Федеральное государственное автономное образовательное учреждение

высшего образования

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

На правах рукописи

КАЛИТА ДИАНА ИВАНОВНА

«РАЗРАБОТКА ФИЛЬТРОВ ВЫСОКОЙ ЭФФЕКТИВНОСТИ ДЛЯ ОБЪЕКТОВ ЦИФРОВЫХ СИСТЕМ ВИДЕОНАБЛЮДЕНИЯ НА ОСНОВЕ СИСТЕМЫ ОСТАТОЧНЫХ КЛАССОВ»

Специальность: 05.13.18. – Математическое моделирование, численные методы и комплексы программ

ДИССЕРТАЦИЯ

на соискание ученой степени кандидата технических наук

Научный руководитель:

доктор технических наук,

профессор Червяков Н.И.

Ставрополь – 2017

Оглавление

Введение

Широкое и повсеместное применение цифровых систем видеонаблюдения приводит к необходимости постоянной обработки цифровых сигналов в виде цифрового изображения. Современные линии видеоконтроля строятся на новейших технических средствах, которые позволяют выполнять цифровую обработку получаемого видеоизображения [10]. Цифровое видеонаблюдение является эффективным, поскольку его работу можно связать с компьютерными системами. Одной из основных задач, которую решают цифровые системы видеонаблюдения, является задача обеспечения видеоконтроля наблюдаемого объекта. В системах видеоконтроля и контроля доступа процесс обработки и идентификации цифровых изображений должен быть максимально быстрым. Автономность объектов цифровых систем видеонаблюдения накладывает ограничения на размеры таких устройств и их энергоэффективность. По этой причине, происходит непрерывный поиск новых, более качественных и надежных средств цифровой обработки сигналов (ЦОС), а также совершенствование известных методов обработки [62].

С другой стороны, системы видеонаблюдения, работающие непосредственно с цифровыми сигналами в виде изображения, должны решать три основные задачи, возникающие в процессе функционирования системы. Эти задачи заключаются в обработке, хранении и анализе получаемого на входе цифрового сигнала [3]. Задача обработки сигнала требует применение эффективных цифровых фильтров в объектах цифровых систем, которые в свою очередь изменяют изображение, в соответствии с используемым алгоритмом обработки. Анализ цифрового сигнала в системах видеонаблюдения заключается в распознавании (идентификации), поступившего сигнала в виде изображения [33]. Таким образом, решение задач обработки изображений и его анализа тесно связаны между собой, так как применение перспективных методов обработки позволит улучшить результаты распознавания [13,35].

В свою очередь, решение обозначенных задач, стоящих перед объектами цифровых систем видеонаблюдения, накладывают определенные условия на их разработку. Возникает противоречие между необходимостью проектирования основных вычислительных структур объектов видеонаблюдения, позволяющих, с одной стороны, наилучшим образом преобразовать обрабатываемое изображение для дальнейшего распознавания образов, а с другой – свести к минимуму время, затраченное на их обработку в режиме реального времени [3,4]. Разрешением этого противоречия является параллельная организация вычисления, которая является перспективным инструментом в решении задачи минимизации времени, необходимого на обработку. Одним из способов обеспечивающих параллельное выполнение операций на арифметико-логическом уровне является использование непозиционных систем счисления вместо традиционной двоичной системы счисления, которая является позиционной системой счисления (ПСС) [26,67].

Большое количество публикаций отечественных и зарубежных исследователей за последнее время предлагают систему остаточных классов (СОК) в качестве перспективной замены позиционной системы счисления, для применения в ряде приложений, в том числе и в цифровой обработке сигналов [49,78,79,82,119]. Эффективность в применении СОК достигается за счет эффективного использования параллелизма и снижения разрядности операндов [53,79,83]. В [93,109] авторы предлагают использовать СОК в качестве основного инструмента в СНС для решения задачи распознавания образов. В предлагаемой архитектуре СНС авторы используют вложенную СОК из наборов модулей произвольного вида и просмотровые таблицы (LUT-таблицы), за счет чего увеличивается логическая глубина сети, что в свою очередь понижает скорость работы сети.

Изучением СОК занимаются многие исследователи по всему миру и применяют СОК при проектировании электронных устройств. Компания Cisco Technology использует СОК для создания некоторых видов электронных устройств (патент номер US 7,027,598 B1 от 11.04.2006). Advantest Corporation использует при проектировании устройств разработанную архитектуру,

способную выполнять многие операции в СОК (патент номер US 8,326,908 B2 от 4.12.2012). Kabushiki Kaisha использует СОК при проектировании полупроводниковой памяти с декодером Рида-Соломона (патент номер WO 2008/099723 A1 от 21.08.2008). Свои разработки в области СОК имеет Harris Corporation, который использует их для генерирования хаотических последовательностей большой длины (патент номер US 7,937,427 B2 от 3.05.2011). Отдельные элементы СОК имеются в патентах компаний Samsung Electronics (патент номер US 7,805,478 B2 от 28.09.2010), Google Inc. (патент номер US 8,386,802 B2 от 26.02.2013) и ZTE Corporation (патент номер EP 2 421 326 A1 от 22.02.2012).

Значительный научный вклад в теорию модулярных вычислений и их приложений внесли отечественные и зарубежные исследователи: И.Я. Акушский, Д.И. Юдицкий, В.М. Амербаев, А.А. Коляда, А.И. Галушкин, И.Т. Пак, М.В. Синьков, В.А. Торгашев, Н.И. Червяков, И.А. Калмыков, О.А. Финько, Д. Свобода, N. Szabo, M. Valach, H.L. Garner, A.S. Fraenkel, A. Huang, B. Purhami, W. Ienkns, H. Krisha, A. Omondi, A. Premkumar, I. Ramires, A. Curcik, L.Yang, D. Zhang, P Steffan, G. Pirlo, L. Sousa и другие.

Особую эффективность СОК показала в проектировании устройств цифровой обработки сигналов [80,112]. Как показано в [62], использование СОК в системах цифровой обработки изображений в реальном масштабе времени позволяет получить технический результат в виде повышения быстродействия выполнения арифметических операций по модулю при расчете разностного уравнения цифрового фильтра. Целесообразность применение СОК при решении задач цифровой обработки изображений подтверждается в [23], [66], [85] где использование СОК при построении КИХ-фильтров дает существенное повышение скорости по сравнению с двоичной системой счисления.

Таким образом, анализ научной литературы [78], [79], [82], [88], [102], [104], [119], [120], [131] показывает, что использование СОК позволяет достичь значительных преимуществ в использовании: снизить мощность устройств, по сравнению с традиционным двоичным кодированием, за счет совместного

использования с ПЛИС; снизить энергопотребление при проектировании КИХ-фильтров; повышение помехоустойчивости при кодировании информации; увеличить скорость работы при решении задач распознавания образов.

Однако, несмотря на эффективное применение при выполнении операций умножения, сложения и вычитания, выполнение ряда других операций на основе СОК является весьма затруднительным. К операциям такого вида относятся деление, обнаружение знака числа, сравнение чисел по величине. Перечисленные операции являются существенным ограничивающим фактором, сказывающимся, в первую очередь, на скорости разрабатываемых устройств.

