1109/access.2020.3038184.

17. Arefi A., Motlagh A.M., Mollazade K., Teimourlou R.F. Recognition and localization of ripen tomato based on machine vision Australian // Journal of Crop Science. — 2011. — Т. 10, № 5. — С. 1144-1149.

18. Zhao Y., Gong L., Huang Y., Liu C. Robust Tomato Recognition for Robotic Harvesting Using Feature Images Fusion [Электронный ресурс] // Sensors. — 2016. — Т. 16, № 2. — С. 173. — Режим доступа: https://doi.org/10.3390/s16020173.

19. Ko K., Yang S.H., Jang I. Real-Time Tomato Ripeness Classification System based on Deep Learning Model for Object Detection [Электронный ресурс] // Journal of Institute of Control, Robotics and Systems. — 2018. — Т. 24, № 11. — С. 999-1004. — Режим доступа: https://doi.org/10.5302/j.icros.2018.18.0166.

20. Ko K., Jang I., Choi J.H., Lim J.H., Lee D.U. Stochastic Decision Fusion of Convolutional Neural Networks for Tomato Ripeness Detection in Agricultural Sorting Systems [Электронный ресурс] // Sensors. — 2021. — Т. 21, № 3. — С. 917. — Режим доступа: https://doi.org/10.3390/s21030917.

21. Joglekar A., Joshi D., Khemani R., Nair S., Sahare S. Depth Estimation Using Monocular Camera // International Journal of Computer Science and Information Technologies. — 2011. — Т. 4, № 2.

22. Mussabayev R.R., Kalimoldayev M.N., Amirgaliyev Y.N., Tairova A.T., Mussabayev T.R. Calculation of 3D Coordinates of a Point on the Basis of a Stereoscopic System [Электронный ресурс] // Open Engineering. — 2018. — Т.

8, № 1. — С. 109-117. — Режим доступа: https://doi.org/10.1515/eng-2018-0016.

23. Kuznetsova A., Maleva T., Soloviev V. Detecting Apples in Orchards Using YOLOv3 and YOLOv5 in General and Close-Up Images [Электронный ресурс] // Advances in Neural Networks – ISNN 2020. — 2020. — С. 233-243. — Режим доступа: https://doi.org/10.1007/978-3-030-64221-1_20.

24. Grandillo S., Zamir D., Tanksley S.D. Genetic improvement of processing tomatoes: A 20 years perspective [Электронный ресурс] // Euphytica. — 1999. — Т. 110, № 2. — С. 85-97. — Режим доступа: https://doi.org/10.1023/a:1003760015485.

25. ФАОСТАТ: Продукты животноводства и сельскохозяственных культур // https://www.fao.org/faostat/ru/#data/QCL / [Электронный ресурс]: 3.12.21.

26. Bergasa L.M., Nuevo J., Sotelo M.A., Barea R., Lopez M.E. Real-Time System for Monitoring Driver Vigilance [Электронный ресурс] // IEEE Transactions on Intelligent Transportation Systems. — 2006. — Т. 7, № 1. — С. 63-77. — Режим доступа: https://doi.org/10.1109/tits.2006.869598

27. Brosnan T., Sun D. Improving quality inspection of food products by computer vision—a review [Электронный ресурс] // Journal of Food Engineering. — 2004. — Т. 61, № 1. — С. 3-16. — Режим доступа: https://doi.org/10.1016/s0260-8774(03)00183-3.

28. Buch N., Velastin S. A., Orwell J. A Review of Computer Vision Techniques for the Analysis of Urban Traffic [Электронный ресурс] // IEEE Transactions on Intelligent Transportation Systems. — 2011. — Т. 12, № 3. — С. 920-939. — Режим доступа: https://doi.org/10.1109/tits.2011.2119372.

29. Maglogiannis I., Doukas C. N. Overview of Advanced Computer Vision Systems for Skin Lesions Characterization [Электронный ресурс] // IEEE Transactions on Information Technology in Biomedicine. — 2009. — Т. 13, № 5. — С. 721-733. — Режим доступа: https://doi.org/10.1109/titb.2009.2017529.

30. Da Silva A.B., Mendonca G.V. Digital Image Processing [Электронный ресурс] // The Electrical Engineering Handbook. — 2005. — С. 891-910. — Режим доступа: https://doi.org/10.1016/b978-012170960-0/50064-5.

