

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ	9
---------------------------	----------

Глава 1. Исследование управляющих систем и обоснование необходимости переосмыслиния парадигмы управления	14
1.1. Эпохи развития промышленности	14
1.2. Анализ проблем существующих интеллектуальных систем . .	17
1.3. Обзор исследовательских работ в области открытых систем	20
1.4. Агентный подход к проектированию систем	22
1.5. Актуальность исследовательской работы и обоснование её проведения	24
1.6. Выводы к первой главе	25
Глава 2. Сравнение и описание существующих алгоритмов определения и классификации объектов на изображениях	27
2.1. Теория распознавания образов	27
2.2. Задача классификации объектов	32
2.3. Обзор существующих алгоритмов и методик обработки изображений	36
2.3.1. Алгоритмы фильтрации изображений	38
2.3.2. Алгоритмы логической обработки изображений . .	42
2.3.3. Алгоритмы обучения	46
2.3.4. Критерии выбора методик обработки	47
2.4. Описание применяемого алгоритма работы	48
2.5. Выводы ко второй главе	50
Глава 3. Состав и техническое описание разрабатываемого компонента машинного зрения	51

3.1. Описание архитектуры разрабатываемой двухкоординатной платформы	51
3.2. Модуль машинного зрения	54
3.2.1. Структура компонентов модуля	54
3.2.2. Состав технических средств	56
3.2.3. Состав программных средств	57
3.3. Практические результаты	58
3.4. Выводы к третьей главе	60
ЗАКЛЮЧЕНИЕ	61
ПЕРЕЧЕНЬ РИСУНКОВ	63
ПЕРЕЧЕНЬ ТАБЛИЦ	64
Список используемых сокращений	65
Список литературы	66
Приложение А. Текст программы определения метки . . .	70
Приложение Б. Общий вид координатной платформы . . .	74
Приложение В. Применяемые технические средства	75

ВВЕДЕНИЕ

Машинное зрение уже довольно давно успешно внедряется в огромном спектре отраслей, среди которых существенную долю практического применения занимают задачи промышленного производства. С развитием информационных технологий существует необходимость делегирования от человека к техническим средствам и алгоритмам разного рода операции, требующие непрерывного контроля и высокой точности. Системы технического зрения или *машинаное зрение* в общем виде – это комплекс технических и программных средств, созданный с целью интерпретирования, классификации и определения объектов реального мира и использование этого знания для решения конкретной задачи. К этому же определению можно отнести совокупность методических средств и теоретических документов, описывающих данное направление с точки зрения функционирования и применяемости.

Основным вектором применения машинного зрения можно назвать такие отрасли, как робототехника и производство, хотя схожие технологии можно встретить и в повседневной жизни, например, при сканировании штрих-кода в супермаркете или так называемые QR-коды. Суть работы такого оборудования в большинстве случаев неизменна – существует реальный объект, из которого необходимо получить некую информацию и соответствующим образом её обработать. В штрих-коде такой информацией будет страна и код изготовителя товара, в QR-коде содержится ссылка на какую-либо страницу в Интернете или же простое текстовое сообщение.

Современная ориентация на кибер-физическное производство требует при проектировании нового поколения технологического оборудования усиленного взаимодействия информационных технологий с физическими инструментами. Такая интеграция двух разных областей открывает новые возможности в организации цифрового производства, повышает его гиб-

кость, масштабируемость и простоту использования.

Но до сих пор основным способом проектирования промышленных систем является монолитная архитектура. Такие системы являются «вешью в себе», закрытой для посторонних сущностью, что, с одной стороны, имеет ряд положительных характеристик. Закрытая жёсткая логика таких систем является своеобразной защитой от несанкционированных действий со стороны пользователя (так называемая «защита от дурака»), уменьшает риск ошибок и отказов в процессе работы. Однако, они находят всё меньшее применения в индустрии нового поколения, где недостатки монолитной архитектуры начинают перевешивать описанные выше достоинства. Такие системы закрыты для внедрения новых компонентов оборудования (либо их нужно долго настраивать) и усложняют перестроение производства под новые техпроцессы изделий. В сочетании с высокой ценой таких систем любое изменение в производственном процессе грозит стать неоправданно дорогим. Это было хорошо проиллюстрировано на предприятиях России в 1990-е годы, когда стали доступны новые технологии производства, в частности, пластмасс и других полимеров. При этом многие заводы всё ещё широко использовали станки и технологии советских времён, поскольку внедрение новых методик и оборудования было слишком затратным: вкладываться в закупку новых станков и материалов, монтаж новых производственных линий и обучение персонала желающих было мало.

В наши дни полным ходом идёт переориентация промышленного производства, набирают популярность небольшие производственные линии, работающим по индивидуальным заказам, так называемые «малые предприятия», что вынуждает применять совершенно другой подход к проектированию интеллектуального оборудования [1]. Такие изменения вынуждают кардинально пересматривать архитектуру оборудования с ЧПУ. Поэтому в последние годы набирает популярность концепции модульных систем, которые лучше подходят для производств нового типа. Такие системы слож-

нее в проектировании, но позже позволяют легко перестраивать их под нужды производства, изменять и улучшать компоненты и встраивать новые, широко использовать преимущества локальных вычислительных сетей (ЛВС) и облачных технологий в ходе работы.

Основной характеристикой модулей в таких архитектурах является *автономность* каждого из них, что при децентрализованной системе управления является достаточно сложной задачей. Под автономностью здесь следует понимать минимально возможный уровень зависимости каждого компонента от всех остальных, ограничивающейся взаимодействием на протокольном уровне при помощи сообщений. Компоненту нужно знать о «соседях» лишь их статус; функционал и данные на входе и выходе должны оставаться заботой только самого компонента.

Компонент технического зрения, являясь частью оборудования, также может существовать в виде отдельного автономного модуля, который можно перенастраивать и конфигурировать. **Объектом исследований** в данной работе является компонент машинного зрения, встраиваемый в систему управления технологическим оборудованием.

Цель данной работы состоит в обосновании применения модульного подхода при организации архитектуры компонента машинного зрения и проектирования структуры его технических и программных средств. Для достижения цели определён ряд необходимых к рассмотрению основных задач:

- исследование характеристик модульной архитектуры систем управления оборудованием, обзор имеющихся материалов по теме;
- исследование методов обработки изображений, методов классификации объектов и существующих алгоритмов распознавания объектов на входных изображениях для выбора наиболее подходящего по характеристикам используемого контроллера;

- обоснование модульного подхода в организации структуры и работы модуля машинного зрения;
- получение описания работоспособной структуры блока, взаимодействия его составляющих друг с другом для выполнения поставленной задачи и возможных слабых мест в функционировании;
- представление практических результатов и проведение их анализа с целью улучшения и определения направления будущих исследований в данной области.

Многие предприятия начинают переходить к использованию достижений Индустрии 4.0¹, поэтому необходимо понимание её принципов и их доступность широкому кругу лиц, интересующихся данной проблемой. Поскольку модульные системы отвечают принципам Индустрии 4.0, **практическая ценность** работы заключается в необходимости создания теоретической базы для их проектирования. Данная работа является лишь частью разрабатываемой большой платформы, состоящей из комплекса разнообразных технических средств. Практическая реализация описываемой системы может стать отправной точкой для дальнейших исследований, улучшения характеристик архитектуры и методов удешевления её составляющих.

Структура выпускной квалификационной работы. Выпускная квалификационная работа содержит введение, три главы, посвященные задачам данной работы, заключение, библиографию и 3 приложения.

В первой главе рассматривается проблематика в области существующих систем управления оборудованием, обосновывается применение модульного подхода, исследуются возможности использования архитектуры,

¹ Индустрия 4.0 – четвёртая промышленная революция, направленная на тесное взаимодействие физического оборудования с новейшими информационными технологиями, предложена в 2011 году объединением бизнесменов Германии как средство повышения конкурентоспособности [2].

приводится обзор литературы по этой теме и содержатся выводы, поясняющие необходимость исследований в этой области.

Вторая глава посвящена проработке теоретической основы для организации архитектуры системы, обзору методов обработки изображений и сравнительному анализу. Описывается методика реализации машинного зрения и направления его применения в работе оборудования.

Третья глава содержит практические результаты: описание используемых технических и программных средств, структуру компонента машинного зрения и описание его работы, примеры работы алгоритма.

В заключении приводится общая характеристика проделанной работы, выводы по ней и направления дальнейшей деятельности по этой теме, а также возможные варианты улучшения и расширения функционала.

Глава 1

Исследование управляющих систем и обоснование необходимости переосмысления парадигмы управления

1.1. Эпохи развития промышленности

Прежде, чем говорить об Индустрии 4.0, нужно заключить, что имеющая место в наши дни промышленная революция является *четвёртой*. Это значит, что уже как минимум три раза человечество полностью перекраивало само понятие промышленной деятельности. Из исторических источников известно, что первая промышленная революция случилась на рубеже XVIII и XIX веков в Великобритании, Франции и США. Связана она была, в первую очередь, с повсеместным внедрением станков и паровых двигателей, а предпосылками к образованию промышленных предприятий стали развитие торговли, финансовых рынков и экономики. Данный период получил название индустриализации.

Вторая промышленная революция растянулась на вторую половину XIX века и ознаменовалась мощным развитием науки и техники. В ней сохранилась направленность на научные достижения того времени, в частности, электрических приборов или нефтяной энергетики, а развитие путей сообщения существенно облегчило доставку товара до места реализации и эксплуатации, что способствовало повышению покупательского спроса. Внедрение новых источников энергии, более точных станков, новых видов получения конструкционных материалов и других новшеств, безусловно, повысили эффективность производства, что не могло не сказаться на конечной цене товара для потребителя.

Сейчас производство опирается на результаты третьей промышленной революции, началом которой принято считать инновационные изменения, проведённые Генри Фордом¹ во время организации им производства автомобилей. Принято считать, что он является изобретателем конвейера, однако в реальности изменения, проведённые им по части организации производства, реализации и обслуживания товара, куда обширнее. В своих мемуарах он писал: «Не проходит недели, чтобы не появилось какого-нибудь улучшения в машинах или процессах производства, иногда даже противоречащего принятым в стране лучшим производственным методам» [3].

Все изыскания, проводившиеся Фордом в 1913–1915 гг., были направлены на совершенствование производственных линий и максимально возможную экономию сил и ресурсов. Так, большинство рабочих на линии практически не перемещались по территории цеха и выполняли одну или несколько конкретных операций, что позволяло сократить время на обучение сотрудников, а «все действия, которые можно было автоматизировать, автоматизировались». Фактически в это время были заложены основы массового производства.

До сих пор все инновации, внедряемые в промышленность, являлись в той или иной степени развитием идей Форда по автоматизации наиболее возможного числа операций в техпроцессе. Приоритетной задачей большинства научно-исследовательских работ по теме организации производственных процессов была направленность именно на массовое производство, поскольку оно занимало главенствующую роль в экономике каждого развитого государства. Этот приоритет не изменился и с приходом на рынок вычислительных машин, на которые переложили сложные расчётные операции. Удешевление со временем технических средств привело к увеличению доли автоматизации всех областей производства, от проектирования изделия до закупочных мероприятий.

¹ Форд, Генри (Henry Ford) (1863 – 1947) – американский изобретатель и промышленник.

Появление вычислительных средств, первых языков программирования и потребность в автоматизации процессов в 50-х годах XX века привело к созданию станков с ЧПУ, которые продолжают развиваться во многих аспектах по сей день. Как уже было сказано, внедрение подобного оборудования в производственную линию отвечает в первую очередь двум критериям: увеличению темпов массового производства и снижению стоимости операций, проводимых над заготовкой. Использование программируемого оборудования разумно в случаях, когда необходимо выполнять ряд однотипных операций, но если речь идёт о более широкопрофильном производстве, то приходится либо покупать новые станки под другие типы производственных операций, либо перекладывать часть работ на неавтоматизированное оборудование, что может негативно сказаться на темпах выпуска продукции, его качестве и стоимости. Несмотря на то, что развитие станков с ЧПУ идёт в направлении повышения точности и универсальности, это одновременно приводит к удорожанию оборудования и к более строгим требованиям к оператору.

За последние 20 лет появилось большое количество сложных автоматизированных систем, затрагивающих весь жизненный цикл выпускаемого изделия. Зачастую системы существуют в виде программных пакетов, и владельцы промышленных предприятий при приобретении могут выбирать нужный им функционал. Существуют как системы автоматизированного проектирования изделия (например, системы Cimatron или 3D Experience от Dassault Systemes), так и системы управления ресурсами предприятия (ERP), производственными процессами (MES), заказами (SSM), поставками (SCM), агрегированием данных оборудования (SCADA) и документооборотом (PLM). Они являются достаточно гибкими, чтобы удовлетворять потребности больших предприятий, но расчитаны на долгий срок функционирования по устоявшимся производственным процессам [4].

1.2. Анализ проблем существующих интеллектуальных систем

Каким бы совершенным ни было выпускающееся сейчас оборудование с ЧПУ, во избежание рисков, ошибок и просто вследствие коммерческой тайны их архитектура остаётся закрытой. Станки являются сложными и производительными системами с огромным набором инструментальных средств и поэтому хорошо подходят для развитых предприятий, выпускающих большое количество типизированных массовых изделий. Однако для организации единичного или малого производства приобретение такого станка будет являться нерентабельным, его окупаемость может занять долгие годы, а переналадка производственных линий будет сложным и утомительным процессом.