Целью диссертационного исследования является разработка математических моделей, методов и алгоритмов реализации повышения скорости работы цифровых фильтров обработки изображений систем видеонаблюдения на основе интеграции системы остаточных классов и искусственных нейронных сетей.

Объект исследования – цифровые системы видеонаблюдения.

Предмет исследования – математические методы, модели и алгоритмы вычислительных средств объектов цифровой обработки изображений.

Научная задача – разработка новых математических моделей цифровых фильтров изображений, численных методах вычисления немодульных операций, а также комплекса программ, применение которых позволит увеличить скорость работы системы.

Для решения поставленной общей научной задачи была произведена ее декомпозиция на ряд частных задач:

1. Анализ математических моделей и алгоритмов вычислительных процедур, используемых в системах цифровой обработки изображений, а также особенностей реализации в модулярном базисе.

2. Разработка и модификация численного метода для процесса цифровой фильтрации изображений на основе вычислений в СОК. Модификация архитектуры процесса фильтрации изображений.

3. Разработка математической модели цифрового фильтра, с высокой скоростью работы за счет использования оптимального набора модулей в системе остаточных классов.

4. Разработка математической модели ошибок, возникающих при цифровой обработки изображений.

5. Разработка алгоритма кратномасштабного анализа сигналов в СОК с использованием коэффициентов фильтра малой разрядности.

6. Разработка модели и архитектуры СНС, основанной на вычислениях в СОК. Применение набора модулей специального вида для модификации существующих архитектур.

7. Оценка эффективности разработанной архитектуры СНС на основе результатов вычислительного эксперимента.

8. Создание комплекса программ для экспериментального исследования разработанной СНС на основе СОК.

Методы исследования включают в себя использование математического аппарата высшей алгебры, теории чисел, теории алгоритмов, численных методов, теории модулярной арифметики, математического моделирования и системного анализа.

На защиту выносятся следующие научные результаты:

1. Математические модели и алгоритмы вычислительных процедур, используемых в системах цифровой обработки изображений при решении задач распознавания образов.

2. Численный метод повышения производительности цифровых фильтров на основе интеграции применения СОК и СНС. Модифицированная архитектура фильтрации изображений.

3. Математическая модель цифрового фильтра, с высокой скоростью работы за счет использования оптимального набора модулей в системе остаточных классов.

4. Предложенный критерий определения достаточности динамического диапазона СОК для цифровой обработки изображений.

5. Алгоритм кратномасштабного анализа сигналов в СОК с использованием коэффициентов фильтра малой разрядности.

6. Архитектура СНС с вычислениями в СОК с модулями специального вида и ее использование для реализации задачи распознавания образов.

7. Программный комплекс для моделирования процесса распознавания изображений с применением СНС и вычислений в СОК и оценка эффективности разработанных моделей, методов и алгоритмов на основе результатов компьютерного моделирования.

8. Комплекс программ моделирования разработанных на языке С++, позволяющая моделировать процесс цифровой фильтрации в СОК с модулями специального вида.

Научная новизна:

1. Разработан метод повышения производительности цифровых фильтров на основе использования набора модулей системы остаточных классов вида $\{2^n, 2^n-1, \ldots, 2^{n+k}-1\}$.

2. Предложен численный метод выполнения операции цифровой фильтрации, основанный на вычислениях в СОК, позволяющий сократить временные затраты в ходе цифровой обработки изображений за счет использования модулей специального вида.

3. Разработана математическая модель коррекции ошибок при цифровой обработки сигналов в СОК, которая позволяет не только получать безошибочно обработанное цифровое изображение, но и рассчитывать вероятность ошибок при обработки.

4. Разработан алгоритм кратномасштабного анализа сигналов СОК, позволивший увеличить скорость вычислений на интегральных схемах за счет использования коэффициентов фильтра малой разрядности.

5. Модифицирована архитектура глубокой СНС с вычислениями в системе остаточных классов с модулями специального вида, позволившая сократить временные и аппаратурные затраты.

6. Разработана архитектура СНС из четырех оптимизированных слоев, реализующая задачу распознавания образов.

7. Разработан программный комплекс, позволяющий производить моделирование процесса распознавания изображений на основе разработанной СНС.

8. Разработан программный комплекс, реализующий разработанный метод цифровой фильтрации с модулями специального вида.

Достоверность результатов обеспечивается корректным и обоснованным применением методов математического моделирования и строгостью проводимых математических доказательств, а также результатами проведенного математического моделирования на базе FPGA фирмы Xilinx.

Практическая значимость результатов исследования состоит в возможности реализации части функциональных устройств, используемых в объектах цифровых систем видеонаблюдения, основанных на СОК, на базе разработанных методов, что способствует повышению скорости работы систем. Разработанный программный продукт и аппаратные решения, зарегистрированные в соответствующем порядке, способствуют оптимизации эксплуатационных возможностей цифровых систем видеонаблюдения.

Внедрение. Результаты диссертационного исследования используются в учебном процессе в СКФУ на кафедре прикладной математики и математического моделирования в дисциплинах «Основы цифровой схемотехники», «Приложения системы остаточных классов в информационных технологиях», «Основы модулярной арифметики», «Моделирование СБИС на языке VHDL», а также при разработке практических занятий и лабораторных работ по курсу «Математические методы обработки изображений», что подтверждено Актом об использовании результатов работы в учебном процессе. Основные научные результаты использованы в опытно-конструкторских разработках ООО «Медицина-ИТ», что подтверждается Актом о внедрении результатов диссертационной работы. Кроме того ряд результатов работы был использован при выполнении научно-исследовательских работ в базовой части

государственного задания СКФУ №2653 «Проблемы интеграции параллельной компьютерной алгебры и искусственных нейронных сетей в области информационно-телекоммуникационных технологий».

Апробация работы. Основные результаты работы были представлены на 15-ом международном Симпозиуме по проблемам избыточности REDUNDANCY-2016 (г. Санкт-Петербург, 2016 г.), Международной научно-практической конференции молодых ученых «Морально-этические аспекты и темпорально-экологические императивы инвенционного процесса генерации новых научно-технических знаний» (г. Ставрополь, 2014 г.), 4-ой Международной научно-практической конференции «Инновации, качество и сервис в технике и технологиях» (г. Курск, 2014 г.), 6-ой Международной научно-технической конференции «Инфокоммуникационные технологии в науке, производстве и образовании» (г. Ставрополь, 2014 г.), Международной научно-практической конференции «Наука, образование, общество: проблемы и перспективы развития» (г. Тамбов, 2014 г.), 1-ой Международной научной конференции «Параллельная компьютерная алгебра и ее приложения в новых инфокоммуникацинных системах» (г. Ставрополь, 2014 г.), 11-ой Международной заочной научной конференции «Основные направления развития научного потенциала в свете современных исследований: теория и практика» (г. Ставрополь, 2017 г.).

Публикации по теме диссертации. Основные результаты исследования отражены в 14 работах, среди которых 5 статей в научных изданиях, входящих в перечень ВАК Министерства образования и науки, а также 2 статьи входящие в систему индексирования научных работ Scopus и Web Of Science.