31. Malik M. H., Zhang T., Li H., Zhang M., Shabbir S., Saeed A. Mature Tomato Fruit Detection Algorithm Based on improved HSV and Watershed Algorithm [Электронный ресурс] // IFAC-PapersOnLine. — 2018. — Т. 51, № 17. — С. 431-436. — Режим доступа: https://doi.org/10.1016/j.ifacol.2018.08.183.

32. Бурибаев Ж.А., Амиргалиева Ж., Джолдасбаев С.К., Жасұзақ М.С., Турегали А.С., Даулетия Д. Tomato maturity recognition using YOLOv5 machine learning // Вестник Национальной инженерной академии Республики Казахстан. — 2021. — Т. 82, № 4. — С. 49-60.

33. Basu M. Gaussian-based edge-detection methods-a survey [Электронный ресурс] // IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews). — 2002. — Т. 32, № 3. — С. 252-260. — ISSN 10946977. — Режим доступа: https://doi.org/10.1109/tsmcc.2002.804448.

34. D'Haeyer P. F. Gaussian filtering of images: A regularization approach [Электронный ресурс] // Signal Processing. — 1989. — Т. 18, № 2. — С. 169-181. — Режим доступа: https://doi.org/10.1016/0165-1684(89)90048-0.

35. Deng G., Cahill L. W. View more Adaptive Gaussian filter for noise reduction and edge detection // IEEE Nuclear Science Symposium & Medical Imaging Conference : Proceedings of the 1993 IEEE Nuclear Science Symposium & Medical Imaging Conference, 1993, 30 окт.-6 ноября 1993. — 1994. — С. 1615 - 1619.

36. Piao W., Yuan Y., Lin H. A Digital Image Denoising Algorithm Based on Gaussian Filtering and Bilateral Filtering [Электронный ресурс] / W. Piao, Y. Yuan, H. Lin // ITM Web of Conferences. — 2018. — Т. 17. — 01006. — Режим доступа: https://doi.org/10.1051/itmconf/20181701006.

37. Gedraite E. S., Hadad M. Investigation on the effect of a Gaussian Blur in image filtering and segmentation // Proceedings Elmar - International Symposium Electronics in Marine : 53rd International Symposium ELMAR-2011, Задар, 14-16 сентября 2011. — 2011. — С. 393–396.

38. Tyagi T., Vishal M. 2D Gaussian Filter for Image Processing: A Study // International Journal for Science Technology and Engineering. — 2017. — Т. 3. — С. 22-24.

39. Villar S.A., Torcida S., Acosta G.G. Median Filtering: A New Insight [Электронный ресурс] // Journal of Mathematical Imaging and Vision. — 2017. — Т. 58, № 1. — С. 130-146. — Режим доступа: https://doi.org/10.1007/s10851-016-0694-0.

40. Zhu Y., Huang C. An Improved Median Filtering Algorithm for Image Noise Reduction [Электронный ресурс] // Physics Procedia. — 2012. — Т. 25. — С. 609-616. — Режим доступа: https://doi.org/10.1016/j.phpro.2012.03.133.

41. Hwang H., Haddad R.A. Adaptive Median Filters: New Algorithms and Results // IEEE Transactions on Image Processing : IEEE, апрель 1995. — 1995. — Режим доступа: https://doi.org/10.1109/83.370679.

42. Dhakshina Kumar S., Esakkirajan S., Bama S. A non destructive tomato grading and sorting mechanism using binary SVM pair based classifier for sustainable agricultural practices // Journal of Green Engineering. — 2020. — Т. 10, № 7. — С. 3637-3654.

43. Li S., Chen S., Liu B., Li Y., Liang Y. Decentralized kinematic control of a class of collaborative redundant manipulators via recurrent neural networks [Электронный ресурс] // Neurocomputing. — 2012. — Т. 91. — С. 1-10. — Режим доступа: https://doi.org/10.1016/j.neucom.2012.01.034.

44. Li Z., Chen W., Luo J. Adaptive compliant force–motion control of coordinated non-holonomic mobile manipulators interacting with unknown non-rigid environments [Электронный ресурс] // Neurocomputing. — 2008. — Т. 71, № 7-9. — С. 1330-1344. — Режим доступа: https://doi.org/10.1016/j.neucom.2007.06.001.