В связи с тем, что за последние годы небольшие производства, работающие по индивидуальным заказам и поэтому вынужденные переналивать производственные линии в соответствии с желанием заказчика, стали появляться всё чаще, то появился уверенный спрос на недорогие и открытые решения в области интеллектуальных систем. Применяемые при массовом производстве системы управления в этих обстоятельствах теряют свою актуальность, поскольку характеризуются следующими свойствами:

- проприетарное исполнение внутренних интерфейсов, их логика зачастую скрыта от потребителя и переналадка таких систем для условий, отличающихся от заданных при производстве, невозможна;
- стандартизованные контроллеры приводов станка, что уменьшает спектр используемого оборудования;
- стандартизованный набор управляющих команд, из-за чего среда написания программ становится менее гибкой и универсальной;

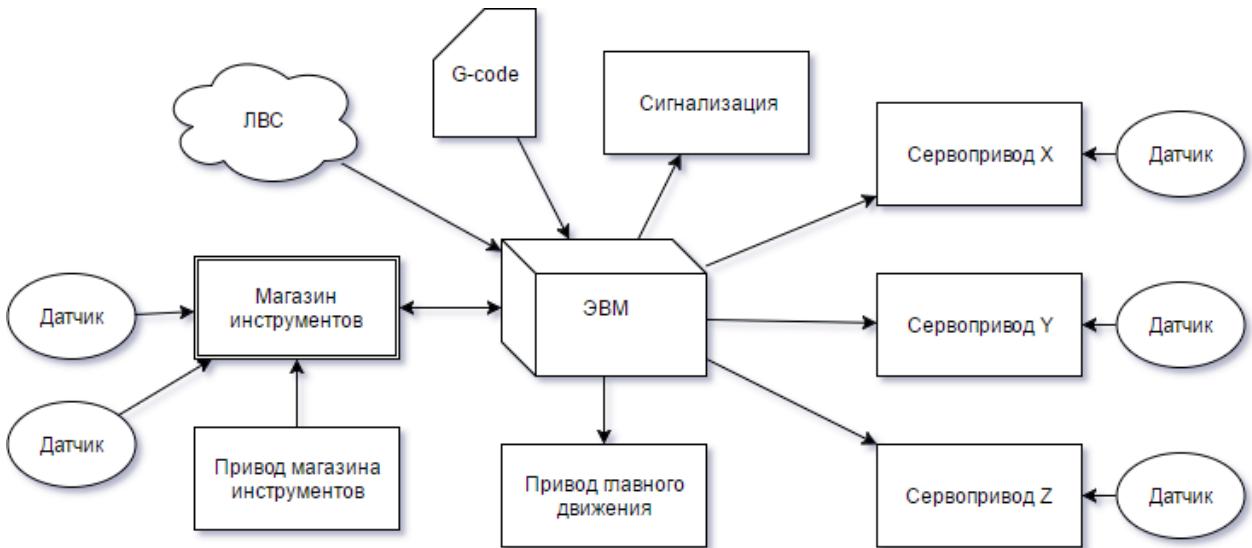


Рисунок 1.1 — Пример централизованной архитектуры станка

- выполнение операций в соответствии с загруженной управляющей программой;
- структура подчинения модулей типа «звезда» (централизованная система управления, как показано на рис. 1.1), при выходе из строя центрального управляющего модуля система становится непригодной для использования.

Обобщая всё перечисленное, можно заметить, что применяемые системы управления ограничены в своей функциональности, поэтому встраивание оборудования в автоматизированную сеть предприятия не имеет преимуществ. Станок будет лишь исполнителем, но никак не интеллектуальной системой. Варианты использования возможностей локальных вычислительных сетей (ЛВС) скучны и заключаются лишь в передаче управляющей программы и формировании отчётов по завершении обработки, полноценный контроль за выполнением управляющих программ отсутствует, а оценить появившуюся критическую ситуацию порой невозможно. Существует ряд мер по повышению степени автоматизации промышленного оборудования, к которым, например, можно отнести:

- принятие решения о смене режима обработки;

- внедрение системы технического зрения;
- отслеживание положения заготовки;
- корректировка положения заготовки и инструмента;
- расширение возможностей управления через web-интерфейс.

Но необходимо понимать, что для внедрения этих мер в существующие системы управления, последние должны обладать другой архитектурой, в которой базовая часть системы управления определена как распределённая вычислительная сеть с диспетчеризацией компонентов. Интеллектуальный агент, как часть распределённой сети, должен представлять собой независимую компьютерную систему, находящуюся в некоторой среде и способную автономно действовать в ней для достижения поставленных целей. Каждое новое оборудование необходимо рассматривать с точки зрения возможности включения его в единую информационно-телеинформацию среду с использование открытого протокола.

Таким образом, резюмируя сказанное выше, можно отметить, что разрабатываемое интеллектуальное оборудование должно отвечать следующему ряду критериев:

1. **Мобильность** – способность сохранять функционал в любых условиях работы, а также возможность быстрого развертывания системы там, где это необходимо;
2. **Гибкость** – важное качество, определяющее способность к быстрому перепрофилированию производственной линии на выпуск новой продукции.
3. **Масштабируемость** – способность внедрять новые компоненты для получения новых функций или поддержки имеющихся.

4. **Интероперабельность** – возможность коммуникации с другими продуктами и устройствами с открытыми интерфейсами без ограничений доступа.
5. **Низкая стоимость** – ещё один определяющий фактор для рынка небольших предприятий, не располагающих обширными денежными ресурсами.
6. **Простота обслуживания** – в случае, когда малые предприятия не могут позволить себе нанимать высококвалифицированный персонал или вкладываться в долгосрочное обучение сотрудников.

1.3. Обзор исследовательских работ в области открытых систем

Разумеется, проблема открытой архитектуры в последние годы является основой для проведения научно-исследовательских работ многими известными университетами мира. Разработки ведутся как в методологическом плане, так и более прикладные, вроде создания специализированного ПО для выполнения конкретных задач.

К примеру, ещё в 2007 году коллектив технологического университета в городе Харбин, Китай, опубликовали исследовательскую работу по проблемам модульной архитектуры станков с ЧПУ [5]. Цель их исследования – разработка открытой архитектуры систем ЧПУ на персональном компьютере в соответствии с открытой модульной архитектурой контроллеров. В процессе разработки широко применялись методы статического моделирования открытой архитектуры управления, исследовалось динамическое поведение контроллеров и представление потока данных в системе. В качестве результата работы представлен испытательный трёхкоординатный стенд под названием HIT-CNC.

Другая, более современная работа, выполненная университетом в Малайзии (University Tun Husein Onn Malaysia), описывает создание интепретирующего программного обеспечения для оборудования с ЧПУ с открытой архитектурой [6]. Оно разрабатывалось в среде LabVIEW² с целью приведения команд в кодах ISO в соответствие с внутренней структурой станков с ЧПУ. В статье рассматривается модульная структура разрабатываемого интерпретатора, а также вопросы получения и представления данных в системах управления.

Авторы Джордж Корреа, Николас Тумбс и Пласид Феррейра (Jorge Correa, Nicolas Toombs, Placid M. Ferreira) опубликовали в рамках Международного Инженерного Конгресса 2016 года (ASME 2016 International Mechanical Engineering Congress and Exposition) описание архитектуры систем управления станками с ЧПУ с открытым исходным кодом [7]. В их статье описывается применение компонентно-ориентированного подхода, при котором каждый компонент представлен в виде конечного автомата. Рассматривается применение многопроцессорного распределённого контроллера, а также основные алгоритмы управления с примерами реализации на платформе Arduino.

В статье авторов Атула Гупта и Вену Шриниваса (Atul Gupta и Venu Srinivasa), опубликованной в 2012 году, рассмотрены проблемы возникновения и профилактики сбоев, возникающих в ходе работы станков [8]. Работа посвящена созданию приложения для анализа и отладки критических данных оборудования, таких как скорость шпинделя, позиции осей, тип инструмента и других. В создании активно использовались возможности ЛВС для получения и обработки данных.

Ещё одна работа по созданию распределённой системы управления была представлена Рони Сапутрой (Roni Saputra) и соавторами в журнале «Mechatronics, Electrical Power and Vehicular Technologies» в 2014 году [9].

² Laboratory Virtual Instrument Engineering Workbench.

Она направлена на исследования в области переносного оборудования с ЧПУ и рассматривает модульную архитектуру управления. Авторы про вели моделирование производительности системы и анализ, что позволило добиться высокой линейной точности и малой стоимости оборудования.

1.4. Агентный подход к проектированию систем

Наиболее интересной с точки зрения подхода к построению систем является агентная гетерархическая³ архитектура, описанная в ряде научных статей [10–12]. Данный подход заключается в представлении системы управления в виде совокупности простых сущностей (агентов). Гетерархическая схема управления предполагает постоянное изменение поведения агентов в соответствии с изменениями во внешней среде. Каждый агент выполняет строго поставленную задачу, оставаясь обособленным компонентом, другими словами, ему не требуется подробной информации о состоянии других агентов сети. Автономность даёт возможность легко отследить изменения в поведении агента, но при этом, поскольку агенты не могут пользоваться глобальными данными, поведение системы в целом сильно зависит от внутренних правил поведения агентов.

В результате представление о много-агентных системах (МАС) вылилось в понятие «холонической производственной системы», которое было предложено в ходе проведения в 90-х гг. консорциума Интеллектуальных Производственных Систем [13]. Термин «холон» впервые был использован Артуром Кёстлером⁴ для объяснения биологических подсистем. В общем смысле он означает нечто, что является чем-то целым и самодостаточным, но при этом может быть частью чего-то большего. Другими словами, это некоторая устойчивая система в рамках другой, более крупной системы.

³ Гетерархия – система, образованная пересекающимися, разнообразными и одновременно сосуществующими структурами управления.

⁴ Кёстлер, Артур (Arthur Köestler) (1905–1983) – британский писатель и журналист.

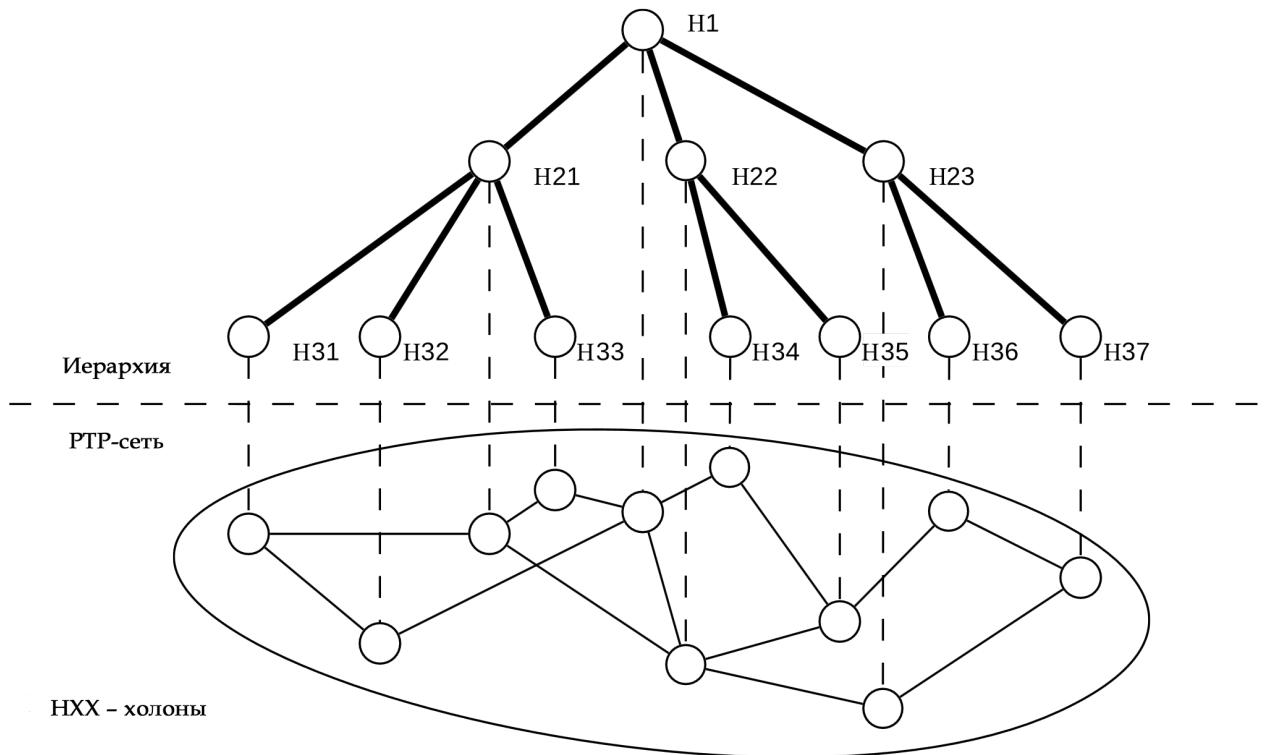


Рисунок 1.2 — Основное представление многоуровневой холонической структуры

Холоны склонны к образованию устойчивых форм, самодостаточных на протяжении своего развития, однако, в таких системах сложно провести черту между «целым» и «частью целого» (рис. 1.2). Существует два основных свойства холона [14]:

- дополнительность – способность холонов дополнять друг друга в пределах конкретной холонической системы, даже в случае разных выполняемых функций;
- уровневость – наличие иерархии в системе и разделение на холоны высших и низших порядков.

Основной особенностью холона является способность динамически менять свои иерархические связи и создавать новые, адаптируясь под состояние среды. Такие иерархические объединения называются *холархиями* и могут содержать сколько угодно уровней вложенности. Холон может состоять одновременно в нескольких холархиях с разным уровнем в них.

В ряде научных работ, опубликованных в последние несколько лет, показано, что холархии могут служить основой для создания производственной системы с децентрализованным управлением. Одна такая система под названием ADACOR была описана Паоло Лейтао и Франсиско Рестиво (Paolo Leitão и Francisco J. Restivo), которую они определили как «адаптивную холоническую архитектуру управления для распределённых промышленных систем» [15].