Личный вклад соискателя. Все изложенные в работе результаты исследований получены при непосредственном участии автора. Авторским вкладом являются разработка методов, моделей и алгоритмов, применяемых в цифровых фильтрах, разработка численного метода выполнения цифровой фильтрации на основе СОК, разработка программного комплекса моделирования цифровой фильтрации в СОК с модулями специального вида, разработка архитектуры сверточной нейронной сети с вычислениями в СОК с модулями

специального вида, разработка программного комплекса среды моделирования распознавания изображений с использованием сверточных нейронных сетей.

Структура диссертации. Диссертационная работа состоит из введения, 4-х глав, заключения, списка сокращений и обозначений и списка использованной литературы, содержащей 133 наименование. Основная часть работы содержит 208 страниц машинописного текста. Работа содержит 84 рисунка, 34 таблицы и 3 приложения. В диссертации принята двойная нумерация формул, рисунков и таблиц: первая цифра указывает номер главы, вторая – порядковый номер рисунка, таблицы или формулы внутри данной главы.

Краткое содержание работы.

Во введении обоснована актуальность темы диссертации, сформулированы цель и задачи работы, выбраны объект и предмет исследования, показана научная новизна, практическая и теоретическая ценность полученных результатов, приведены основные положения, выносимые на защиту.

В первой главе представлен анализ научно-технической литературы, посвященной существующим математическим моделям, методам и алгоритмам цифровой обработки изображений. Отмечены основные задачи, которые должны решать цифровые системы видеонаблюдения. Основной задачей, которую решают системы видеонаблюдения является задача распознавания образов напрямую связанная с решением задачи цифровой обработки изображений. В качестве основного инструмента для решения задач цифровой обработки изображений выбраны цифровые фильтры. Показывается, что цифровые фильтры обеспечивают наилучшую надежность, точность и простоту использования. В качестве основного инструмента для решения задачи распознавания образов выбраны сверточные нейронные сети, поскольку они способны эмитировать процесс человеческого зрения и способны к обучению. Отмечены основные особенности проектирования систем видеонаблюдения, которые состоят в увеличении скорости обрабатываемых данных в масштабе реального времени. Отмечены основные особенности разработки фильтров высокой эффективности, которые применяются для решения задач цифровой обработки изображений

(ЦОИ). Обоснована целесообразность применения непозиционной системы счисления СОК в решении задач ЦОИ, поскольку модулярная арифметика обладает большим потенциалом для увеличения производительности вычислительных систем за счет естественного арифметического параллелизма. Сформулирована научная задача диссертационной работы и проведена ее декомпозиция на частные подзадачи.

Во второй главе приведены и разработаны основные модели и алгоритмы для процесса цифровой обработки изображений, основанные на вычислениях в СОК. Разработана архитектура КИХ-фильтра на основе вычислений в СОК. Проведен анализ производительности разработанного фильтра с использованием различных модулей. Выявлена наилучшая модель набора модулей СОК. Разработанная архитектура КИХ-фильтра, использующая наилучший набор модулей позволила увеличить скорость работы фильтра на 93% по сравнению с известными аналогами.

Проведено подробное исследование предложенных архитектур для процесса фильтрации цифрового изображения. Установлен вариант модифицирования известных архитектур, с целью получения корректных результатов обработки изображений. Приведен новый численный метод решения задачи фильтрации изображений на основе вычислений в СОК. Установлено, что данный метод позволит сократить временные затраты связанные с выполнением арифметических вычислений в ходе процесса цифровой фильтрации, а также способен увеличивает скорость работы цифрового объекта.

Проведено подробное исследование вопроса выбора достаточного динамического диапазона, необходимого для процесса цифровой обработки изображений. Предложен критерий определения достаточного динамического диапазона для СОК. Полученный критерий позволяет получать безошибочно обработанные изображения.

Исследован вопрос о влиянии снижения разрядности коэффициентов фильтра при вейвлет-обработке изображений в оттенках серого. Показано, что снижение разрядности коэффициентов вейвлетного фильтра ведет к снижению

качества получаемого изображения при обработке. Установлено, что пригодное на практике изображение можно получить при использовании 12 и более бит представления разрядности коэффициентов, а визуально неотличимое при использовании 28 и более бит разрядности коэффициентов фильтра. Применение данного результата позволяет использовать разрядность данных, меньшую, чем 64 или 32 бита, что приводит к существенному снижению аппаратных затрат на реализацию данного алгоритма.

В третьей главе разработана архитектура СНС на основе вычислений в СОК с использованием модулей специального вида. Проведено подробное исследование известных архитектур СНС, позволившее выявить существенные недостатки в них. На основе исследования, разработана новая архитектура СНС, которая использует в своей конструкции модифицированные модулярные сумматоры и блоки прямого и обратного преобразований. Данная архитектура позволяет сократить скорость выполнения этих операций за счет оптимизации вычислений по модулям специального вида, а также требует меньше аппаратных затрат по сравнению с известными архитектурами.

В четвертой главе разработана СНС для распознавания образов, которая состоит из четырех слоев и использует фильтры Собеля для извлечения признаков из изображений. На основе разработанной архитектуры проведено обучение СНС. Проведено моделирование с целью оценки временных затрат работы СНС, основанных на различных конфигурациях сети (разные размеры масок фильтров, шаг вычислений, разрешение входного изображения), а также с целью определения оптимальной аппаратной конфигурации (разное количеств фильтров для первого и третьего слоя, разрядность дробной части). Установлено, что для разработанной архитектуры СНС из четырех слоев оптимальной будет сеть, использующая 8 фильтров для операции свертки, 10 нейронов третьего слоя, 16 разрядов дробной части, разрешение 256×192. Разработанная сеть позволяет сократить временные и аппаратные затраты в блоков преобразования из ПСС в СОК и из СОК в ПСС.

В заключении подведены итоги и обобщены результаты проведенных исследований.

В приложениях приведен листинг разработанной программы на языке C++, позволяющей моделировать процесс цифровой фильтрации с вычислениями в системе остаточных классов с модулями специального вида; листинг программы в Matlab, позволяющей моделировать процесс распознавания изображений с использованием сверточных нейронных сетей; свидетельство о государственной регистрации программы для ЭВМ.

Автор выражает искреннюю благодарность научному руководителю – доктору технических наук, профессору, заслуженному деятелю науки и техники РФ, академику МАИ, создателю и руководителю научной школы «Нейроматематика, модулярные нейрокомпьютеры и высокопроизводительные вычисления», почетному профессору Северо-Кавказского федерального университета, Николаю Ивановичу Червякову, а также кандидату физико-математических наук, доценту кафедры прикладной математики и математического моделирования Северо-Кавказского федерального университета, Павлу Алексеевичу Ляхову за помощь, оказанную при написании диссертации, и критические замечания, высказанные при ее обсуждении.

ГЛАВА 1. АНАЛИТИЧЕСКИЙ ОБЗОР МОДЕЛЕЙ ЦИФРОВОЙ ОБРАБОТКИ ИЗОБРАЖЕНИЙ

1.1 Анализ архитектур систем видеонаблюдения

Современные системы видеонаблюдения решают определенный круг задач, согласно цели их эксплуатации. Наиболее востребованными задачами, которыми решают системы видеонаблюдения, является видеоконтроль за объектом и анализ информации, полученной в ходе работы системы [3,4]. Анализ информации чаще всего происходит с целью распознавания объектов с полученных видеоизображений. Другими словами, система видеонаблюдения – это телевизионная система замкнутого типа, для получения телевизионных изображений.