45. Le T., Kang H. An adaptive tracking control for parallel robot manipulators based on fully tuned radial basic function networks // Neurocomputing. — 2014. — Т. 137. — С. 12–23.

46. Giorelli M. Renda F. Ferri G. Laschi C. A feed-forward neural network learning the inverse kinetics of a soft cable-driven manipulator moving in three-dimensional space / M. Giorelli [и др.] // IEEE International Conference on Intelligent Robots and Systems : 2013 26th IEEE/RSJ International Conference on Intelligent Robots and Systems: New Horizon, IROS 2013, Токио, 3-8 ноября 2013. — 2013. — С. 5033–5039

47. Xiao L., Zhang Y. A New Performance Index for the Repetitive Motion of Mobile Manipulators [Электронный ресурс] // IEEE Transactions on Cybernetics. — 2014. — Т. 44, № 2. — С. 280-292. — Режим доступа: https://doi.org/10.1109/tcyb.2013.2253461.

48. Yeo S.H., Yang G., Lim W.B. Design and analysis of cable-driven manipulators with variable stiffness [Электронный ресурс] // Mechanism and Machine Theory. — 2013. — Т. 69. — С. 230-244. — Режим доступа: https://doi.org/10.1016/j.mechmachtheory.2013.06.005.

49. Yang C., Luo J., Pan Y., Liu Z., Su C.Y. Personalized Variable Gain Control With Tremor Attenuation for Robot Teleoperation [Электронный ресурс] // IEEE Transactions on Systems, Man, and Cybernetics: Systems. — 2018. — Т. 48, № 10. — С. 1759-1770. — Режим доступа: https://doi.org/10.1109/tsmc.2017.2694020.

50. Belgiu M., Drăguţ L. Random forest in remote sensing: A review of applications and future directions [Электронный ресурс] // ISPRS Journal of Photogrammetry and Remote Sensing. — 2016. — Т. 114. — С. 24-31. — Режим доступа: https://doi.org/10.1016/j.isprsjprs.2016.01.011.

51. Starovoitov V. V., Golub Y. I. Comparative study of quality estimation of binary classification [Электронный ресурс] // Informatics. — 2020. — Т. 17, № 1. — С. 87-101. — Режим доступа: https://doi.org/10.37661/1816-0301-2020-17-1-87-101.

52. Breiman L. Random forests [Электронный ресурс] // Machine Learning. — 2001. — Т. 45, № 1. — С. 5-32. — ISSN 08856125. — Режим доступа: https://doi.org/10.1023/a:1010933404324.

53. Brereton R.G., Lloyd G.R. Support Vector Machines for classification and regression [Электронный ресурс] // The Analyst. — 2010. — Т. 135, № 2. — С. 230-267. — Режим доступа: https://doi.org/10.1039/b918972f.

54. Chen T. XGBoost: A scalable tree boosting system [Электронный ресурс] // Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining : KDD 2016, Сан-Франциско, 13-17 августа 2016. — 2016. — Режим доступа: https://doi.org/10.1145/2939672.2939785.

55. Jin L., Li, S., Hu B., Liu M. A survey on projection neural networks and their applications [Электронный ресурс] // Applied Soft Computing. — 2019. — Т. 76. — С. 533-544. — Режим доступа: https://doi.org/10.1016/j.asoc.2019.01.002.

56. Hastings G.G., Book W.J. A linear dynamic model for flexible robotic manipulators [Электронный ресурс] // IEEE Control Systems Magazine. — 1987. — Т. 7, № 1. — С. 61-64. — Режим доступа: https://doi.org/10.1109/mcs.1987.1105233.

57. Koivo A.J., Guo T.H. Adaptive linear controller for robotic manipulators [Електронний ресурс] // IEEE Transactions on Automatic Control. — 1983. — Т. 28, № 2. — С. 162-171. — Режим доступу: https://doi.org/10.1109/tac.1983.1103211.

58. Golla D.F., Garg S.C., Hughes P.C. Linear state-feedback control of manipulators [Электронный ресурс] // Mechanism and Machine Theory. — 1981. — Т. 16, № 2. — С. 93-103. — Режим доступа: https://doi.org/10.1016/0094-114x(81)90055-0.