1.5. Актуальность исследовательской работы и обоснование её проведения

Во многих странах в настоящее время исследователи уже начали применять многоагентные системы в разных областях промышленного производства, от кооперации в рамках виртуального предприятия до планирования производственного процесса и учётом ресурсов и материалов. Особое место в развитии МАС занимает проблема управления оборудованием, которая является предметной областью основной части исследований по тематике распределённых сетей.

Поскольку холонические системы представляют собой логическое развитие многоагентных сетей, подобный подход позволяет сочетать надёжность и отказоустойчивость классических иерархических систем с гибкостью и масштабируемостью агентных сетей с децентрализованным управлением. Поэтому для реализации интеллектуальной системы для разрабатываемого компонента была выбрана холоническая архитектура.

Основным преимуществом холонической гетерархической производственной системы является то, что она не разделяет технические и программные компоненты. С точки зрения системы любой модуль или объединение модулей отвечает только за себя и управляет своей частью технологического процесса. Единственное, что необходимо для добавления в сеть

нового модуля – его регистрация в агентной распределённой сети как интерфейса с описанием его назначения.

К сожалению, в России тема проектирования распределённых систем управления пока не так развита. Конечно, разработки российских специалистов в этом направлении уже ведутся, к примеру, платформа Winnum от инновационного центра «Сколково», которая основана на концепции «Интернета вещей»⁵, но пока что данная отрасль находится в развивающемся состоянии. Поэтому рассмотрение возможности проектирования таких систем является актуальной темой для исследований на сегодняшний день. Автор подчёркивает важность исследований в данной области, поскольку промышленность является основой экономики государства и определяет уровень его материально-технической базы.

1.6. Выводы к первой главе

1. В последние годы появилась тенденция к повсеместному внедрению информационных технологий и вычислительных сетей в промышленные системы с охватом всех производственных отделов. Наблюдается переосмысление парадигм производства в связи с увеличением роли информационных технологий в производственный процесс. Ведутся обширные исследования в области создания современных систем управления и, в основном, эти исследования направлена на развитие систем с модульной архитектурой.
2. Согласно результатам анализа предметной области, можно заключить, что существующие системы управления технологическим оборудованием имеют недостатки, не позволяющие внедрять эти системы при организации производственных линий на малых предприятиях,

⁵ Интернет вещей (Internet of Things) – методология вычислительной сети физических предметов, оснащённых технологиями для обмена информацией между собой или с внешней средой.

работающих по индивидуальным заказам. В числе основных проблем – монолитная архитектура и централизованная система управления.

3. В качестве архитектуры системы управления наиболее интересна гетерархическая холоническая структура компонентов, поддающаяся масштабированию и динамическому изменению межкомпонентных связей.
4. Задачей текущей работы является применение холонического подхода для разработки компонента машинного зрения, являющегося частью системы управления.

Глава 2

Сравнение и описание существующих алгоритмов определения и классификации объектов на изображениях

2.1. Теория распознавания образов

Задача создания саморазвивающихся систем ставилась ещё в 60-е годы, на заре развития вычислительных машин, когда существовала наука под названием «кибернетика», из которой позже сформировалась современная информатика. Кибернетика, являясь, в свою очередь, подразделом математического анализа, изучает закономерности получения, хранения и обработки информации не только в вычислительных системах, но и в сложно устроенных системах вообще, в том числе биологических. Принципы, описываемые данной наукой, напрямую вели к формированию концепции *искусственного интеллекта* – вычислительной системы искусственного происхождения, склонной к анализу собственных действий и самообучению на его основе.

Искусственный интеллект – не только математическая и техническая задача, она дополнительно опирается на принципы функционирования биологических организмов и психологии. Сейчас наука всё ещё не в состоянии точно объяснить работу мозга и теории о природе его функционирования сильно разнятся: от описания химических преобразований до представления мозга в виде своеобразной «микросхемы», с импульсами и состояниями [16, 17]. Поэтому, хотя исследования в этой области ведутся уже на протяжении более чем полувека, создание полноценной самообучающейся системы остаётся недостижимой целью многих специалистов.

Но, разумеется, нельзя сказать, что всё это время было потрачено впустую. В ходе исследований выделилось большое количество технических новшеств, многие из которых находят применение и в настоящее время. Самые важные из них: нейронные сети, экспертные системы и системы распознавания образов.

В ходе решения задачи распознавания образов необходимо не только «увидеть» объект, но и определить совокупность его *признаков*, классифицировать их и выполнить принятие решения. Признак – некоторая количественная или качественная характеристика объекта. В природе для живых организмов распознавание образов является самой часто встречающейся задачей на их жизненном пути. Все организмы, имеющие органы чувств, априори способны к анализу поступающей информации и реагированию в соответствии с принятым на его основе решением.

Однако, человечество не знает точно принципы функционирования сознания, и поэтому невозможно спроектировать искусственные системы распознавания образов так, как это устроено у живых существ, поэтому существующие работы по тематике распознавания образов опираются на математическое описание системы, часто имеющее вид простой последовательности условий «если X, то Y».

Разумеется, в корне неверно расценивать образы только как визуальные. Если считать образ некой совокупностью данных, то умение извлекать из него необходимую информацию и является целью разработки распознавающих систем. Анализируемые данные могут иметь любую структуру: изображение, звуковая дорожка, результат спектрального анализа и т. д. Поэтому любая распознающая система строится, исходя из результатов анализа предметной области и структуризации области данных, с которыми система будет взаимодействовать.

Общая структура системы распознавания образов представлена на рисунке 2.1 и состоит из двух общих этапов: разработка системы и сам

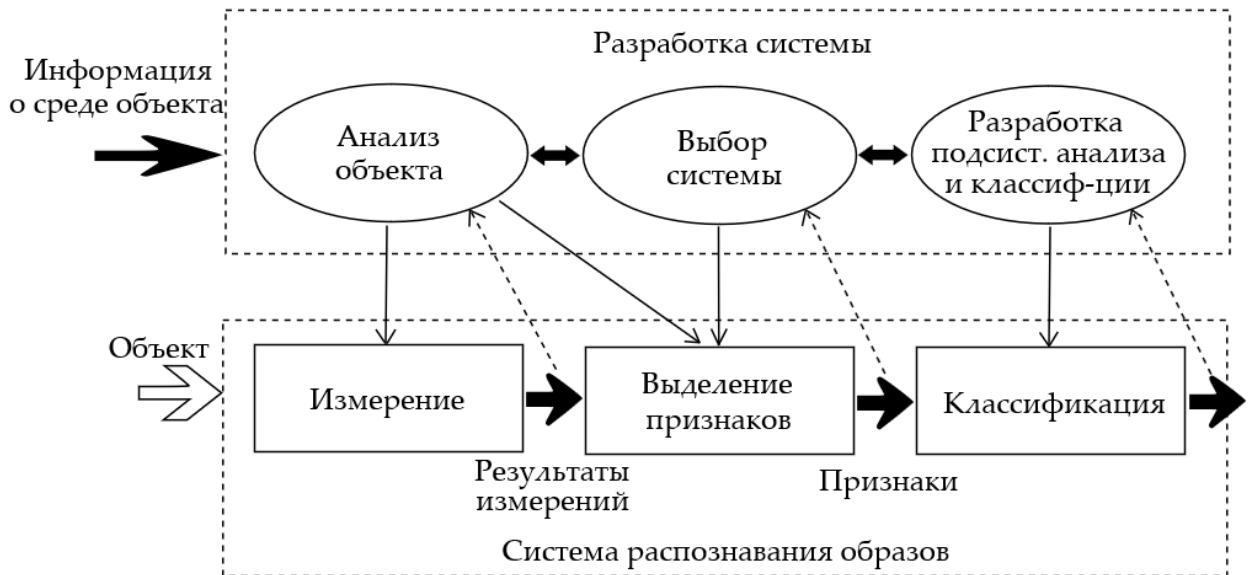


Рисунок 2.1 – Общая структура системы распознавания образов

процесс распознавания. Как видно, оба этапа тесно связаны между собой и состоят из последовательных блоков, обозначающих операции проектирования в укрупнённом виде. На стадии разработки анализируется информация об среде объекта, с которым будет работать будущая система, от количества и степени проработки информации напрямую зависит эффективность работы системы и точность классификации объектов. На следующей стадии проводится анализ собранной информации с результатом в виде перечня типовых характеристик объекта, которые будут заложены в алгоритмы системы. После всего вышеперечисленного проводится выбор подхода к созданию системы и, наконец, разработка подсистемы классификации, где исследуемые характеристики должны быть особым образом упорядочены и структурированы.

В процессе работы система принимает на вход объект, проводит измерения и выделяет характеристики, сравнивая их с заложенными в её базу знаний на более ранних этапах. Затем, на основе полученных характеристик, происходит присваивание объекта определённому классу или нескольким классам, что, в конечном счёте, является результатом, который можно использовать для решений более глобальных задач.

Любой алгоритм распознавания представляется как некая абстрактная функциональная система F , которая содержит три составляющих:

$$F = \{C, P, R\}, \quad (2.1)$$

где $C = \{c_i\}, i \in 1...N$ – множество известных классов (classes), по которым требуется распределить объекты, $P = \{p_j\}, j \in 1...M$ – множество признаков (properties), составляющих описание входного образа, а $R = \{r_k\}, k \in 1...K$ – множество правил (rules), которых придерживается система в процессе принятия решения.

Компоненты подразделяются на две составляющие: информационную (множества C и P) и методологическую (множество R). Способ описания образа во многом зависит от его физической природы, а от способа представления зависят и методы принятия решений, поэтому любая система включает в себя процессы синтеза и анализа образов. Синтез подразумевает формирование описания образа в системе на основе его признаков, а анализ подразумевает процесс принятия решения. Всего выделяют три общих подхода:

- 1. Метод сравнения с эталоном.** Данный метод является самым простым в реализации, но требует создания базы данных. Суть метода заключается в последовательном сравнении объекта с конечным набором эталонов-отображений, хранящимися в базе данных, где даже для одного объекта может быть множество его интерпретаций и модификаций. Каждый класс сопоставляется со своим набором эталонов $\Omega_k = \{\omega_i, i \in 1...N_k\}$. Разумеется, такой подход работает тем лучше, чем обширней и подробней созданная база данных и иногда попробная проработка может занять огромное количество времени и ресурсов. Одной из задач искусственного интеллекта при применении данного подхода является расширение базы данных при помощи

вычислительных алгоритмов, в соответствии с концепцией самообучения системы.

2. **Принцип кластеризации.** Данный принцип применяется, когда существует ряд признаков без какой-либо выраженной взаимосвязи друг с другом. Образ в этом случае представляется как n-мерный вектор в пространстве признаков:

$$X : \vec{x} = \{x_1, \dots, x_n\} \quad (2.2)$$

Каждому классу сопоставляется некоторое множество векторов в пространстве X . В результате признаковое пространство разбивается на ряд областей, каждая из которых обозначает один конкретный класс, такие области называются *кластерами* или *таксонами*. Принцип находит своё применение в области обработки больших объёмов количественных данных.

3. **Принцип общности свойств.** Данный принцип используется, если конечное число образов каждого класса слишком велико для получения надёжного описания эталонов, поэтому используется методика выявления отличительных особенностей класса. Образ обычно хранится в виде структуры или функции, а процесс распознавания заключается в выявлении необходимых свойств образа и последующем его сравнении со схемой конкретного класса.
4. **Искусственная нейронная сеть.** Данный метод предполагает, что существует большое количество примеров решений задачи распознавания, на которое опирается алгоритм в процессе отнесения к классу. В общем случае речь идёт о задаче обучения распознавания образов, где требуется выявить закономерность классификации объектов, основываясь на имеющейся выборке [18]. Задача сводится к поиску

среди множества зависимостей $F(x, \alpha)$ той, для которой вероятность классификации, отличной от представленной на входе, была бы минимальной:

$$\min(I(\alpha)) = \sum_{\omega=0,1} \int (\omega - F(x, \alpha))^2 \rho(\omega|x) \rho(x) dx, \quad (2.3)$$

где $I(\alpha)$ – функционал вида:

$$I(\alpha) = \int_{X,\omega} (\omega - F(x, \alpha))^2 \rho(x, \omega) dx d\omega, \quad (2.4)$$

а функция $\rho(x, \omega) = \rho(\omega|x)\rho(x)$ – совместная плотность пар x, ω в пространстве $X\omega$.

2.2. Задача классификации объектов

Основой системы распознавания является классификация. Под классификацией обычно понимается разбиение ряда объектов реального или виртуального мира по общим характеристикам. Как известно, классификация основывается на прецедентах, то есть, на объектах, правильная классификация которых известна. Прецеденты обычно выступают в роли образца при решении задачи [19]. Выделяют три класса характеристик-признаков:

1. **Качественные характеристики.** Описывают качества объекта: размер, цвет, положение и т. д. Они обычно описываются лингвистическими переменными при помощи методов теории нечётких множеств.
2. **Физические характеристики.** Они представляют собой результаты измерений или показания датчиков: плотность, температура и т. д. Этот класс характеристик обычно описывается и обрабатывается как элементы векторного пространства.

3. Структурные характеристики. Характерны для сложных объектов. Для их описания прибегают к формальным языкам, к примеру, к теории графов.