Таким образом, главным преимуществом применения систем видеонаблюдения является получение, обработка и регистрация текущей видеоинформации всего объекта, в реальном масштабе времени, по заданному алгоритму событий [3,14]. Так как видеоинформация должна поступать в реальном масштабе времени, с максимально высокой скоростью, то эти условия накладывают определенные ограничения на разработку архитектуры видеосистемы.

Классификация видеосистем не носит строго нормативного характера, но существуют некоторые практические подходы по их разделению в зависимости от основного предназначения и видов наблюдения [21]. Такой подход также определяет на этапе задания или проектирования системы видеонаблюдения ее состав и архитектуру.

Качество видеоинформации определяется, прежде всего, видеокамерой, представляющей собой законченное устройство, которое при подключении к видеовходу монитора или телевизора позволяет наблюдать изображение на экране на значительном расстоянии от объекта съемки. Поэтому, одним из видов

классификации систем видеонаблюдения является их разделение по видам используемых камер, устройств контроля и регистрации [10]:

– Аналоговые системы видеонаблюдения – используются обычно в тех случаях, когда необходимо вести запись на видеокассеты. Такие системы строятся на базе мультиплексоров и спецвидеомагнитофонов с функцией длительной записи. Под мультиплексором понимается прибор, который обрабатывает видеоизображение, получаемое от видеокамеры, анализирует изображение и передает их в заданном формате на видеомонитор [4].

– Цифровые системы видеонаблюдения – имеют более гибкие возможности по работе с видео и аудио информацией и более просты в управлении. Основным компонентом цифровых систем видеонаблюдения является цифровой видеорегистратор – Digital Video Recorder (DVR). К видеорегистратору подключаются аналоговые видеокамеры, оцифровка видеосигнала происходит внутри видеорегистратора [3].

– Компьютерные системы видеонаблюдения – строятся на базе компьютеров. Функциональность таких систем определяется программным обеспечением, поставляемым с платой захвата видеосигнала [39].

– IP системы видеонаблюдения – строятся на специальных видеокамерах, имеющих встроенную сетевую плату для подключения к локальной вычислительной сети. Оцифровка и сжатие видеосигнала происходит внутри самой камеры [4].

Рассмотрим более подробно основные различия между цифровыми и аналоговыми системами видеонаблюдения.

В основе аналоговой системы видеонаблюдения лежит использование аналоговых средств мониторинга и записи видеосигнала с помощью специализированных устройств – видеомагнитофонов, мультиплексоров, мониторов и т.д [3,4]. Аналоговые системы отличаются невысоким показателем скорости и качества записываемого изображения. Кроме того, такие системы

имеют низкую функциональность, высокие эксплуатационные расходы, быстрый износ кассет и т.д.

В основе цифровых систем лежит преобразование аналогового видеосигнала, поступающего с видеокамеры в цифровой вид для его последующей обработки, записи, хранения и отображения. Цифровая система, осуществляющая процесс преобразования, основана на базе персональных (или серверных) компьютеров либо на базе устройства, внешне похожего на видеомагнитофон, в основе которого лежит компьютерная архитектура [33]. Основой всех цифровых систем видеонаблюдения является набор методов и алгоритмов, с помощью которых изображение, поступающее с камеры на компьютер, оцифровывается и записывается на жесткий диск, передается по цифровым каналам связи для текущего и последующего анализа произошедших на объекте событий. Особенностью рассматриваемых систем является оцифровка изображения [3]. Под оцифровкой понимается преобразование аналогового сигнала, поступающего с камеры на компьютер, в цифровой вид для последующей обработки цифровой системой. Этот процесс осуществляют сетевые видеосерверы. Принцип работы видеосервра состоит в следующем: на вход видеосервера поступают сигналы от аналоговой видеокамеры, после чего поступивший сигнал оцифровывается и сжимается [39]. Далее все входящие видеосигналы обрабатываются цифровым сигнальным процессором и синхронизируются в единую последовательность кадров. При этом, используемый в видеосервере алгоритм сжатия определяет качество видеоизображения и влияет на скорость передачи видео в сети [115]. Для получения оцифрованного потока с хорошей полосой пропускания используются алгоритмы сжатия, основанные на дискретном косинусном или вейвлет-преобразовании. Видеосерверы, сжимающие видео таким способом, обеспечивают гораздо больший коэффициент сжатия, чем форматы JPEG или MJPEG, обеспечивая при этом достаточно высокое качество изображения при относительно небольшом объеме передаваемых данных [39].

Однако, применяемые алгоритмы сжатия обладают и рядом недостатков, которые в свою очередь снижают скорость передачи данных, а также влияют на качество изображений, подверженных сжатию [65,115]. По этим причинам актуальным становится вопрос поиска новых математических алгоритмов, которые базировались бы на известных методах сжатия, но давали бы наилучший результат в практическом применении.

К основным преимуществам цифровых систем видеонаблюдения относятся высокое соотношение параметра качество/скорость записи, наличие высокой функциональности, интеграционные возможности, низкие эксплуатационные расходы [3,21]. Недостатками являются высокие требования к качеству видеосигнала, исходящего от видеокамеры. Кроме того, на качество сигнала влияет даже маленькая ошибка, которая может привести к искажению всего изображения.

Сравнивая между собой цифровые и аналоговые системы видеонаблюдения можно сказать, что аналоговые заметно отстают по своим характеристикам, что делает цифровые системы видеонаблюдения гораздо предпочтительными в использовании. В таблице 1.1 приведена сравнительная характеристика по основным показателям аналогового и цифрового видеонаблюдения [3,45].

Таким образом, как видно из таблицы преимущества цифровых систем очевидны, что делает их гораздо предпочтительными в использовании.

Перед тем как обрабатываться, все поступившие видеосигналы представлены в аналоговом формате. Для дальнейшей работы необходимо, чтобы видеосигнал был оцифрован. Видеосервер представляет собой устройство, предназначенное для работы в составе аналогово-цифровой системы видеонаблюдения и преобразования аналогового видеосигнала в цифровой формат для последующей передачи его по компьютерной сети или записи на цифровой носитель информации [33].

Таблица 1.1. Сравнение цифровых и аналоговых систем видеонаблюдения

Характеристика	Цифровые системы видеонаблюдения	Аналоговые системы видеонаблюдения
Метод записи	Цифровой	Аналоговый
Носитель информации	HDD	Видео кассеты
Время записи	Ограничено объемом HDD	Ограничено одной видеокассетой
Качество изображения	Отличное качество с высоким разрешением. Не ухудшается со временем	Низкое качество, шумы. Ухудшение качества при многократном использовании лент
Поиск	Способность искать в многочисленных местах хранения информации с различными параметрами поиска. Мгновенный поиск	Трудоемкий поиск и проблемы с извлечением видео
Запись	Возможность выборочной записи с различным способом	Неэффективная, постоянная запись
Удаленный доступ	PSTN/ISDN/ADSL/LAN	Нет
Детектор движения	Возможность записи по движению. Встроенная функция	Нет
Статичная картинка	Отличное качество	Шумы или искажение изображения
Возможности разделения экрана	Различные варианты разделения экрана	Только с дополнительным устройством
Стоимость эксплуатации	Не требуется дополнительных затрат	Замена, хранение и обслуживание видеокассет
Периферийные устройства	Не требуется дополнительных устройств	Квадраторы, мультиплексоры, свитчеры и т.д.
Разрешение	Высокое и очень высокое разрешение изображения (от 1 Мп до 10 Мп – качество HDTV – и выше) в диапазоне от 640×480 до 2560×2048 и выше.	Максимум 0,4Мп в диапазоне 720×576 или 720×480.