59. Wang Q., Qi F., Sun M., Qu J., Xue J. Identification of Tomato Disease Types and Detection of Infected Areas Based on Deep Convolutional Neural Networks and Object Detection Techniques [Электронный ресурс] // Computational Intelligence and Neuroscience. — 2019. — Т. 2019. — С. 1-15. — Режим доступа: https://doi.org/10.1155/2019/9142753.

60. Wu Y., X, L. Crop Organ Segmentation and Disease Identification Based on Weakly Supervised Deep Neural Network [Электронный ресурс // Agronomy. — 2019. — Т. 9, № 11. — С. 737. — Режим доступа: https://doi.org/10.3390/agronomy9110737.

61. Afonso M., Fonteijn H., Fiorentin F.S., Lensink D., Mooij M., Faber N., Polder G., Wehrens R. Tomato Fruit Detection and Counting in Greenhouses Using Deep Learning [Электронный ресурс] // Frontiers in Plant Science. — 2020. — Т. 11. — Режим доступа: https://doi.org/10.3389/fpls.2020.571299.

62. Gomes J.F.S., Leta F.R. Applications of computer vision techniques in the agriculture and food industry: a review [Электронный ресурс] // European Food Research and Technology. — 2012. — Т. 235, № 6. — С. 989-1000. — Режим доступа: https://doi.org/10.1007/s00217-012-1844-2.

63. Uijlings J.R.R., Van De Sande K.E.A., Gevers T., Smeulders A.W.M. Selective Search for Object Recognition [Электронный ресурс] // International Journal of Computer Vision. — 2013. — Т. 104, № 2. — С. 154-171. — Режим доступа: https://doi.org/10.1007/s11263-013-0620-5.

64. Girshick R., Donahue J., Darrell T., Malik J. Region-Based Convolutional Networks for Accurate Object Detection and Segmentation [Электронный ресурс] // IEEE Transactions on Pattern Analysis and Machine Intelligence. — 2016. — Т. 38, № 1. — С. 142-158. — Режим доступа: https://doi.org/10.1109/tpami.2015.2437384.

65. Girshick R. Fast r-cnn // Proceedings of the IEEE International Conference on Computer Vision : 15th IEEE International Conference on Computer Vision, ICCV 2015, Сантьяго, 11-18 декабря 2015. — 2015. — С. 1440–1448

66. Ren S., He K., Girshick R., Sun J. Faster R-CNN: Towards real-time object detection with region proposal networks // Advances in Neural Information

Processing Systems : 29th Annual Conference on Neural Information Processing Systems, NIPS 2015, Монтреал, 7-12 декабря 2015. — 2015. — С. 91–99

67. Redmon J., Divvala S., Girshick R., Farhadi A. You only look once: unified, real-time object detection // Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition : 29th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Лас-Вегас, 26 июнь-1 июля 2016. — 2016.

68. Urmashev B., Buribayev Z., Amirgaliyeva Z., Ataniyazova A., Zhassuzak M., Turegali A. Development of a weed detection system using machine learning and neural network algorithms [Электронный ресурс] // Eastern-European Journal of Enterprise Technologies. — 2021. — Т. 6, № 2 (114). — Режим доступа: https://doi.org/10.15587/1729-4061.2021.246706.

69. He K., Gkioxari G., Dollar P., Girshick R. Mask R-CNN // Proceedings of the IEEE International Conference on Computer Vision : 16th IEEE International Conference on Computer Vision, ICCV 2017Venice, Венеция, 22-29 октября 2017. — 2017. — С. 2980 – 2988.

70. Wang Q., Wu B., Zhu P., Li P., Zuo W., Hu Q. ECA-Net: Efficient Channel Attention for Deep Convolutional Neural Networks [Электронный ресурс] // 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). — 2020. — Режим доступа: https://doi.org/10.1109/cvpr42600.2020.01155.

71. Marengoni M., Stringhini D. High level computer vision using OpenCV // Proceedings - 24th SIBGRAPI Conference on Graphics, Patterns, and Images Tutorials: SIBGRAPI - 2011, Масейо, 28-31 августа 2011. — 2011.