Предположим, существует некое множество образов $O = \{o_1, \dots, o_n\}$. У каждого объекта из множества существует пространство признаков и, как правило, оно является конечным. Пусть у объекта существует три характеристики: x_1, x_2 и x_3 , каждая из которых варьируется в пределах множества вариантов. То есть, множество описывается как:

$$X_i = \{x_{j_1}, \dots, x_{j_n}\}, j \in \{1, 2, 3\}, \quad (2.5)$$

где n – это количество конкретных видов каждого признака, а X_i – множество признаков конкретного объекта o_i . В таком случае каждый объект o_i в данной нотации будет описываться кортежем из значений множества признаков:

$$o_i = \langle x_{1i}, x_{2i}, x_{3i} \rangle \quad (2.6)$$

Задача системы распознавания состоит в том, чтобы разбить исходное множество O на подмножества, которые и будут являться классами. Обычно число классов конечно, но в общем виде их может быть бесконечно много. Для проведения разбиения необходимо составить выборку из множества вариантов сочетаний признаков с конечными результатами (*обучающую выборку*) и с вводом так называемого *решающего правила*¹ $a(o_i) : \{x_{1i}, x_{2i}, x_{3i}\} \rightarrow Y$, где $Y = \{y_1, \dots, y_m\}$. Задача алгоритма – используя решающее правило, соотнести объект с определённым классом из перечня $C = \{c_i\}, i \in 1\dots N$, то есть, поставить объекту o_i метку y_i о соответствии объекта классу c_i [20].

¹ Решающее правило, классификатор или индикаторная функция – правило отнесения образа к какому-либо классу, исходя из его вектора признаков.

У каждого решающего правила $a(o_i)$ существует понятие качества, определённого как вероятность отнесения объекта к различным классам. Данную характеристику формально можно представить в виде:

$$I(a) = \sum_{i=0}^{p-1} \int \Theta(a(O_i) - \omega_i) \rho(\omega_i | o_i) \rho(o_i) do_i, \quad (2.7)$$

где:

$$\Theta(z) = \begin{cases} 0, & \text{если } z = 0, \\ 1, & \text{если } z \neq 0. \end{cases} \quad (2.8)$$

Наиболее общей постановкой задачи классификации является вероятностная постановка, поскольку в подавляющем большинстве случаев приходится иметь дело со степенью неопределённости вектора признаков объекта. Другими словами, вектор признаков будет иметь вид многомерной случайной величины ξ . Неопределённость может иметь вероятностный характер и в этом случае множество пар обучающей выборки «объект-класс» будет называться *вероятностным пространством* с некой вероятностной мерой P . Появление объекта o_i будет случайным событием, которое описывается с помощью законов распределения вероятностей многомерной случайной величины. Задача распознавания сводится к определению на основе обучающей выборки вероятностных характеристик конкретной среды. К вероятностным характеристикам относятся:

- функция плотности распределения вероятностей $f_\xi(o_i)$;
- условная вероятность принадлежности объекта имеющимся классам $p(c_i | o_i^0)$;
- вероятность появления новых классов $p_i = p(c_i)$;
- функция условной плотности распределения вероятностей объектов внутри классов $f_i(x) = f_\xi(x | c_i)$.

Перечисленные выше вероятностные характеристики связаны отношениями. Если считать, что классы образуют полную группу событий, т. е. $c_i \cap c_j = \emptyset, i \neq j$, а сумма вероятностей всех классов равна единице, то:

$$f_\xi(o_i) = \sum_{k=1}^m p_k f_k(o_i) \quad (2.9)$$

$$p(c_i|o_i) = \frac{p_i f_i(o_i)}{f_\xi(o_i)} \quad (2.10)$$

Формула 2.9 носит название *формулы полной вероятности*, а формула 2.10 – *формула Байеса*².

В случае, если вероятностные характеристики известны, то в байесовской классификации задача сводится к тому, что при разбиении пространства R_n на некоторое количество непересекающихся областей необходимо добиться минимизации средней ошибки Q и средних потерь R от случая неправильной классификации объекта:

$$Q = \sum_{k=1}^m \int_{R_n/X_k} p(c_i|o_i) f(o_i) do_i \quad (2.11)$$

$$R = \sum_{i=1}^m \int_{X_i} R(c_i|o_i) f(o_i) do_i \quad (2.12)$$

Ошибка Q складывается из суммы вероятностей отнесений вектора o_i к ошибочным классам. Другими словами, пусть имеется три непересекающиеся области пространства R_n : X_1 , X_2 и X_3 , а множество классов содержит три элемента $C = \{c_1, c_2, c_3\}$. Тогда средняя ошибка Q_1 для области X_1 будет равна:

$$Q_1 = \int_{X_1} p(c_2|o_i) f(o_i) do_i + \int_{X_1} p(c_3|o_i) f(o_i) do_i, \quad (2.13)$$

² Байес, Томас (Thomas Bayes) (1702–1761) – английский математик.

то есть, сумме ошибок отнесения вектора o_i к классам, соответственно, c_2 и c_3 , тогда как он принадлежит классу c_1 . Соответственно определяются средние ошибки Q_2 и Q_3 для областей X_2 и X_3 . Средняя ошибка Q будет минимальна, если:

$$X_i = \{x \in c_i : p_i f_i(o_i) > p_j f_j(o_i)\}, i \neq j \quad (2.14)$$

Что касается потерь, то они также будут равны сумме потерь для каждого класса из множества C . Для случая из трёх классов:

$$\begin{aligned} R = & \int_{X_1} (r_{11} p_1 f_1(o_i) + (r_{21} p_2 f_2(o_i) + (r_{31} p_3 f_3(o_i))) d o_i + \\ & + \int_{X_2} (r_{12} p_1 f_1(o_i) + (r_{22} p_2 f_2(o_i) + (r_{32} p_3 f_3(o_i))) d o_i + \\ & + \int_{X_3} (r_{13} p_1 f_1(o_i) + (r_{23} p_2 f_2(o_i) + (r_{33} p_3 f_3(o_i))) d o_i \end{aligned} \quad (2.15)$$

2.3. Обзор существующих алгоритмов и методик обработки изображений

Машинное зрение, как очевидно из названия, имеет дело с изображениями, чаще всего растрового типа, а поступление на вход алгоритма изображения носит характер случайного события. Важно заметить, что задача распознавания и классификации образов по сей день не решена в полном объёме, несмотря на труды математиков всей второй половины XX века, и поэтому построение надёжной системы – вопрос далёкого будущего. Тем не менее, существует масса методик, позволяющих решить поставленную задачу с результатом, приближённым к идеальному, и степень приближенности сильно зависит от условий задачи.

При проектировании систем распознавания изображений, важно учитывать не только логику алгоритма обработки, но и технические характе-

ристики используемого оборудования и среды, где это техническое оборудование будет работать. Не существует универсального алгоритма, который одинаково хорошо работал бы в любых условиях и с любыми данными, поэтому для выбора алгоритма необходимо сперва ознакомиться с условиями рабочей среды, воздействиями среды на работу технических средств и организацией передачи изображений на микроконтроллер с программой-обработчиком. На качество получаемого цифрового изображения напрямую влияет множество факторов, которые можно условно разбить на четыре группы:

1. *Факторы, непосредственно связанные с композицией и типом изображения.* Это масштаб, фон, угол поворота объекта и т. д. Они напрямую зависят от качества принимающего оборудования и учёта технических характеристик оптических и разрешающих средств (например, фокусное расстояние объектива камеры, разрешение матрицы или посторонние тела на линзе).
2. *Факторы, вызванные условиями рабочего пространства,* то есть, то, что обычно можно видеть или ощущать даже невооружённым взглядом. К ним относятся такие параметры, как освещённость, запылённость, механические вибрации. В этом случае необходимо учитывать возможность дооснащения приёмопередающего устройства дополнительным освещением, виброгасящим или гиростабилизированным подвесом и другими защитными средствами.
3. *Факторы, вызванные полями наведения.* К ним относятся помехи, обусловленные наличием большого количества электроприборов в непосредственной близости от оборудования, электростатические или радиационные возмущения. Опять же требуют дополнительного дооснащения оборудования средствами защиты.

4. *Факторы, вызванные особенностями каналов передачи данных.* К ним относятся потери при передаче, которые разнятся при использовании разных каналов связи. Необходимо проанализировать допустимую меру искажения изображения и выбирать подходящий канал связи.

Обрабатывающие алгоритмы можно разделить по классам, согласно выполняемым ими функциям. Существует три основных категории алгоритмов: фильтрационные, обрабатывающие и обучающие [21, 22].

2.3.1. Алгоритмы фильтрации изображений

Главная задача алгоритмов фильтрации состоит в том, чтобы на исходном изображении выделить интересующие области, не прибегая к анализу изображения. Фильтры обычно применяются одинаково ко всем пикселям изображения, при этом области, прошедшие фильтрацию можно назвать областями с особыми свойствами. Чаще всего к ним прибегают для решения относительно несложных задач с входными изображениями общего типа. Существует несколько классов фильтров, среди которых можно назвать следующие:

1. **Пороговая бинаризация.** Является самым простым видом преобразования, суть которого состоит примерно в следующем: на исходном изображении выбирается определённое значение яркости пикселя (или значение цвета, если алгоритм имеет дело с изображением в RGB), а затем каждый пиксель, чья яркость выше заданной, получает максимум яркости (то есть, 255), а остальные, соответственно, минимум (0). Данный метод подходит для простых задач, к примеру, выделения объектов на однотонном фоне.

Порог может задаваться вручную, но в большинстве алгоритмов реализуется автоматическое его определение, что расширяет область под-

дающихся фильтру изображений. В качестве значения порога может выбираться среднее значение яркости пикселей из встречающихся на изображении или наибольший пик гистограммы.

2. **Классические фильтры.** К ним относятся преобразования Фурье, фильтры высоких и низких частот(ФВЧ и ФНЧ).

Под преобразованием Фурье обычно понимают *дискретное преобразование Фурье (ДПФ)*, а это, в свою очередь, есть аналог непрерывного преобразования Фурье для дискретного сигнала, содержащего N отсчётов [23]. Пусть сигнал имеет значения $a_n, n = 0, \dots, N - 1$, тогда ДПФ имеет вид:

$$A_k = \sum_{n=0}^{N-1} a_n e^{-\frac{2\pi i}{N} kn}, k \in \{0; N - 1\} \quad (2.16)$$

Однако, ДПФ изначально разрабатывался для анализа звуковых частотных сигналов, то есть, он может работать лишь с одномерными массивами. Для анализа изображений применяется двумерное ДПФ:

$$A_{uw} = \frac{1}{M \cdot N} \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} a_n e^{-2\pi j \left[\frac{mu}{M} - \frac{nw}{N} \right]}, m, n \in \{0; N - 1\} \quad (2.17)$$

Данный вид преобразования проводит ДПФ последовательно по всем строкам изображения и поэтому он весьма ресурсоёмок.

Что касается ФВЧ и ФНЧ, то их реализация примерно одинакова. Для каждого пикселя изображения выбирается окно, которое умножается на маску соответствующего фильтра. Различают фильтр Гаусса (для низких частот) [24] и фильтр Гabora (для высоких) [25].

3. **Вейвлет-преобразование.** Анализ изображения с применением математической функции. График функции имеет волнообразную фор-

му, откуда и появилось название «вейвлет»³. Суть метода заключается в проведении свёртки⁴ вейвлет-функции с сигналом. Метод также предназначен для обработки звуковых волн, поэтому в случае изображений проводится свёртка части изображения с паттерном. Существует ряд классических вейвлет-функций, использующихся для анализа изображений: вейвлет Хаара [26], вейвлет Морле [27] и другие.

4. Фильтрация функций. Она позволяет найти на изображении простую математическую функцию. Для каждой точки изображения отрисовывается множество порождающих функций. Наиболее классический случай – преобразование Хафа [28]. Алгоритм преобразования Хафа для прямых использует специальный массив (аккумулятор) для определения присутствия прямой $y = kx + b$. Для каждой точки изображения $(x; y)$ отрисовывается множество точек $(k; b)$, для которых равенство $y = kx + b$ верно.

Преобразования Хафа позволяют находить на изображениях любые функции, которые можно описать параметрически: окружность, синусоиду, эллипс и т. д. Но алгоритм Хафа нельзя назвать надёжным из-за того, что он имеет довольно низкую скорость работы и высокую чувствительность к бинаризации.

5. Фильтрация контуров. Данный тип фильтра применяется для случая, когда необходимо перейти от работы с изображением к работе с объектами на нём. Существует ряд контурных фильтров: Собеля [29], Лапласа [30] или Превитта [31], но чаще всего имеют дело с фильтром Кэнни [32], как наиболее эффективным. Алгоритм делится на несколько последовательных шагов:

³ От англ wave – волна.

⁴ Свёртка – математическое преобразование над двумя функциями, которое в общем виде можно определить как результирующую функцию, отражающую меру схожести двух исходных функций.

Сглаживание убирает шум при помощи размытия исходного изображения. Для этого оператор Кэнни использует фильтр, приближенный к первой производной гауссианы $\sigma = 1.4$:

$$B = \frac{1}{159} \begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix} \cdot A \quad (2.18)$$

Поиск градиентов выполняет поиск областей, где значение градиента максимально. Алгоритм Кэнни использует 4 фильтра в разных направлениях:

$$G = \sqrt{G_x^2 + G_y^2} \quad (2.19)$$

$$\Theta = \arctg\left(\frac{G_y}{G_x}\right) \quad (2.20)$$

Подавление немаксимумов или *Двойная пороговая фильтрация* для того, чтобы отображались только границы (локальные максимумы).

Трассировка области неоднозначности для итогового уточнения границ, выполняется путём подавления их краёв.

Существуют и другие, более экзотические версии фильтрационных алгоритмов (курвлет-преобразования⁵, итерационные фильтры и др.), однако они предназначены для совсем специфических задач и рассматривать их нет смысла.

⁵ От англ. curve – кривая.

2.3.2. Алгоритмы логической обработки изображений

В предыдущем разделе были рассмотрены фильтрационные методы, которые позволяют получить данные из входного изображения без анализа. Но в некоторых задачах недостаточно получить данные, для использования их нужно особым образом обработать. Далее приводится несколько типов обрабатывающих алгоритмов.

1. **Математическая морфология.** К ним относятся простейшие операции по наращиванию и эрозии изображений в бинарном виде. Они применяются, в основном, для выравнивания контуров и снижению шума на изображении.