Выше было отмечено, что одной из самых важных функций, которую должна выполнять цифровая система видеонаблюдения является функция распознавания объектов (образов). В свою очередь, система цифрового видеонаблюдения не всегда дает возможность получать видеоинформацию, которая будет визуально хорошего качества. Получаемая видеоинформация чаще всего подвергается каким-либо внешним или внутренним воздействиям. Примером внешних воздействий, которые могут ухудшить качество получаемой

видеоинформации является климатические условия работы системы, светочувствительность, время съемки, напряжение питания и т.д. Внутреннее воздействие на цифровую информацию напрямую зависит от аппаратных возможностей ее обработки, а именно от цифровых систем обработки. Цифровые системы обработки включают весь математический аппарат, который используется при решении задач цифровой обработки информации, в частности изображений [33,39]. Для получения цифровых изображений визуально хорошего качества, которые будут максимально приближены к оригиналу необходимо применения методов цифровой обработки сигналов (фильтрации изображений) [5,11]. Фильтрация изображений требует большого количества аппаратных и временных затрат, что приводит к снижению скорости обработки данных в системе, а также увеличению затрат на энергопотребление. Однако, вне зависимости от основных целей, для которых используются цифровые системы видеонаблюдения, информация, полученная на их основе должна обрабатываться максимально быстро. Процесс обработки включает цифровую фильтрацию, а также процесс распознавания образов. Процесс фильтрации изображений состоит из 90% арифметических операций (сложения и умножения) [11]. Поскольку цифровая система видеонаблюдения должна функционировать на достаточно высокой скорости в режиме реального времени, то эту скорость можно увеличить за счет применения новых методов обработки. Распараллеливание процесса обработки является одним из перспективных решений данной задачи, а подходящим инструментом в решении этой задачи является применение системы счисления, обладающей свойством параллелизма. Система остаточных классов помимо наличия свойства высокого параллелизма поможет не только увеличить скорость работы всего приложения, но и понизить затраты на энергопотребление [79,126,128,133].

Таким образом, описанные требования, предъявляемые к любой системе видеонаблюдения, подтверждают, что основным ядром работы всей системы в целом являются ее цифровые системы обработки.

Однако, анализ цифровых систем видеонаблюдения показал наличие противоречия между практическим и научным решением задачи. Возникает необходимость разработки моделей основных вычислительных узлов цифровой системы обработки, которые смогут работать в реальном масштабе времени на высокой скорости. Но в свою очередь, цифровая обработка изображений требует выполнения большого количества операций, что может привести к снижению скорости работы системы. Решением этого противоречия является применение системы остаточных классов.

1.2 Анализ проблем цифровой обработки изображений

Увеличение интереса к использованию цифровых систем видеонаблюдения влечет за собой необходимость в обработке сигналов в виде видео и изображений [3]. При этом обработанное изображение должно обладать такими качествами, благодаря которым его восприятие человеком было бы максимально комфортным. Часто возникает необходимость подчеркнуть, усилить какие-то черты, особенности, нюансы наблюдаемой картины с целью улучшения ее субъективного восприятия [33].

Однако, субъективность восприятия сильно усложняет применение формализованного подхода в достижении данных целей. Поэтому при обработке изображений для визуализации получили распространение методы, в которых часто отсутствуют строгие математические критерии оптимальности. Их заменяют качественные представления о целесообразности той или иной обработки, опирающиеся на субъективные оценки результатов [11].

Таким образом, одной из основных задач, которую необходимо решать при цифровой обработке изображения и, которая требует особого внимания, является задача улучшения визуального качества изображения. Однако, следует отметить, что цифровая обработка изображений подразумевает не только обработку в понимании улучшения визуального восприятия изображения, но и классификацию объектов, выполняемых при анализе изображений [11,18,19]. Из

этого следует актуальность решения еще одной задачи цифровой обработки изображений – задачи распознавания образов.

Рассматривая решение задачи для цифрового изображения в контексте повышения уровня его визуального качества, следует сказать, что решение этой задачи может быть достигнуто за счет решения частных подзадач:

1. Улучшение визуального качества изображения путем поэлементного преобразования.
2. Улучшение визуального качества изображения за счет фильтрации изображения.
3. Улучшение визуального качества изображение путем сжатия изображения.

Решение первой указанной подзадачи связано с улучшением согласования динамического диапазона изображения и экрана, на котором выполняется визуализация. Как правило, для цифрового представления каждого отсчета изображения отводится 1 байт (8 бит) запоминающего устройства, а входной и выходной сигнал при этом могут принимать одно из 256 значений. Обычно в качестве рабочего используется диапазон $[0,255]$, при этом значение 0 соответствует при визуализации черному цвету, а значение 255 – белому. Если параметры минимальной и максимальной яркости исходного изображения ($x_{\min}$ и $x_{\max}$ соответственно) существенно отличаются от граничных значений яркостного диапазона, то визуализированная картина выглядит как ненасыщенная, неудобная, утомляющая при наблюдении [11]].

Обычно изображения, сформированные системами видеонаблюдения, искажаются действиями помех, что в свою очередь затрудняет визуальный анализ человеком и автоматическую обработку на ЭВМ [18]. При этом помехами могут быть те или иные компоненты самого изображения. За счет применения процесса фильтрации происходит ослабление действия помех. Поскольку, фильтрация является процедурой обработки всего изображения в целом и происходит она по средствам цифрового фильтра, то в связи с этим возникает задача правильного

выбора такой модели цифрового фильтра или другими словами вычислительной процедуры, чтобы при этой обработке достигался наилучший результат [36,44,57]. Таким образом, решение второй указанной подзадачи сводится к выбору цифрового фильтра и обработке им цифрового изображения.

Чаще всего при записи полученной информации с объектов видеонаблюдения при недостаточном объеме памяти, в частности изображений, возникает существенная проблема – сохранение качества изображения [65]. Данная проблема возникает, как правило, при применении неподходящего алгоритма сжатия изображения, который ведет к большим потерям информации, что в свою очередь также ухудшает визуальное качество результата обработки [39].

В связи с вышесказанным, идет активный поиск новых качественных и надежных средств и методов цифровой обработка сигналов (ЦОС) и, как следствие, цифровой обработки изображений (ЦОИ), которые могут быть применены для объектов в цифровых системах видеонаблюдения.

1.3 Анализ методов цифровой обработки изображений

Интерес к методам ЦОИ произрастает из двух основных областей ее применения, которыми являются повышение качества изображений для улучшения его визуального восприятия человеком и обработка изображений для их хранения, передачи и представления в автономных системах машинного зрения [18].