72. Yeshmukhametov A., Buribayev Z., Amirgaliyev Y., Amirgaliyev B., Latuta K. Bio-inspired a novel continuum robot arm with variable backbone design: Modelling and validation / A. Yeshmukhametov [и др.] // Journal of Theoretical and Applied Information Technology. — 2019. — Т. 97, № 19. — С. 5036 - 5047.

73. Yeleussinov A., Islamgozhayev T., Satymbekov M., Kozhagul A. CVCER: Robot to Learn Basics of Computer Vision and Cryptography // IOP Conference Series: Materials Science and Engineering : 5th International Conference on Mechanics and Mechatronics Research, ICMMR 2018, Токио, 19-21 июля 2018. — 2018. — Режим доступа: https://doi.org/10.1088/1757-899X/417/1/012013.

74. Pethick M., Liddle M., Werstein P., Huang Z. Parallelization of a Backpropagation Neural Network on a Cluster Computer // Proceedings of the IASTED International Conference on Parallel and Distributed Computing and Systems : Conference Proceedings, Марина дель Рей, 3-5 ноября 2003. — 2003.

75. The World Bank , Global consumption database for 2019, fresh or chilled vegetables section, 2019

76. Zhao Y., Gong L., Liu C., Huang Y. Dual-arm Robot Design and Testing for Harvesting Tomato in Greenhouse [Электронный ресурс] // IFAC-PapersOnLine. — 2016. — Т. 49, № 16. — С. 161-165. — Режим доступа: https://doi.org/10.1016/j.ifacol.2016.10.030.

77. Ling X., Zhao Y., Gong L., Liu C., and Wang T. Dual-arm cooperation and implementing for robotic harvesting tomato using binocular vision [Электронный ресурс] // Robotics and Autonomous Systems. — 2019. — Т. 114. — С. 134-143. — Режим доступа: https://doi.org/10.1016/j.robot.2019.01.019.

78. Feng Q., Wang X., Wang G., Li Z. Design and test of tomatoes harvesting robot / Q. Feng [и др.] // 2015 IEEE International Conference on Information and Automation, : ICIA 2015 - In conjunction with 2015 IEEE International Conference on Automation and Logistics, Лицзян, 8-10 августа 2015. — 2015. — Режим доступа: https://10.1109/ICInfA.2015.7279423.

79. Tokunaga T., Oka K., Harada A. 1segment continuum manipulator for automatic harvesting robot - Prototype and modeling // 2017 IEEE International Conference on Mechatronics and Automation, ICMA 2017 : 14th, Такамацу, 6-9 августа 2017. — 2017.

80. Hayashi S., Shigematsu K., Yamamoto S., Kobayashi K., Kohno Y., Kamata J., Kurita M. Evaluation of a strawberry-harvesting robot in a field test [Электронный ресурс] // Biosystems Engineering. — 2010. — Т. 105, № 2. — С. 160-171. — Режим доступа: https://doi.org/10.1016/j.biosystemseng.2009.09.011.

81. Hiroaki Y, Kotaro N, Takaomi H, and Masayuki I. Development of An Autonomous Tomato Harvesting Robot with Rotational Plucking Gripper // IEEE International Conference on Intelligent Robots and Systems : IROS 2016, Тэджон, 9-14 октября 2016. — 2016. — Режим доступа: https://doi.org/10.1109/IROS.2016.7759122.

82. Root AI // https://root-ai.com/#intro / [Электронный ресурс]: 15.08.19.

83. Panasonic company. Introducing AI-equipped Tomato Harvesting Robots to Farms May Help to Create Jobs // https://news.panasonic.com/global/stories/2018/57801.html / [Электронный ресурс]: 29.04.18.

84. Yeshmukhametov A., Koganezawa K., Yamamoto Y. Design and Kinematics of Cable-Driven Continuum Robot Arm with Universal Joint Backbone // 2018 IEEE International Conference on Robotics and Biomimetics : ROBIO 2018, Куала-Лумпур, 12-15 декабря 2018. — 2018.

85. Yeshmukhametov A., Koganezawa K., Yamamoto Y. A Novel Discrete Wire-Driven Continuum Robot Arm with Passive Sliding Disc: Design, Kinematics and Passive Tension Control [Электронный ресурс] // Robotics. — 2019. — Т. 8, № 3. — С. 51. — Режим доступа: https://doi.org/10.3390/robotics8030051.