Исходными данными для алгоритма служат исходное бинарное изображение и некий структурный элемент. В роли последнего, как правило, выступает геометрическая фигура в двоичном отображении, как показано на рисунке 2.2. На каждом элементе выделяется начальная точка, которая может быть любой точкой изображения. Результирующий кадр заполняется элементами со значением «1», формируя белое изображение, а затем каждому пикселию проводится зондирование, т. е. совмещение пикселя и структурного элемента так, чтобы начальная точка и пиксель совпали. После чего выполняется проверка некоторого условия на соответствие пикселей и, в случае его выполнения, пикселию присваивается значение «0». Таким образом может выполняться наращивание или сужение (эрзия) контура, в зависимости от поставленного условия.

2. **Контурный анализ.** Позволяет получать контуры из бинарных изображений. Наиболее распространённый алгоритм этого типа – алгоритм жука [33]. Данный метод получил название по схожести с поведением жука, который обходит препятствие.

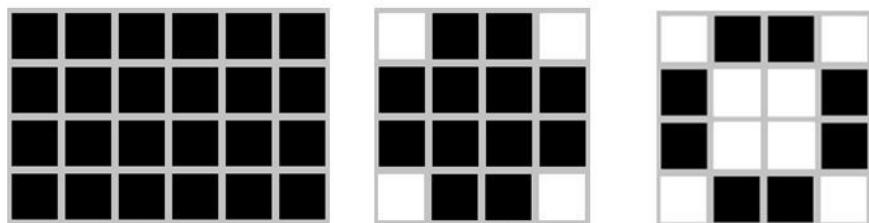


Рисунок 2.2 — Основные структурные элементы: прямоугольник, круг и кольцо

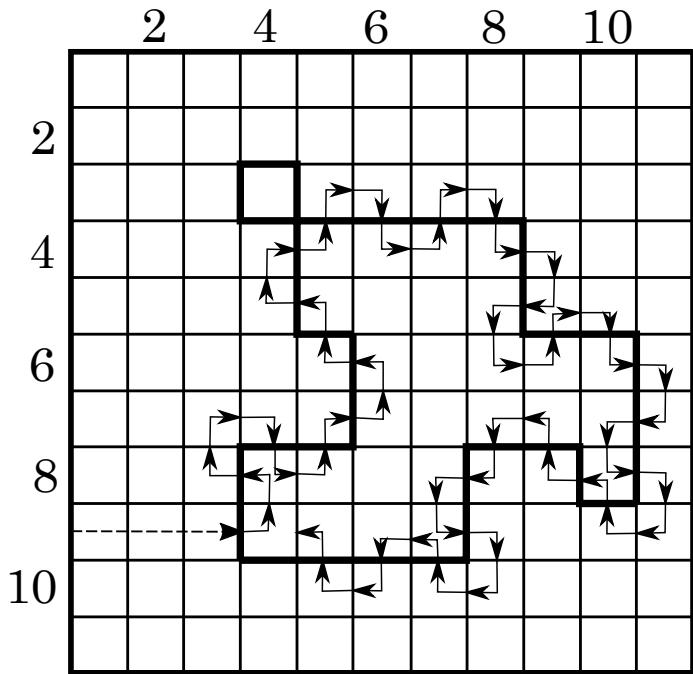


Рисунок 2.3 — Прослеживание контура алгоритмом жука

Порядок действий состоит в следующем. Некая активная точка, начиная свой путь с пикселя со значением «1», двигается по пикселям вперёд. Если она достигает пикселя со значением «0», то направление движения изменяется и точка начинает идти влево. Если следующий пиксель тоже имеет значение «0», то снова происходит поворот налево, а если «1», то направо (рис. 2.3). Алгоритм выполняется до достижения активной точкой своей исходной позиции, а местоположение границы определяется декартовыми координатами переходов с одного значения на другое. К сожалению, применение данного метода также требует наличия идеальных качественных характеристик для входного изображения.

3. Метод особых точек. Данный метод предназначен для работы сeriaми кадров, поскольку использует в своей основе методы выделения точек с уникальными характеристиками объекта для сопоставления с похожими классами. Методов выделения точек существует довольно много и их можно разделить по сложности реализации.

Выделение точек, стабильных в течение короткого времени. Чаще всего это или локальные максимумы изображения, области углов или точки максимумов дисперсии. Методы применяются для сведения нескольких кадров видео в один.

Выделение точек, стабильных при смене освещения и небольших движении объекта. Методами могут служить некоторые из вейвлетов, поиск бликов или специфических функций. Применяется в обучающих алгоритмах для последующей классификации.

Выделение точек, стабильных всё время. Достаточно сложные алгоритмы, к которым относятся SURF [34] и SIFT [35].

Метод SURF предназначен для поиска особых точек и создания их дескрипторов⁶ и основан на использовании детерминанта матрицы Гессе (гессиана), который в точках максимального градиента достигает экстремума. Для двумерной функции детерминант имеет вид:

$$H(fx, y) = \begin{bmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} \\ \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial y^2} \end{bmatrix} \quad (2.21)$$

$$\det(H) = \frac{\partial^2 f}{\partial x^2} \frac{\partial^2 f}{\partial y^2} - \left(\frac{\partial^2 f}{\partial x \partial y} \right)^2 \quad (2.22)$$

Метод SURF использует для нахождения гессианов фильтры разного масштаба. Для каждой точки изображения считается градиент (через

⁶ Дескрипторы – описательные элементы, инвариантные к повороту и изменению масштаба.

фильтры Хаара) и масштаб. После нахождения особых точек метод формирует их дескрипторы, имеющие вид массива из 64 (иногда 128) чисел, отражающих флюктуации⁷ градиента вокруг ключевой точки.

Алгоритм SIFT схож с алгоритмом SURF в общей методике расчёта. В методе SIFT детектирование особых точек выполняется через формирование пирамиды гауссианов и разностей гауссианов (2.23). Гауссиан имеет вид размытого при помощи Гауссова шума исходного изображения.

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y), \quad (2.23)$$

где L – значение гауссиана в точке $(x; y)$, G – гауссово ядро⁸, σ – радиус размытия, I – исходное изображение, $*$ – функция свёртки. Разность гауссианов – это изображение, полученное через попиксельное вычитание гауссиана исходного изображения из гауссиана с другим радиусом размытия. Точка считается особой, если она является локальным экстремумом разности гауссианов.

На следующем шаге алгоритма выбранные ранее особые точки уточняются при помощи аппроксимирования функции разности гауссианов (DoGF⁹) через многочлен Тейлора второго порядка:

$$D(x) = D + \frac{\partial D^T}{\partial x}x + \frac{x^T}{2} \frac{\partial^2 D}{\partial x^2}x, \quad (2.24)$$

где D – DoGF, $X = (x, y, \sigma)$ – вектор смещения, первая производная DoGF – градиент, вторая производная DoGF – матрица Гессе. Результатом уточнения является вычисление отклонения расчётного

⁷ Флюктуация – любое случайное отклонение от какой-либо величины.

⁸ Ядро (матем.) – некоторая функция $F(x, y)$, представимая в виде скалярного произведения в некотором пространстве[36].

⁹ Difference of Gaussian Function

экстремума от реального. Далее точка подвергается ещё нескольким проверкам, в том числе на определение её направления.

Наконец, для найденной особой точки вычисляется дескриптор. В методе SIFT им является вектор, направление которого также рассчитывается на ближайшем по масштабу ключевой точки гауссиане. Перед расчётом дескриптора окно ключевой точки поворачивают на угол её направления, найденный на прошлом этапе, чем достигается инвариантность относительно поворота.

Как можно заметить, алгоритмы SURF и SIFT являются достаточно трудоёмкими для реализации и требуют больших вычислительных мощностей. Кроме того, они охраняются патентом, что не даёт права использовать их в коммерческих проектах, без получения соответствующего разрешения.

2.3.3. Алгоритмы обучения

Алгоритмы обучения не работают с изображениями напрямую, но позволяют принимать решения относительно имеющихся объектов. В 80 % суть работы алгоритма реализует описанное в разделе 2.2, поэтому останавливаться на этом подробно нет смысла. Автор лишь хочет отметить, что алгоритмов обучения существует огромное количество и среди имеющихся можно отметить, например, K-means [37], AdaBoost [38] и Support Vector Machine (SVM) [39].

K-means – достаточно простой обучающий алгоритм кластеризации, известный ещё с 50-х годов. Его действием является минимизация суммарного квадратичного отклонения точек кластеров от их центров.

AdaBoost (Adaptive boosting) – популярный алгоритм усиления классификаторов, который используется в сочетании с другими алгоритмами для усиления их эффективности. Обычно он используется при проведении

бинарной классификации.

SVM (метод опорных векторов) – довольно мощный классификатор, работающий аналогично AdaBoost, но обучающий алгоритм сложнее и требует выбора правильного ядра. Идея заключается в переводе исходных векторов в пространство с более высокой размерностью и поиск разделяющей гиперплоскости с наибольшим зазором в данном пространстве.

2.3.4. Критерии выбора методик обработки

В ходе анализа имеющихся методов распознавания объектов на изображении является очевидным, что далеко не все из них могут использоваться при решении поставленной задачи. Какие-то требуют больших вычислительных мощностей, какие-то тратят слишком много времени на выполнение. Исходя из описания состава технических средств, формирующих компонент машинного зрения (состав и характеристики оборудования описаны в разделе 3.2 данной выпускной квалификационной работы), следует отметить, что для ускорения расчётов и обеспечения обработки в реальном времени необходимо выбрать наиболее оптимизированные из них. С другой стороны, хотя задача обеспечения высокой точности в рамках данной работы не стоит, в перспективе будет необходимо рассмотреть этот вопрос, поэтому выбор наиболее быстрого алгоритма в ущерб точности и надёжности не является наилучшим.

Поскольку ни один алгоритм не подходит для выполнения поставленной задачи на 100 %, логичным является сочетание двух-трёх разных методов обработки, направленных на разные задачи. Реализованный алгоритм описан в следующем разделе.

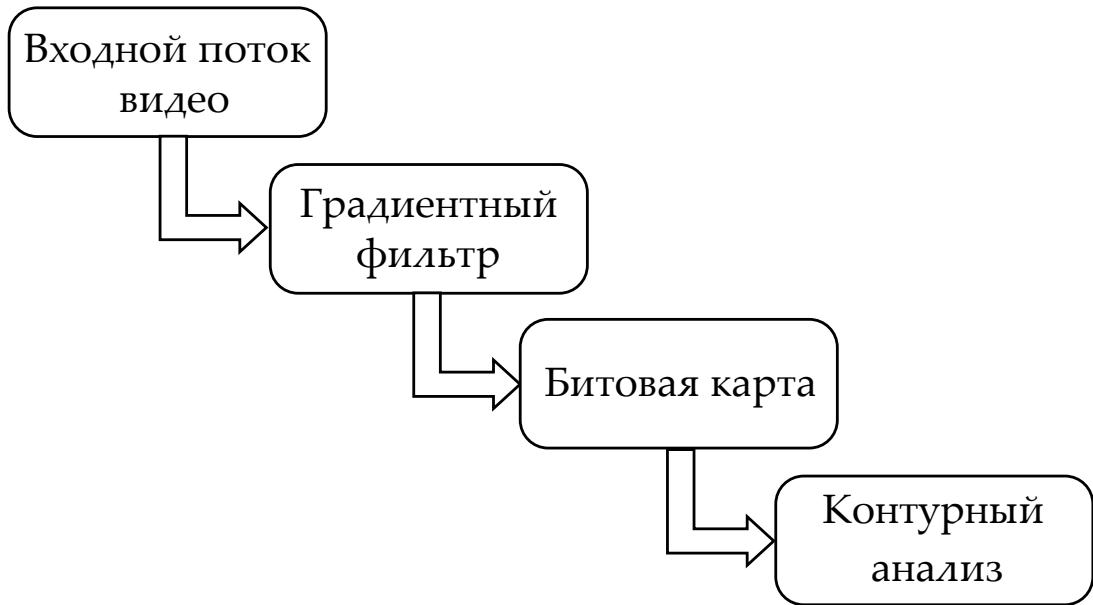


Рисунок 2.4 — Основные стадии используемого алгоритма обработки

2.4. Описание применяемого алгоритма работы

Очевидно, что при разработке обрабатывающей программы необходимо изучить условия, в которых она будет работать. В разработке алгоритма распознавания меток учитывались следующие условия работы модуля:

1. Процесс обработки кадра видео выполняется на микрокомпьютере Odroid, который обладает сравнительно невысокими техническими характеристиками, и поэтому необходимо избегать проведения лишних вычислений.
2. Метка имеет вид круглого чёрно-белого изображения, поэтому надобность в обработке цветных изображений отсутствует.
3. Метка, как правило, самый высококонтрастный объект в кадре.

Приближенный порядок действия обрабатывающего алгоритма состоит из четырёх общих этапов, как показано на рисунке 2.4.

I этап. Приём на вход потокового видео. Входной поток представлен в виде набора кадров (изображений), поступающих на обработчик с некото-

рой частотой. Для оптимизации расчётов каждое изображение переводится в полутоновый формат по формуле:

$$G = \frac{R + G + B}{3}, \quad (2.25)$$

где G – значение яркости пикселя в полутоновом изображении в диапазоне $\{0\text{--}225\}$, R, G и B – значения яркости, соответственно, красной, зелёной и синей составляющих пикселя исходного изображения.

II этап. На данном этапе входное изображение проходит обработку детектирования границ методов градиентной фильтрации. В алгоритме применяется фильтр Собеля, уступающий фильтру Кэнни в качестве выходного результата, но работающий на порядок быстрее из-за отсутствия уточняющих вычислений.