Все основное множество методов, которые решают задачи ЦОИ с целью улучшения изображения, делится на методы обработки в частотной (частотные методы) и пространственной (пространственные методы) областях. Методы обработки в частотной области основываются на модификации сигнала, формируемого путем применения к изображению преобразования Фурье [7,8]. Однако, несмотря на то, что способы обработки в частотной области достаточно развиты, при решении практических задач они применяются реже, в связи с тем,

что требуют значительных вычислительных затрат. Чаще всего используются пространственные методы обработки изображений, наиболее известными из которых являются [11]:

- Разностные методы;
- Ранговые алгоритмы;
- Гистограммные методы;
- Метод локальных контрастов;
- Координатный метод анализа изображения;
- Спектральный метод анализа изображения.

Существенным преимуществом указанных методов является возможность быстрой обработки изображений в масштабе реального времени. Основные недостатки заключаются в ограниченности функциональных возможностей и недостаточной эффективности.

Суть пространственного метода заключается в работе непосредственно с плоскостью изображения, а именно со значениями пикселей, которые это изображение составляют. Процессы пространственной обработки описываются уравнением:

$$g(x,y)=T\left[f(x,y)\right] \qquad (1.1.)$$

где $f(x,y)$ – входное изображение, $g(x,y)$ – обработанное изображение, T – оператор над f, определенный в некоторой окрестности точки (x,y). Оператор T выполняется в каждой точке (x,y), давая в результате выходное значение g для данной точки. Кроме того, T может оперировать над последовательностью входных изображений, например, выполняя поэлементное суммирование. Поскольку результат улучшения каждого элемента изображения зависит только от яркости этого же элемента, по этой причине методы данной категории относятся к процедурам поэлементной обработки [11,18].

Одним из основных подходов в поэлементной обработке изображения является подход на основе использования масок (или другими словами, фильтров, ядра, шаблонов и т.д.) [32]. Маска представляет собой небольшой двумерный

массив, состоящий из значений коэффициентов. Коэффициенты маски определяют суть процесса обработки изображения. Соответственно методы на основе такого подхода относятся к методам обработке по маске или фильтрации по маске. Многообразие масок фильтров и правил фильтрации складывается из многообразия задач, которые необходимо решить, обработав изображение. В соответствии с этим различают [11,75]:

- Линейные фильтры;
- Сглаживающие фильтры;
- Контрастоповышающие фильтры;
- Разностные фильтры;
- Фильтры двумерной циклической свертки;
- Нелинейные фильтры;
- Пороговая фильтрация;
- Медианная фильтрация;
- Фильтры экстремумов.

Более подробная информация, касающаяся обработки изображений вышеуказанными способами фильтрации дается в [11].

1.4 Анализ методов распознавания образов

Ранее было отмечено, что задача распознавания образов является актуальной задачей в рассматриваемой области ЦОИ. Необходимость в таком распознавании помимо систем видеонаблюдения, возникает в самых разных областях – от военного дела и систем безопасности до оцифровки аналоговых сигналов [38,54,97]. Следует отметить, что рассматриваемая задача распознавания образов включает в себя проблемы визуализации изображений, рассмотренные в пункте 1.2.

Задача распознавания в ЦОИ сводится к принятию решения относительно полученного изображения, т.е. его классификации. Под классификацией

понимается процесс назначения определенного признака (метки) класса объектам, согласно некоторому описанию свойств этих объектов [11]. Таким образом, распознавание образов можно определить, как отнесение исходных данных к определенному классу с помощью выделения существенных признаков или свойств, характеризующие эти данные из общей массы несущественных деталей.

Поскольку в системах видеонаблюдения исходным материалом является полученное с камеры изображение, тогда задачу распознавания образов можно сформулировать как получение векторов признаков для каждого класса на рассматриваемом изображении. Тогда для решения обозначенной задачи сперва необходимо, провести процесс кодирования, заключающийся в присвоении значения каждому признаку из пространства признаков для каждого класса, а затем выделить характерные признаки или свойства из исходных изображений. Признак должен представлять из себя характерное свойство конкретного класса, при этом общие для класса. Признаки общие для всех классов не несут полезной информации и не рассматриваются в задаче распознавания. Выбор признаков является одной из важных задач, связанных с построением системы распознавания [11],[18].

Решение задачи распознавания образов, которая связанна с предварительной обработкой изображения, выделением признаков и получением оптимального решения и классификации происходит так или иначе на основе сравнения объектов. Сравнение объектов можно производить на основе их представления в виде векторов измерений, записанных на основе вещественных чисел. Тогда сходство векторов признаков двух объектов может быть описано с помощью евклидова расстояния.

$$\|x_1 - x_2\| = \sqrt{\sum_{i=1,d} \left(x_1[i] - x_2[i]\right)^2} \tag{1.2}$$

где d – размерность вектора признака.

Следует отметить, что процесс сравнения происходит по выбранному заранее методу распознавания образов. В целом, можно выделить три метода распознавания образов [37]:

– Сравнение с образцом. В эту группу входит классификация по ближайшему среднему, классификация по расстоянию до ближайшего соседа. Также в группу сравнения с образцом можно отнести структурные методы распознавания.

– Статистические методы. Эти методы используют статистическую информацию при решении задачи распознавания. Метод определяет принадлежность объектов к конкретному классу на основе вероятности. В ряде случаев это сводится к определению апостериорной вероятности принадлежности объекта к определенному классу, при условии, что признаки этого объекта приняли соответствующие значения.

– Нейронные сети. Отдельный класс методов распознавания. Отличительной особенностью от других методов является способность обучаться. Метод распознавания на основе нейронных сетей требует либо большого количества примеров задачи распознавания при обучении, либо специальной структуры нейронной сети, учитывающей специфику данной задачи. Тем не менее, по сравнению с другими методами, его отличает более высокая эффективность и производительность.

Последний метод является наиболее востребованным в данной области. Популярность в применение нейронных сетей связана с тем, что по своей структуре они подобны структуре головного мозга, но при этом представляют собой математическую модель параллельных вычислений, на основе простых процессорных элементов [70]. При построении нейронной сети не требуется заранее знать обо всех закономерностях исследуемой области, но необходимо располагать достаточным количеством примеров для настройки разрабатываемой системы, которая после обучения будет способна получать требуемые результаты с определенной степенью достоверности.

Параллельная структура работы нейронной сети позволяет работать своим элементарным функциональным элементам параллельно, что помогает сократить временные затраты в ходе обработки. Применения в качестве вычислительного

инструмента арифметики, которая будет также обладать свойством параллелизма, позволит в несколько раз увеличить скорость ее работы [68,69]. Мировой опыт показывает, что применение нейронный сетей вместе с системой остаточных классов является перспективным развитие [109].

1.5 Анализ моделей и методов фильтрации в цифровой обработке изображений

ЦОС в цифровых системах видеонаблюдения является неотъемлемой частью, которая необходима для ее функционирования. При этом ЦОС может включать себя ряд операций, для которых наилучшей является определенная математическая модель обработки. Так, операции, которые влияют на функционирование системы, а также обработку изображений выполняются за счет применения цифровых фильтров [11]. Частотно временные преобразования выборки данных осуществляются за счет применения быстрого преобразования Фурье (БПФ) или в некоторых случаях, вейвлет-преобразования.

Как было сказано в пункте 1.1, объекты в цифровых системах видеонаблюдения требуют получения результатов обработки информации при высокой скорости. В этом плане, хорошим математическим алгоритмом, который подходит моделям, описанным выше, является непозиционная система счисления [82,109]. В данном разделе покажем работу математических моделей обработки сигналов с применением системы остаточных классов.