86. Yeshmukhametov A., Koganezawa K., Buribayev Z., Amirgaliyev Y., Yamamoto, Y. Development of Continuum Robot Arm and Gripper for Harvesting Cherry Tomatoes [Электронный ресурс] // Preprint. — 2019. — Режим доступа: https://doi.org/10.20944/preprints201912.0237.v1.

ПРИЛОЖЕНИЕ А

Код Yolo на python

```
from yolo_video1 import yolo
import numpy as np
import copy

def YoLo(frame):
    f = frame.copy()
    yolo_frame, boxes, labels = yolo(frame)
    array = np.array(boxes)
    if len(boxes):
        box = np.zeros((len(boxes), 4))
        for i in range(len(boxes)):
            box[i, 0] = array[i, 1]
            box[i, 1] = array[i, 1] + array[i, 3]
            box[i, 2] = array[i, 0]
            box[i, 3] = array[i, 0] + array[i, 2]
    else:
        box = np.array([])
    box = box.astype('int32')
    if len(labels) == box.shape[0]:
        return yolo_frame, labels, box
    else:
        box = np.array([])
        labels = []
        return f, labels, box
```

ПРИЛОЖЕНИЕ Б

Код линейного метода на python

```
import psutil
import os
import numpy as np
import imutils
import time
from functions.yolo import YoLo
from functions.clahe import Clahe
import cv2
import glob
import copy

p = psutil.Process()
processes = range(2)
p.cpu_affinity(processes)

cv_file = cv2.FileStorage()
cv_file.open('/home/robo/Jupyter_notebook/chessboard/stereoMap.xml',
cv2.FileStorage_READ)
stereoMapL_x = cv_file.getNode('stereoMapL_x').mat()
stereoMapL_y = cv_file.getNode('stereoMapL_y').mat()
stereoMapR_x = cv_file.getNode('stereoMapR_x').mat()
stereoMapR_y = cv_file.getNode('stereoMapR_y').mat()

capL = cv2.VideoCapture(0)
capR = cv2.VideoCapture(2)

T = 15
f = 812
n = 20

Q = np.array([[1, 0, 0, -288],
              [0, 1, 0, -238],
              [0, 0, 0, 812],
              [0, 0, -1/15, 0]])

try:

    while True:

        retL, frame_left = capL.read()
```

```
        retR, frame_right = capR.read()

        frame_left = Clahe(frame_left)
        frame_right = Clahe(frame_right)

        frame_left = cv2.remap(frame_left, stereoMapL_x, stereoMapL_y,
cv2.INTER_LANCZOS4, cv2.BORDER_CONSTANT, 0)
        frame_right = cv2.remap(frame_right, stereoMapR_x, stereoMapR_y,
cv2.INTER_LANCZOS4, cv2.BORDER_CONSTANT, 0)

        l = frame_left.copy()
        r = frame_right.copy()

        frame_left, List1, array1 = YoLo(frame_left)
        frame_right, List2, array2 = YoLo(frame_right)

        grayL = cv2.cvtColor(l, cv2.COLOR_BGR2GRAY)
        grayR = cv2.cvtColor(r, cv2.COLOR_BGR2GRAY)

        points = [[0, 0, 0, 0]] * len(List1)
        S = 0

        for i in range(len(List1)):
            Max = np.ones((2*n+1, 2*n+1)) * 255
            S = (2*n+1)**2

            x1 = (array1[i, 0] + array1[i, 1])//2
            y1 = (array1[i, 2] + array1[i, 3])//2
            for j in range(len(List2)):
                x2 = (array2[j, 0] + array2[j, 1])//2
                y2 = (array2[j, 2] + array2[j, 3])//2
                if (x1 - 100 < x2 and x2 < x1 + 100):
                    if List1[i] == List2[j]:
                        if array2[j, 2] >= n and array2[j, 3] <= grayL.shape[1]-n-1:
                            for k in range(array2[j, 2], array2[j, 3]):
                                M = abs(grayL[x1-n:x1+n+1, y1-n:y1+n+1]-
                                        grayR[x2-n:x2+n+1, k-n:k+n+1])
                                count = 0
                                for w in range(0, M.shape[0]):
                                    for h in range(0, M.shape[1]):
                                        if M[w, h] < Max[w, h]:
                                            count = count + 1
                                if S > count:
                                    S = count
```

```
                        Max = M.copy()
                        points[i][0] = x1
                        points[i][1] = y1
                        points[i][2] = x2
                        points[i][3] = k

        for i in range(len(points)):
            if points[0][0] != 0:
                frame_left[points[i][0]-3:points[i][0]+4, points[i][1]-3:points[i][1]+4] =
[0, 0, 255]
                frame_right[points[i][2]-3:points[i][2]+4, points[i][3]-3:points[i][3]+4] =
[0, 0, 255]
                images = np.hstack((frame_left, frame_right))
                print('distance =', (f * T / abs(points[i][3] - points[i][1])))

        for i in range(len(points)):
            if points[0][0] != 0:
                A = np.array([points[i][0], points[i][1], points[i][3] - points[i][1], 1])
                W = Q.dot(A)
                print('x =', W[0]/W[3])
                print('y =', W[1]/W[3])
                print('z =', W[2]/W[3])
        images = np.hstack((frame_left, frame_right))
        cv2.imshow('image', images)
        if cv2.waitKey(1) & 0xFF == ord('q') or not retL:
            break
    capL.release()
    capR.release()
    cv2.destroyAllWindows()
finally:
    capL.release()
    capR.release()
```