III этап. Пороговая бинаризация. Для дальнейшей простоты определения границ выполняется бинаризация изображения, после чего изображение принимает вид двумерного массива из нулей и единиц.

IV этап. Контурный анализ. На этом этапе изображение анализируется с точки зрения закрытых контуров, найденный контур подразумевает под собой метку. Выполняется проверка на закрытость контура, таким образом отсеиваются случайные флюктуации на изображении и метки, частично ушедшие за границу кадра.

Результат работы алгоритма представлен на рисунке 2.5, но следует оговориться, что данное изображение иллюстрирует отладочную работу алгоритма. В ходе отладок в качестве приёмного оборудования использовалась веб-камера с разрешающей матрицей 0,3 МРх (размер кадра 640×480 точек). Кроме того, в качестве выходного кадра используется изменённое оригинальное изображение, тогда как в работе платформы демонстрация входного кадра при определении метки не осуществляется (за исключением трансляции видеопотока на пользовательский виджет). Текст программы

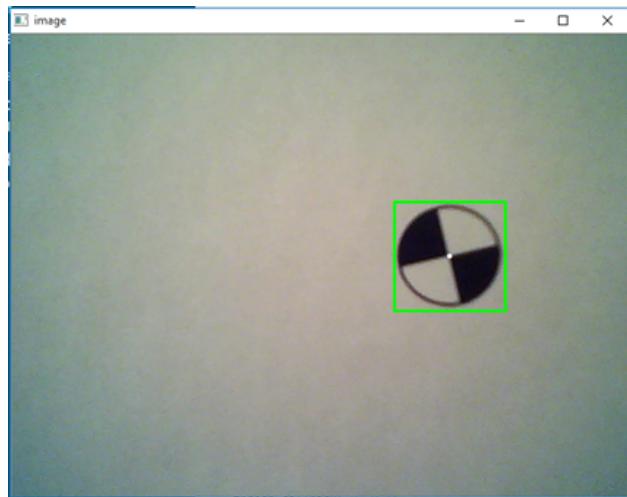


Рисунок 2.5 – Пример работы обрабатывающего алгоритма на языке Python представлен в приложении [A](#).

2.5. Выводы ко второй главе

1. Приведено описание и постановка задачи классификации объектов, описаны имеющиеся методы. Определён предмет распознавания образов и его роль в исследованиях.
2. Описаны и проанализированы методы обработки изображений, проведена их классификация в соответствии с проводимой функцией. Каждый метод подходит под свой класс задач реализации и в сложных задачах может потребоваться их комбинация.
3. Приведена структура обрабатывающего алгоритма с описанием основных этапов обработки, исходя из рассмотренных условий работы оборудования.

Глава 3

Состав и техническое описание разрабатываемого компонента машинного зрения

3.1. Описание архитектуры разрабатываемой двукоординатной платформы

Очевидно, что для разработки и тестирования модульной системы управления необходимо устройство, на котором эту систему можно запускать. В качестве тестирующего стенда разрабатывается установка для селективного отверждения фотополимера, предназначенная для работы с поверхностями любой формы. В основе процесса обработки лежит лазерная обработка, аналог технологии LDS¹.

В составе установки находятся: неподвижный рабочий стол, на котором находится обрабатываемый объект (заготовка), лазерная головка с возможностью перемещения по двум координатам при помощи шаговых электродвигателей и направляющих, источник лазерного излучения. Лазерная головка снабжена гиростабилизированным подвесом на основе модифицированной платформы Стюарта, перемещения осуществляются с помощью пьезомоторов. Данный подвес позволяет удерживать лазерный луч в фокусе и осуществлять обработку наклонных поверхностей за счёт возможности осуществления наклона источника лазерного излучения и перемещения его по координате Z. Ещё одна функция лазерного блока - гашение колебаний, для определения которых применяются одноосевые акселерометры, а также камера, отслеживающая в реальном времени форму пятна.

¹ Laser Direct Structuring

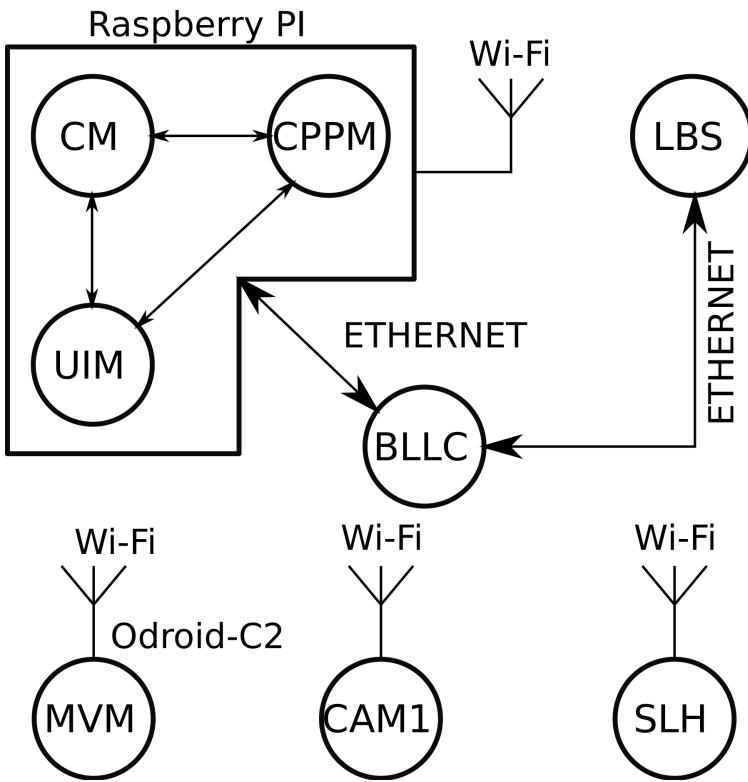


Рисунок 3.1 — Структура модулей системы управления установкой

Общее изображение установки представлено в приложении Б. Далее приводится общее описание модулей, из которых состоит система управления установкой (рис. 3.1):

Коммуникационный модуль (СМ). Основное назначение модуля состоит в регистрации и диспетчировании новых модулей и обеспечении устойчивости системы к отказам.

Базовый низкоуровневый управляющий контроллер (BLLC). Физически является блоком управления, основные функции: генерация управляющих сигналов для электроприводов координатного стола и обработка сигналов, поступающих от датчиков установки. Интеракция модуля с внешней средой заключается в приёме на вход управляющей программы (G-code²), передаче данных от датчиков и непосредственно включением и выключением источника лазерного излучения, используя для приёма/передачи протокол USART. В остальном процессе работы модуль автономен

² G-code (G-код) – условное название языка программирования для оборудования с ЧПУ. Утверждён как ГОСТ 20999-83 (в СССР [40]) и как стандарт ISO 6983-1:2009 (в остальном мире [41]).

и способен самостоятельно обрабатывать ошибки и реагировать на них, все действия модуля регистрируются в журнале событий.

Также в некоторых случаях BLLC может принимать одиночные управляющие сигналы, необходимые для взаимодействия установки с оператором. BLLC имеет отладочный управляющий порт USART, посредством которого оператор может подавать на контроллер команды в текстовом виде. Данная функция предусмотрена для реагирования на нештатные ситуации и отладки работы модуля.

Источник лазерного излучения (LBS). Функционал LBS напрямую связан с настройками лазера: включение/выключение, измерение мощности и обработка данных от датчиков. LBS занимает подчинённую роль по отношению к BLLC, но держит с ним постоянную связь, поскольку в случае нештатной ситуации лазер должен сразу же выключиться во избежание порчи заготовки или пожара. LBS способен обрабатывать исключения самостоятельно, но в некоторых случаях требуется связь с BLLC для подтверждения своих действий.

Интеллектуальная лазерная головка (SLH). Его функционал был описан выше: наклон источника лазерного излучения, фокусировка и компенсация колебаний. Как и остальные модули, обрабатывает события автономно и оповещает сеть при помощи соответствующих сигналов.

Модуль машинного зрения (MVM). Основными его функциями являются поиск реперной точки на заготовке и трансляция видеопотока на пользовательский виджет. Перед запуском управляющей программы на исполнение MVM по запросу возвращает найденные координаты нулевой точки, после чего переходит в режим ожидания.

Модуль интерфейса пользователя (UIM). То, с чем напрямую работает пользователь системы. Представляет собой агрегатор пользовательских панелей (виджетов) и веб-приложение для их отображения. Набор виджетов зависит от типа взаимодействия с другими компонентами.

Логика интерфейса прописана в каждом из компонентов и подгружается в приложение при необходимости.

Модуль подготовки управляющих программ (CPPM). С его помощью выполняется подготовка G-кода в браузере. Физически это веб-приложение, встраиваемое в UIM.

3.2. Модуль машинного зрения

3.2.1. Структура компонентов модуля

С точки зрения холонического подхода к проектированию систем управления, модуль машинного зрения находится в составе *интеллектуального холона* вместе с SLH. Его общая структура и компоненты представлены на рисунке 3.2. Он содержит две составляющие: модуля управления пьезомоторами и модуля определения положения, которые взаимодействуют между собой. Интеллектуальный блок имеет связь с остальной системой при помощи Wi-Fi модуля, передача данных внутри блока и между блоком и внешними модулями осуществляется с использованием открытой библиотеки коротких сообщений Nanomsg.

Для создания встраиваемых систем нецелесообразно использовать общепринятые библиотеки, применяемые в промышленных решениях. В открытых системах лучшим решением будет использовать преимущества низкоуровневых сокетов в сочетании с маршрутизацией сообщений. Поэтому протокольное сообщение основано на так называемых *очередях сообщений*. Этот протокол является асинхронным, то есть отправитель и получатель взаимодействуют друг с другом только через очередь. В системе применяется два вида сообщений: децентрализованный обмен сообщениями между модулями и широковещательные пакеты.

Поскольку MVM по большей части автономен, а обмен сообщениями происходит лишь в исключительных случаях, то отказ этого компонента

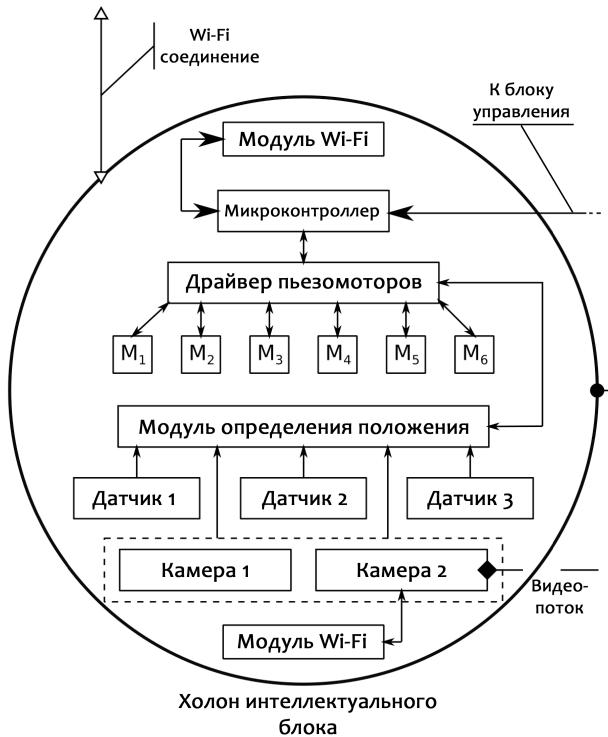


Рисунок 3.2 — Структура интеллектуального холона

критичен только в процессе поиска нулевой точки заготовки. В остальных случаях СМ оповещает оператора, что модуль недоступен, а процесс обработки заготовки при этом не нарушается. Как и остальные крупные модули, данный блок имеет внутреннее хранилище для журнала событий, базы данных и компонентов пользовательского интерфейса.

Сеть испытывает нагрузку лишь в случае передачи потокового видео пользователю, поэтому есть смысл в организации отдельного канала для данной функции. Применение отдельного канала особенно актуально в связи с тем, что BLLC и LBS вынуждены поддерживать обмен сообщениями из соображений безопасности. Таким образом, интеллектуальный блок имеет две связи с остальной системой: через USART с BLLC и через RTSP³, специализированным протоколом для передачи видеопотока.

³ Real Time Streaming Protocol

3.2.2. Состав технических средств

В качестве приёмного оборудования используется IP-камера с Wi-Fi модулем для обеспечения беспроводной связи для передачи видеопотока. Её характеристики представлены в таблице 3.1.

Таблица 3.1 — Характеристики используемой IP-камеры.

Параметр	Значение
Максимальное разрешение, точек	1280 × 920
Частота кадров, в с.	25–30
Диаметр линзы, мм	8
Размеры платы, мм	38 × 38
Интерфейсы	Wi-Fi, IR-CUT
Тип сенсора	1/3 "AR 0130 CMOS
Поддерживаемые виды связи	10/100M Ethernet / RTSP / PPPoE / FTP / DHCP
Вид сжатия видео	H.264, AVI

В качестве контроллера для исполнения обрабатывающей программы используется одноплатный компьютер Odroid-C2. Его характеристики представлены в таблице 3.2.

Таблица 3.2 — Характеристики Odroid-C2.

Параметр	Значение
Процессор	Amlogic ARM Cortex-A53 1.5 GHz
Память	SDRAM 2048 Mb DDR3
Графический процессор	ARM Mali-450 MP3 GPU
Кодеки	H.264 30FPS и H.265 60FPS VPU
Размеры, мм	85 × 56 × 18

Тестирование программной реализации обрабатывающего алгоритма проводилось на ноутбуке марки Lenovo B570e со следующими характеристиками (таблица 3.3):

Таблица 3.3 — Характеристики ноутбука Lenovo B570e.