1.5.1 Математические модели цифровых фильтров на основе системы остаточных классов

Выше было отмечено, что цифровая фильтрация выполняет ряд операций, оказывающих влияние на функционировании системы видеонаблюдения. Отсюда следует, что немаловажное значений на прием и обработку цифрового сигнала в

виде изображения в цифровых системах видеонаблюдения оказывают цифровые фильтры.

По существу фильтр преобразует входные сигналы в выходные таким образом, что определенные полезные особенности входного сигнала сохраняются в выходном сигнале, а нежелательные подавляются [100]. Другими словами, цифровой фильтр – это математический алгоритм, реализованный на аппаратном или программном уровне, который с заданной целью действует на входной и генерирует выходной цифровой сигнал [101].

Поскольку, цифровой фильтр обрабатывает цифровые сигналы [75], то соответственно для этого требуется предварительное преобразование сигнала из аналоговой (непрерывной) формы в цифровую (дискретную). По этой причине, различают цифровые и аналоговые фильтры, обрабатывающие соответственно цифровые и аналоговые сигналы [11]. Многое в теории цифровых фильтров, как для расчета, так и для применения, берет начало в области аналоговых фильтров [1]. Аналоговый фильтр обрабатывает аналоговый сигнал, свойства которого являются недискретными и соответственно передаточная функция зависит от внутренних свойств составляющих его элементов. Однако, на сегодняшний день, практическое применение цифровых фильтров, в отличии от аналоговых намного шире, что связано с рядом определенных качеств и свойств, влияющих на актуальность в применении.

Преимущества использования цифрового фильтра перед аналоговым фильтром представлены ниже [11,100]:

– высокая точность;

– линейная фаза;

– отсутствие дрейфа характеристик, вызванного погрешностью компонентов;

– легкость в моделировании и разработке;

– гибкость настройки, просты в изменении (возможна адаптивная фильтрация);

– компактность.

Недостаток аналоговых фильтров заключается в изменении параметров при изменении условий работы, т.е. при изменении температуры, давления влажности. Это приводит к возникновению погрешности выходного сигнала, или, иными словами, к низкой точности обработки сигналов. В цифровых фильтрах эти эффекты отсутствуют, что обуславливает их широкое применение [1].

Однако, применение цифровых фильтров, влечет за собой ряд недостатков. Основным, из которых, считается трудность работы с высокочастотными сигналами. Так как, полоса частот цифрового сигнала ограничена частотой Котельникова, равной половине частоты дискретизации сигнала, то по этой причине для высокочастотных сигналов применяют аналоговые фильтры [42]. Если же на высоких частотах нет полезного сигнала, тогда сначала с применением аналогового фильтра подавляют высокочастотные составляющие, после чего сигнал обрабатывают цифровым фильтром. Еще одним существенным недостатком цифрового фильтра является трудность его работы в реальном времени [47]. Цифровой фильтр должен проводить все вычисления в течение периода дискретизации, чтобы быть готовым к обработке следующего отсчета данных. Также следует отметить, что для большой точности и высокой скорости обработки сигналов, кроме мощного процессора, требуется и дополнительное, дорогостоящее, аппаратное обеспечение в виде высокоточных и быстрых цифро-аналоговых и аналого-цифровых преобразователей [6,12].

В [1] приводится методика расчета фильтра, которую можно использовать для реализации цифрового фильтра, удовлетворяющего заданным требованиям по обработке сигнала. Блок-схема такой методики приведена на рисунке 1.1 и включает себя 7 этапов, совокупность которых представляет собой алгоритм перехода от расчета аналоговых фильтров к расчету цифровых. Анализируя предложенную методику проектирования, можно сделать вывод, что основная задача проектирования цифровых фильтров складывается из задачи нахождения частотной характеристики или передаточной функции, параметры которых

удовлетворяют предъявленным к фильтру техническим требованиям [12]. Следовательно, в своей основе расчет фильтра представляет собой процесс нахождения математической аппроксимации.

Обобщенная функциональная схема на основе которой происходит процесс цифровой фильтрации сигнала представлена на рисунке 1.2. [1]. Работа такого фильтра состоит в выполнении ряда операций, в начале, которых непрерывный сигнал $x(t)$ поступает на вход аналого-цифрового преобразователя (АЦП). АЦП фиксирует значение $x(n)$ сигнала в дискретные моменты времени $t = nT$, $n = 0,1,...$ и преобразует их в цифровой код в виде двоичного числа. После чего последовательность $x(n)$ поступает в процессор, состоящий из арифметического устройства (АУ) и памяти (П). В процессоре происходит преобразование последовательности $x(n)$ в соответствии с определенным алгоритмом [1,12]. Результатом данного процесса является образованная последовательность $y(n)$. На следующем этапе обработки последовательность $y(n)$ поступает на цифро-аналоговый преобразователь (ЦАП), в котором текущее значение $y(n)$, представленное в цифровом виде, преобразуется в постоянное напряжение, удерживаемое в течении соответствующего интервала дискретности. На выходе ЦАП получает непрерывный сигнал $y(t)$ в виде ступенчатой функции. Фильтр нижних частот (ФНЧ) устраняет высокочастотные колебания, и выходной сигнал $\tilde{y}(t)$ цифрового фильтра приобретает сглаженный вид.

Цифровой фильтр полностью описывается его импульсной характеристикой. Импульсная характеристика – это реакция фильтра на единичный импульс, поданный на его вход [1,12,75]. Другая характеристика, используемая при изучении и проектировании цифрового фильтра, называется передаточной функцией. Она показывает, что делает фильтр с разными частотными составляющими входного сигнала, как он изменяет свой спектр сигнала.