ПРИЛОЖЕНИЕ В

Код параллельного метода на python

```
import psutil
p = psutil.Process()
processes = range(6)
p.cpu_affinity(processes)
import imutils
import time
from functions.yolo import YoLo
from functions.clahe import Clahe
import cv2
import numpy as np
import multiprocessing as mp
from multiprocessing import Queue, Process
import os

def L_camera(coordinates_left, disparity_left, label_left):
    cv_file = cv2.FileStorage()
    cv_file.open('/home/robo/disser/new_photo_chess/stereoMap.xml',
cv2.FileStorage_READ)
    stereoMapL_x = cv_file.getNode('stereoMapL_x').mat()
    stereoMapL_y = cv_file.getNode('stereoMapL_y').mat()
    stereoMapR_x = cv_file.getNode('stereoMapR_x').mat()
    stereoMapR_y = cv_file.getNode('stereoMapR_y').mat()

    capL = cv2.VideoCapture(0)
    while True:
        ret_left, frame_left = capL.read()
        frame_left = cv2.remap(frame_left, stereoMapL_x, stereoMapL_y,
cv2.INTER_LANCZOS4, cv2.BORDER_CONSTANT, 0)

        l = frame_left.copy()
        frame_left, List, array = YoLo(frame_left)
        if label_left.qsize() < 1:
            disparity_left.put(l)
            coordinates_left.put(array)
            label_left.put(List)
        cv2.imshow('YoLo_left', frame_left)
        key = cv2.waitKey(1) & 0xFF
        if key == ord("q") or not ret_left:
            break
    capL.release()
```

```
    cv2.destroyAllWindows()

def R_camera(coordinates_right, disparity_right, label_right, label_left):
    cv_file = cv2.FileStorage()
    cv_file.open('/home/robo/disser/new_photo_chess/stereoMap.xml',
cv2.FileStorage_READ)
    stereoMapL_x = cv_file.getNode('stereoMapL_x').mat()
    stereoMapL_y = cv_file.getNode('stereoMapL_y').mat()
    stereoMapR_x = cv_file.getNode('stereoMapR_x').mat()
    stereoMapR_y = cv_file.getNode('stereoMapR_y').mat()

    capR = cv2.VideoCapture(2)
    while True:
        ret_right, frame_right = capR.read()
        frame_right = cv2.remap(frame_right, stereoMapR_x, stereoMapR_y,
cv2.INTER_LANCZOS4, cv2.BORDER_CONSTANT, 0)
        r = frame_right.copy()
        frame_left, List, array = YoLo(frame_right)
        if label_left.qsize() < 1:
            disparity_right.put(r)
            coordinates_right.put(array)
            label_right.put(List)
        cv2.imshow('YoLo_right', frame_right)
        key = cv2.waitKey(1) & 0xFF
        if key == ord("q") or not ret_right:
            break
    capR.release()
    cv2.destroyAllWindows()

def distance(disparity_left, disparity_right, coordinates_left, coordinates_right,
label_left, label_right):

    T = 15
    f = 812
    n = 20

    Q = np.array([[1, 0, 0, -288],
                  [0, 1, 0, -238],
                  [0, 0, 0, 812],
                  [0, 0, -1/15, 0]])

    try:
        while True:
```

```
frame_left = disparity_left.get()
frame_right = disparity_right.get()

frame_left = Clahe(frame_left)
frame_right = Clahe(frame_right)

array1 = coordinates_left.get()
array2 = coordinates_right.get()

List1 = label_left.get()
List2 = label_right.get()

grayL = cv2.cvtColor(frame_left, cv2.COLOR_BGR2GRAY)
grayR = cv2.cvtColor(frame_right, cv2.COLOR_BGR2GRAY)

points = [[0, 0, 0, 0]] * len(List1)
S = 0

for i in range(len(List1)):
    Max = np.ones((2*n+1, 2*n+1)) * 255