Параметр	Значение
Процессор	Intel Celeron 1.5 GHz dual core
Память	SDRAM 2048 Mb DDR2
Графический процессор	GeForce GT410M 1024 Mb VRAM
Операционная система	Microsoft Windows 7 x64

Таким образом, можно заметить, что по техническим характеристикам ноутбук близок к одноплатному компьютеру Odroid-C2, поэтому результаты тестирования обработчика можно считать близкими к реальной работе программы. Изображения используемых технических средств представлены в приложении В.

3.2.3. Состав программных средств

Python – высокоуровневый язык программирования общего назначения, разработанный Гвидо ван Россумом в 1991 году, с тех пор постоянно улучшается и совершенствуется. Поддерживает написание скриптов, функциональное и объектно-ориентированное программирование. На данный момент существует две параллельные ветви его развития, Python 2.X.X и Python 3.X.X, поскольку между ними имеются фундаментальные различия, что исключает возможность их совместимости.

В ходе работы Python версии 2.7.6 использовался для составления и отладки алгоритма поиска метки, поскольку данный язык характеризуется наглядностью кода, простотой использования и большим количеством литературы. Версия 2.X.X была выбрана по причине того, что библиотека

OpenCV на тот момент ещё не имела стабильной версии для Python 3.X.X. К сожалению, Python не лишен недостатков, и основной среди них – низкая скорость работы, поэтому для реализации алгоритма лучшим решением будет использовать C.

OpenCV – библиотека обработки изображений и потокового видео, реализованная для большого количества языков программирования, среди которых есть все основные: C/C++, Java, Python и другие. Данная библиотека имеет множество реализованных алгоритмов, которые достаточно просто встраивать в логику программы, кроме того, она является ПО с открытым исходным кодом.

Поскольку OpenCV достаточно громоздкое решение для реализации алгоритма на микроконтроллере, данная библиотека также использовалась на этапе отладки алгоритма распознавания.

3.3. Практические результаты

Разработанная управляющая программа была протестирована в работе оборудования. Реальный инструмент в рабочем органе установки был заменён шариковой ручкой. Тесты проводились по следующему сценарию:

1. Рабочий орган установки с закреплённой на нём камерой выполнял последовательный проход рабочего пространства стола. На столе был размещён лист бумаги А4 с изображением метки. Использовалось несколько листов с разным размером изображения.
2. Как только метка попадает в поле зрения камеры, алгоритм начинает определять, попала ли метка в кадр полностью, для чего проводится вычисление размеров контура. Так как метка круглая, то её длина и ширина должны быть равны. В алгоритм заложена возможность небольшого отклонения ввиду небольшого разрешения камеры.

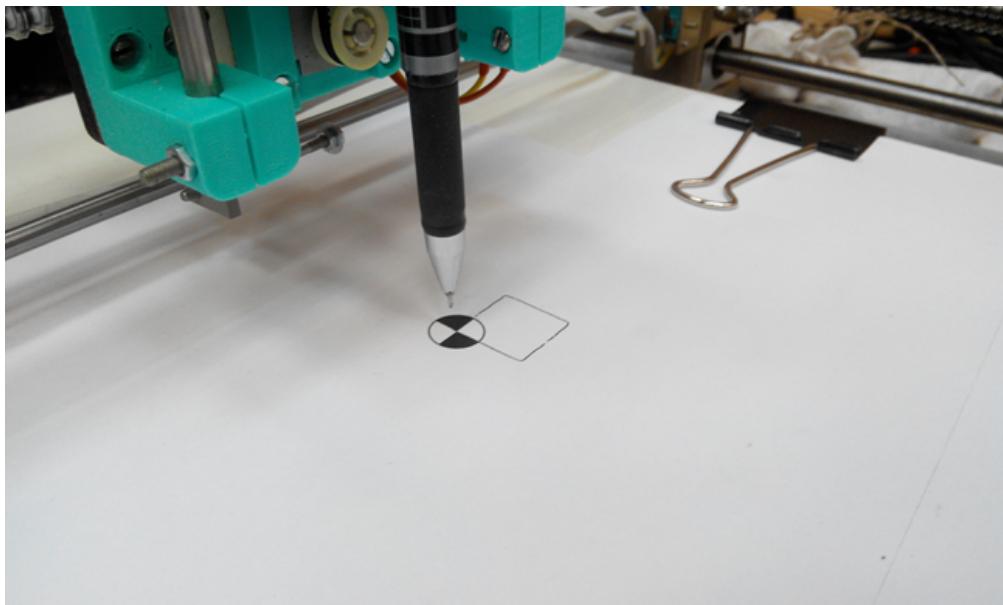


Рисунок 3.3 — Результат выполнения тестовой программы

3. Если метка попала в кадр полностью, то она подсвечивается на выходном изображении зелёным цветом (рис. 2.5), после чего программа посылает на вход управляющего контроллера установки сигналы перемещения рабочего органа по координатам, пока центр кадра не совпадёт с центром метки. Также программа возвращает координаты первого вхождения метки в кадр.
4. Начинается выполнение тестовой управляющей программы.

В качестве одного из результатов было получено изображение квадрата, одна из вершин которого совпадает с центром реперной метки. Можно заключить, что программа успешно определила метку и использовала её как нулевую координату для запуска на выполнение управляющей программы (рис. 3.3).

Также для размещения технических элементов компонента был спроектирован в пакете SolidWorks корпус и в дальнейшем построен на 3D-принтере (приложение В). При проектировании модели корпуса учитывалось наличие в устройстве охлаждающего вентилятора во избежание перегрева элементов платы камеры.

3.4. Выводы к третьей главе

1. Спроектирована архитектура координатного станка с описанием работы модулей, входящих в его систему управления. Также выбран протокол коммуникации модулей.
2. Описана структура модуля машинного зрения, состав технических и программных средств, сценарии коммуникации компонентов модуля друг с другом и с внешней средой.
3. Представлены некоторые практические результаты работы, полученные при выполнении тестовых запусков обрабатывающей программы.

ЗАКЛЮЧЕНИЕ

Исходя из анализа предметной области, можно заключить, что промышленность в наши дни проходит процесс переосмыслиния, что подтверждается большим научным интересом со стороны специалистов всего мира и выражается в актуальности исследовательских работ. Данная работа также является попыткой создания современной системы управления, широко использующей возможности вычислительных систем для совершенствования её рабочих характеристик. В процессе выполнения выпускной квалификационной работы были решены следующие задачи:

1. Был проведён анализ существующих промышленных решений в области систем управления оборудование с ЧПУ, определены недостатки его монолитной архитектуры. Показано, что для небольших производств применение таких систем нерентабельно из-за технических ограничений, недостаточной гибкости и прозрачности, а также высокой стоимости.
2. Был предложен подход к проектированию оборудования и систем управления этого оборудования, основываясь на теории агентных систем, в частности, с использованием холонического подхода. Для современных станков, ориентированных на малое производство, важно иметь децентрализованное управление, возможность расширения функционала за счёт внедрения новых компонентов, возможность проведения быстрой переориентации производства и более строгий контроль за выполнением технологических процессов.
3. Была составлена структура модуля машинного зрения, определено его место в системе управления и его поведение в работе оборудования. Определён ряд технических средств, включая протоколы связи модуля с внешней системой.

4. Была рассмотрена теория распознавания образов и существующие алгоритмы обработки изображений. Исходя из технических характеристик устройств, входящих в модуль машинного зрения, был спроектирован алгоритм распознавания меток на заготовках.
5. Выбранный алгоритм был реализован в виде обрабатывающей программы, прошедшей ряд тестов на определение изображения метки и использование полученной информации в качестве нулевой координаты для управляющей программы

Автор надеется, что в будущем подобные системы станут доступнее для организаторов малых предприятий и займут свою нишу на рынке промышленных систем. В качестве будущих исследовательских работ планируется улучшение качества разрабатываемой системы и тестирование при выполнении реальных технологических процессов по производству изделий.

ПЕРЕЧЕНЬ РИСУНКОВ

1.1	Пример централизованной архитектуры станка	18
1.2	Основное представление многоуровневой холонической структуры	23
2.1	Общая структура системы распознавания образов	29
2.2	Основные структурные элементы: прямоугольник, круг и кольцо	43
2.3	Прослеживание контура алгоритмом жука	43
2.4	Основные стадии используемого алгоритма обработки	48
2.5	Пример работы обрабатывающего алгоритма	50
3.1	Структура модулей системы управления установкой	52
3.2	Структура интеллектуального холона	55
3.3	Результат выполнения тестовой программы	59
Б.1	Координатная платформа	74
Б.2	Интеллектуальная лазерная головка	74
B.1	IP-камера	75
B.2	Компоненты интерфейса Wi-Fi камеры	75
B.3	Одноплатный компьютер Odroid-C2	76

ПЕРЕЧЕНЬ ТАБЛИЦ

3.1	Характеристики используемой IP-камеры.	56
3.2	Характеристики Odroid-C2.	56
3.3	Характеристики ноутбука Lenovo B570e.	57

СПИСОК ИСПОЛЬЗУЕМЫХ СОКРАЩЕНИЙ

API – Application Programming Interface (прикладной программный интерфейс)

ERP – Enterprise Resource Planning (планирование ресурсов предприятия)

MES – Manufacturing Execution System (система исполнения производства)

PTP – Peer-to-peer (одноранговая сеть)

SCADA – Supervisory Control And Data Acquisition (система диспетчерского управления и сбора данных)

SCM – Supply Chain Management (управления цепочкой поставок)

SIFT – Scale Invariant Feature Transform (масштабирование инвариантных функций)

SSM – Sales and Services Management (управление продажами и услугами)

SURF – Speeded Up Robust Features (ускоренные функции)

SVM – Support Vector Machine (метод опорных векторов)

USART – Universal Synchronous/Asynchronous Receiver/Transmitter (универсальный синхронный/асинхронный приёмопередатчик)

ДПФ – Дискретное преобразование Фурье

ЛВС – Локальная вычислительная сеть

МАС – Много-агентная система

ПК – Персональный компьютер

ПО – Программное обеспечение

ФВЧ – Фильтр высоких частот

ФНЧ – Фильтр низких частот

ЧПУ – Числовое программное управление

СПИСОК ЛИТЕРАТУРЫ

1. Утин Я. И. Цифровая перестройка // Iron Magazine. — 2016. — Т. 2, № 12. — С. 18–23.
2. Hermann Mario, Pentek Tobias, Otto Boris. Design principles for Industrie 4.0 scenarios // System Sciences (HICSS), 2016 49th Hawaii International Conference. — 2015. — Vol. 1. — P. 3928–3937.
3. Форд Генри. Моя жизнь, мои достижения. — Москва : ACT, 2015. — 352 с.
4. Frazer Julie, Hakanson Bill, Schaefer Tom. MES explained: A high level vision. — MESA International, 1997.
5. Xiong-bo Ma, Yong-zhang Wang. Development of a PC-based open architecture software-CNC system // Chinese Journal of Aeronautics. — 2007. — Vol. 20. — P. 272–281.
6. Yusof Yusri, Latif Kamran. New interpretation module for open architecture control based CNC systems // Procedia CIRP. — 2015. — Vol. 26. — P. 729–734. — 12th Global Conference on Sustainable Manufacturing.
7. Correa Jorge, Toombs Nicolas, Ferreira Placid M. Implementation of an open-architecture control for CNC systems based on open-source electronics // Procedia CIRP. — 2016. — Vol. 2. — ASME 2016 International Mechanical Engineering Congress and Exposition.
8. Gupta Atul, Srinivasa Venu U. Black box for machine tools; based on open CNC architecture control systems // International Journal on Control System and Instrumentation. — 2012. — Vol. 3.
9. Saputra Roni P., Atmaja Tinton D., Prawara Budi. Distributed control system design for portable pc based CNC machine // Mechatronics, Electrical Power and Vehicular Technology. — 2014. — Vol. 5. — P. 37–44.
10. Formal specification of holonic control system ADACOR product holon, using high-level Petri nets / Paulo Leitao, Armando W. Colombo, Fran-

- cisco Restivo, Roland Schoop // IEEE International Conference on Industrial Informatics, 2003. INDIN 2003. Proceedings. — 2003.
11. Han Zhang Ruifeng Guo Cong Geng. An architecture model of cloud manufacturing based on multi-agent technology // The 26th Chinese Control and Decision Conference (2014 CCDC). — 2014. — Vol. 1. — P. 3725–3729.
 12. Morales-Velazquez Luis, Romero-Troncoso Rene J., Osornio-Rios Roque A. Open-architecture system based on a reconfigurable hardware-software multi-agent platform for CNC machines // Journal of Systems Architecture. — 2010. — Vol. 56. — P. 407–418.
 13. Wang Lihui, Nee Andrew Y. C. Collaborative Design and Planning for Digital Manufacturing. — Springer-Verlag London, 2009. — 413 p. — ISBN: [978-1-84882-286-3](#).
 14. Дорфман Л. Я. Системная традиция: теория холомности и холон. — СПб : Алетейя, 2006. — С. 52–54.
 15. Leitao Paulo, Restivo Francisco J. Implementation of a holonic control system in a flexible manufacturing system // IEEE Transactions on systems, man and cybernetics. — 2008. — Vol. 38, no. 5. — P. 699–709.
 16. Савельев С. В. Возникновение мозга человека. — Москва : МЕДИ, 2010.
 17. Rock David. Your Brain at Work: Strategies for Overcoming Distraction, Regaining Focus, and Working Smarter All Day Long. — 1 edition. — HarperCollins e-books, 2009. — ISBN: [978-0062312822](#).
 18. Вапник В. Н., Глазкова Т. Г., Кощеев В. А. Алгоритмы и программы восстановления зависимостей. — Москва : Наука, 1984.
 19. Падун Б. С. Методы поиска технических решений. — СПб : СПбНИУ ИТМО, 2014. — С. 37–48.
 20. Местецкий Л. М. Математические методы распознавания образов. — Москва, ВМиК, кафедра «Математические методы прогнозирования» : МГУ, 2004. — 85 с.