    S = (2*n+1)**2

    x1 = (array1[i, 0] + array1[i, 1])//2
    y1 = (array1[i, 2] + array1[i, 3])//2
    for j in range(len(List2)):
        x2 = (array2[j, 0] + array2[j, 1])//2
        y2 = (array2[j, 2] + array2[j, 3])//2
        if (x1 - 100 < x2 and x2 < x1 + 100):
            if List1[i] == List2[j]:
                if array2[j, 2] >= n and array2[j, 3] <= grayL.shape[1]-n-1:
                    for k in range(array2[j, 2], array2[j, 3]):
                        M = abs(grayL[x1-n:x1+n+1, y1-n:y1+n+1]-
                                grayR[x2-n:x2+n+1, k-n:k+n+1])
                        count = 0
                        for w in range(0, M.shape[0]):
                            for h in range(0, M.shape[1]):
                                if M[w, h] < Max[w, h]:
                                    count = count + 1
                        if S > count:
                            S = count
                            Max = M.copy()
                            points[i][0] = x1
                            points[i][1] = y1
                            points[i][2] = x2
```

```
                                points[i][3] = k

            for i in range(len(points)):
                if points[0][0] != 0:
                    frame_left[points[i][0]-3:points[i][0]+4, points[i][1]-3:points[i][1]+4]
= [0, 0, 255]
                    frame_right[points[i][2]-3:points[i][2]+4, points[i][3]-
3:points[i][3]+4] = [0, 0, 255]
                    images = np.hstack((frame_left, frame_right))
                    print('distance =', (f * T / abs(points[i][3] - points[i][1])))

            for i in range(len(points)):
                if points[0][0] != 0:
                    A = np.array([points[i][0], points[i][1], points[i][3] - points[i][1], 1])
                    W = Q.dot(A)
                    print('x =', W[0]/W[3])
                    print('y =', W[1]/W[3])
                    print('z =', W[2]/W[3])

            images = np.hstack((frame_left, frame_right))
            cv2.imshow('image', images)
            if cv2.waitKey(1) & 0xFF == ord('q'):
                break
        cv2.destroyAllWindows()
    finally:
        cv2.destroyAllWindows()

if __name__ == '__main__':

    disparity_left = Queue()
    disparity_right = Queue()
    coordinates_left = Queue()
    coordinates_right = Queue()
    label_left = Queue()
    label_right = Queue()

    process_0 = Process(target=L_camera, args=(coordinates_left, disparity_left,
label_left))
    process_1 = Process(target=R_camera, args=(coordinates_right, disparity_right,
label_right, label_left))
    process_2 = Process(target=distance, args=(disparity_left, disparity_right,
                                        coordinates_left, coordinates_right, label_left,
label_right))
```

```
process_0.start()
process_1.start()
process_2.start()

disparity_left.close()
disparity_right.close()
coordinates_left.close()
coordinates_right.close()
label_left.close()
label_right.close()

disparity_left.join_thread()
disparity_right.join_thread()
coordinates_left.join_thread()
coordinates_right.join_thread()
label_left.join_thread()
label_right.join_thread()

process_0.join()
process_1.join()
process_2.join()
```