21. Jahne Bernd. Digital Image Processing. — Springer-Verlag Berlin Heidelberg, 2005. — 608 p. — ISBN: [978-3-540-24035-8](#).
22. Gonzalez Rafael C., Woods Richard E. Digital Image Processing. — Pearson, 2007. — 976 p. — ISBN: [978-0131687288](#).
23. Brigham E. Oran. The fast Fourier transform and its applications. — Englewood Cliffs, N.J.: Prentice Hall, 1988. — 448 p. — ISBN: [0-13-307505-2](#).
24. Haddad R.A., Akansu A.N. A class of fast gaussian binomial filters for speech and image processing // IEEE Transactions on Acoustics, Speech and Signal Processing. — 1991. — Vol. 39. — P. 723–727.
25. Feichtinger Hans G., Strohmer Thomas. Gabor Analysis and Algorithms. — Birkhauser, 1994. — 496 p. — ISBN: [0-8176-3959-4](#).
26. Faber Georg. Uber die orthogonalfunktionen des herrn haar // Deutsche Math. — 1910. — Vol. 19. — P. 104–112.
27. Morlet wavelets in quantum mechanics // Quanta. — 2012. — Vol. 1. — P. 58–70.
28. Ballard Dana H. Generalizing the Hough transform to detect arbitrary shapes // Pattern regognition. — 1981. — Vol. 13, no. 2. — P. 111–122.
29. Vincent Rebeca, Folorunso Olusegun. A descriptive algorithm for sobel image edge detection // Proceedings of Informing Science and IT Education Conference (InSITE). — 2009. — Vol. 1. — P. 97–107.
30. Reuter Martin, Biasotti Silvia, Giorgi Daniela. Discrete laplace–beltrami operators for shape analysis and segmentation // Computers & Graphics. — 2009. — Vol. 33. — P. 381–390.
31. Gao Wenshuo, Yang Lei, Zhang Xiaoguang. Based on soft-threshold wavelet de-noising combining with prewitt operator edge detection algorithm // Education Technology and Computer (ICETC), 2010 2nd International Conference on. — 2010. — Vol. 5. — P. 155–162.
32. Canny John. A computational approach to edge detection // IEEE

Trans. Pattern Analysis and Machine Intelligence. — 1986. — Vol. 8. — P. 679–698.

33. Андреев А. Ю., Бобков С. П. Сегментация символов в изображении модифицированных методом жука // Инженерно-технические науки. — 2014. — Т. 1, № 37. — С. 85–88.
34. Speeded-up robust features (SURF) / Herbert Bay, Andreas Ess, Tinne Tuytelaars, Luc Van Gool // Computer Vision and Image Understanding. — 2008. — Vol. 110. — P. 346–359.
35. Lowe David. Method and apparatus for identifying scale invariant features in an image and use of same for locating an object in an image. — 2004. — 03. — URL: http://www.patentlens.net/patentlens/patent/US_6711293/.
36. Соколов Е. А. Семинары по линейным классификаторам, глава 5: Ядра и их применение в машинном обучении // ВМК МГУ. — 2013. — С. 1–9.
37. Likas Aristidis, Vlassis Nikos, Verbeek Jacob J. The global k-means clustering algorithm // Pattern Recognition. — 2003. — Vol. 36. — P. 451–461.
38. Rojas Raul. Adaboost and the super bowl of classifiers a tutorial introduction to adaptive boosting // Computer Science Department. — 2009.
39. Cortes Corinna, Vapnik Vladimir. Support-vector networks // Machine Learning. — 1995. — Vol. 20. — P. 273–297.
40. Новиков Ю. И., Громова Г. Я. ГОСТ 20999-83 Устройства числового программного управления для металлообрабатывающего оборудования. — 1983.
41. ISO 6983-1:2009 Numerical control of machines - Program format and definition of address words. — 1982.

Приложение А

Текст программы определения метки

Листинг А.1 — Программа определения метки на заготовке

```
1 # подключение модулей
2 import numpy as np
3 import argparse
4 import sys, cv2 as cv
5 import serial
6
7 def imgProcess(img): # обработка кадра
8     porog_value = 185
9
10    # полуточковый вид
11    gray = cv.cvtColor(img, cv.COLOR_BGR2GRAY)
12    gray = cv.GaussianBlur(gray, (7, 7), 1.5)
13
14    # градиентный фильтр
15    gradX = cv.Sobel(gray, ddepth = cv.CV_32F, dx = 1, dy = 0,
16                      ksize = -1)
17    gradY = cv.Sobel(gray, ddepth = cv.CV_32F, dx = 0, dy = 1,
18                      ksize = -1)
19
20    gradient = cv.subtract(gradX, gradY)
21    gradient = cv.convertScaleAbs(gradient)
22
23    blurred = cv.blur(gradient, (9, 9))
24    (_, thresh) = cv.threshold(blurred, porog_value, 255,
25                              cv.THRESH_BINARY)
26
27    # анализ контуров
28    (cnts, _) = cv.findContours(thresh.copy(),
29                                cv.RETR_EXTERNAL, cv.CHAIN_APPROX_SIMPLE)
```

```

30
31     return cnts
32
33 def centerOfImage(img): # центр кадра
34     cntrOfImg = [0,0]
35     width = img.shape[0]
36     height = img.shape[1]
37     cntrOfImg[0] = int(width / 2)
38     cntrOfImg[1] = int(height / 2)
39     return cntrOfImg
40
41 def moving(labelCenter):
42     center = centerOfImage(img)
43
44     ser = serial.Serial(
45         port='/dev/ttyUSB1', #порт связи с контроллером
46         baudrate=9600,
47         parity=serial.PARITY_ODD,
48         stopbits=serial.STOPBITS_TWO,
49         bytesize=serial.SEVENBITS
50     )
51
52     ser.isOpen()
53
54     if labelCenter[0] < center[0]: # сдвиг по X
55         while labelCenter[0] != center[0]:
56             ser.write('e') # вправо
57             ser.write('r')
58     elif labelCenter[0] > center[0]:
59         while labelCenter[0] != center[0]:
60             ser.write('q') # влево
61             ser.write('w')
62
63     if labelCenter[1] < center[1]: #сдвиг по Y
64         while labelCenter[1] != center[1]:

```

```

65             ser.write('t') # sleep
66             ser.write('y')
67     elif labelCenter[1] > center[1]:
68         while labelCenter[1] != center[1]:
69             ser.write('u') # вниз
70             ser.write('i')
71
72     ser.write('o') # cmap
73
74     ser.close()
75
76
77 cap = cv.VideoCapture(0)
78
79 #frameCount = 0
80
81 while True:
82     ok, img = cap.read() #npuiëm na sevod kadrä
83     if not ok:
84         break
85
86     cnts = imgProcess(img)
87
88     c = []
89     if (len(cnts) != 0):
90         c = sorted(cnts, key = cv.contourArea,
91                     reverse = True)[0]
92
93         # построение рамки
94     x,y,w,h = cv.boundingRect(c)
95
96         # если стороны равны, то зелёный, иначе красный
97     if (w/h <= 1.05) and (w/h >= 0.95):
98         cv.rectangle(img,(x,y),(x+w,y+h),(0,255,0),2)
99     else:

```

```
100         cv.rectangle(img,(x,y),(x+w,y+h),(0,0,255),2)
101
102         # центральная точка
103         centerpoint = (x+w/2, y+h/2)
104         radius = 1
105         cv.circle(img,centerpoint,radius,(255,255,255),2)
106
107
108     cv.imshow("image", img)
109
110     if cv.waitKey(30) > 0:
111         break
```

Приложение Б

Общий вид координатной платформы

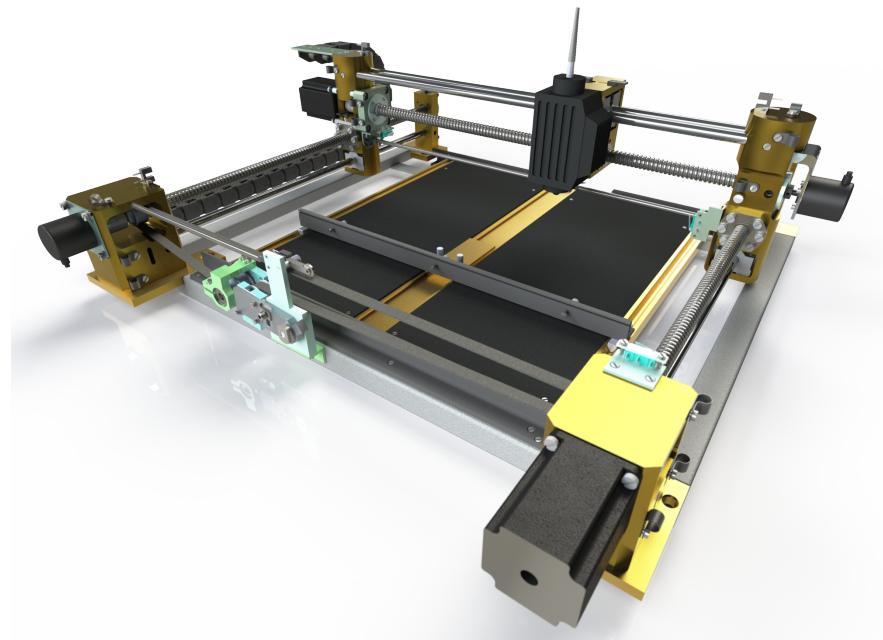


Рисунок Б.1 — Координатная платформа

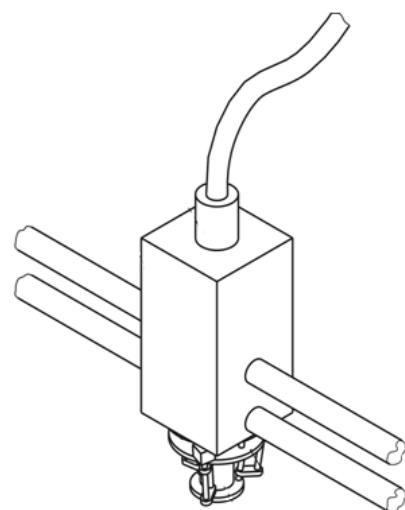


Рисунок Б.2 — Интеллектуальная лазерная головка

Приложение В

Применяемые технические средства



Рисунок В.1 — IP-камера

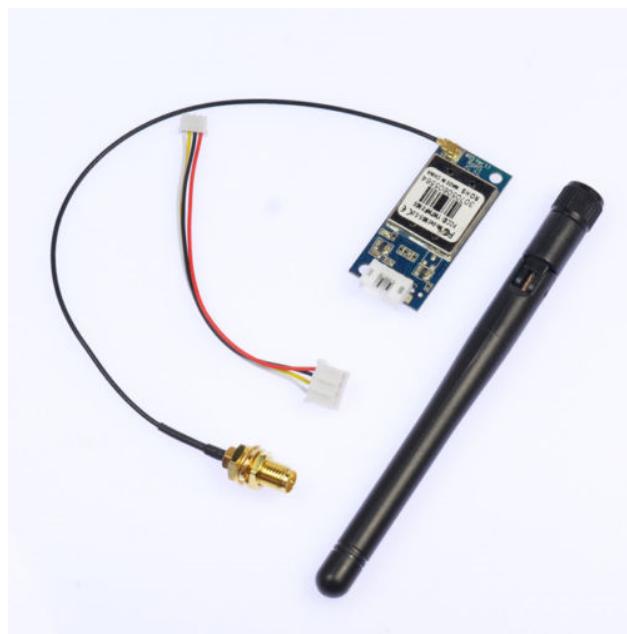


Рисунок В.2 — Компоненты интерфейса Wi-Fi камеры

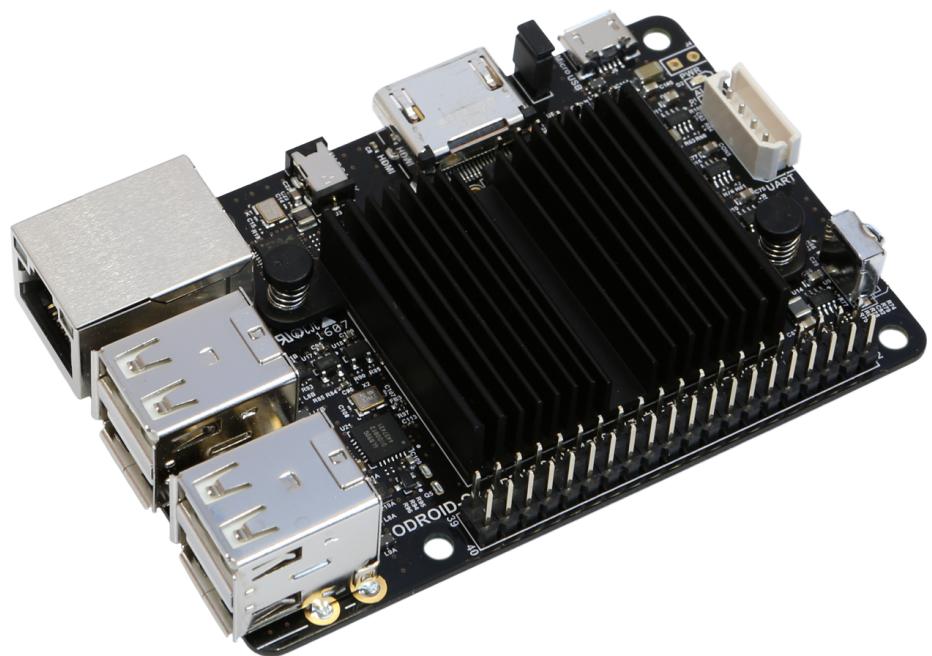


Рисунок В.3 – Одноплатный компьютер Odroid-C2