Рисунок 1.1. – Методика проектирования фильтра

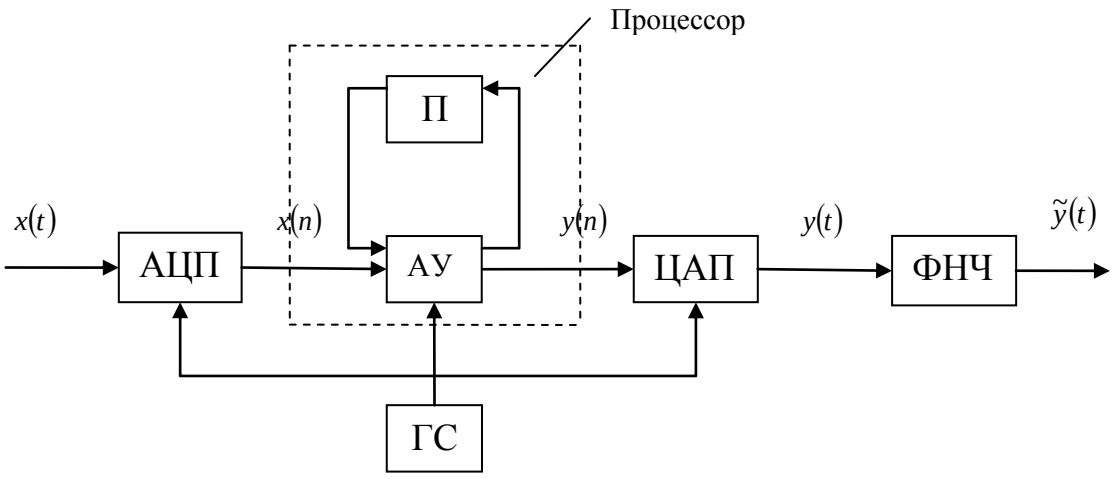


Рисунок 1.2. – Функциональная схема цифровой фильтрации сигнала

В [1] рассматриваются три основные характеристики фильтра: импульсная, переходная и частотная (рис. 1.3). Каждая из указанных характеристик в полной мере определяет свойства линейного фильтра, так как содержит исчерпывающую информацию о его фильтрующих свойствах. Все три характеристики играют важную роль при проектировании цифрового фильтра, так как они позволяют определить реакцию фильтра при различных исходных данных. При этом, зная хотя бы одну из них, можно рассчитать две другие. Переходная характеристика связана с импульсной интегральной зависимостью. Амплитудно-частотная характеристика рассчитывается (АЧХ) рассчитывается как быстрое преобразования Фурье (БПФ) импульсной характеристики с последующим вычислением модуля преобразования. Логарифмическая АЧХ (ЛАЧХ) получается при переходе к логарифмическому масштабу по оси ординат [12].

В зависимости от классификации импульсной характеристики существует два основных типа цифровых фильтров: фильтры с конечной импульсной характеристикой (КИХ-фильтры) и фильтры с бесконечной импульсной характеристикой (БИХ-фильтры). Фильтр каждого типа можно представить через коэффициенты его импульсной характеристики $h(k)$ $(k = 0,1,...)$, как представлено ниже на рисунке 1.4.

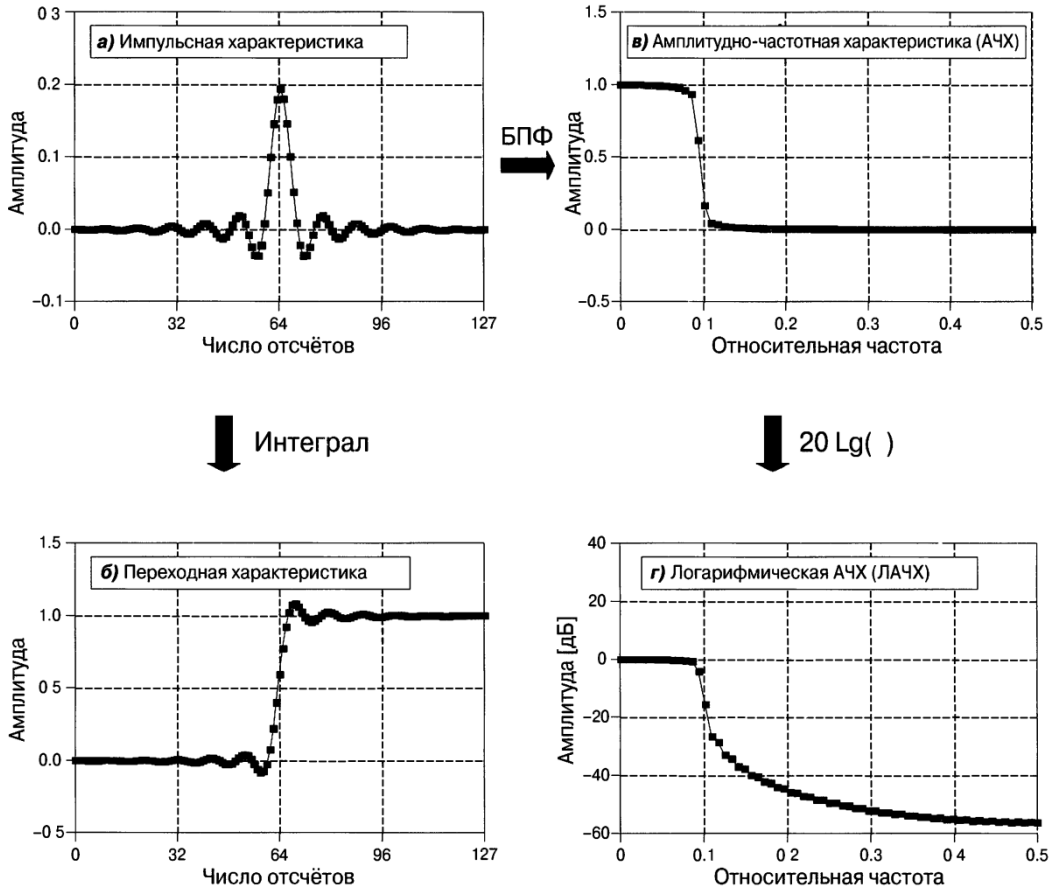


Рисунок 1.3. – Основные характеристики фильтров а) импульсная характеристика; б) переходная характеристика; в) амплитудно-частотная характеристика; г) логарифмическая АЧХ

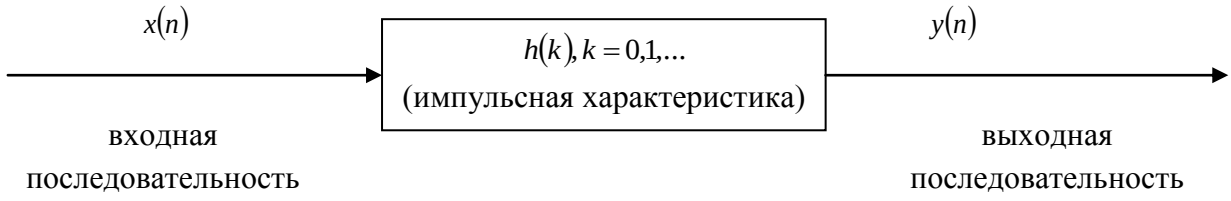


Рисунок 1.4. – Концептуальное представление цифрового фильтра

Как говорилось выше, любой из видов этих фильтров характеризуется своей передаточной функцией, уравнение которой представлено ниже:

$$H(z)=\frac{B(z)}{A(z)}=\frac{b_0+b_1z^{-1}+\ldots+b_Nz^{-N}}{1+a_1z^{-1}+\ldots+a_Mz^{-M}} \qquad (1.3)$$

где большее из N или M – это порядок фильтра. Главная задача передаточной функции состоит в описании того, как фильтр будет реагировать на входной сигнал.

Фильтром с бесконечной импульсной характеристикой (БИХ-фильтром) называют фильтр, у которого импульсная характеристика может принимать отличные от нуля значения на бесконечном множестве значений $n = 0,1,\ldots$ Формула (1.3) описывает работу такого фильтра [32]. Разностное уравнение, которое описывает дискретный БИХ-фильтр и устанавливает связь между входными и выходными сигналами во временной области, представлено формулой (1.4):

$$y(n) = \sum_{i=0}^{N} b_i x(n-i) - \sum_{k=1}^{Q} a_k y(n-k), \tag{1.4}$$

В этой формуле b_i – коэффициенты входного сигнала, N – порядок входного сигнала, Q – порядок обратной связи, a_k – коэффициенты обратной связи, $x(n)$ и $y(n)$ – входной и выходной сигнал соответственно. На рисунке 1.5 приведена каноническая форма реализации БИХ-фильтра.

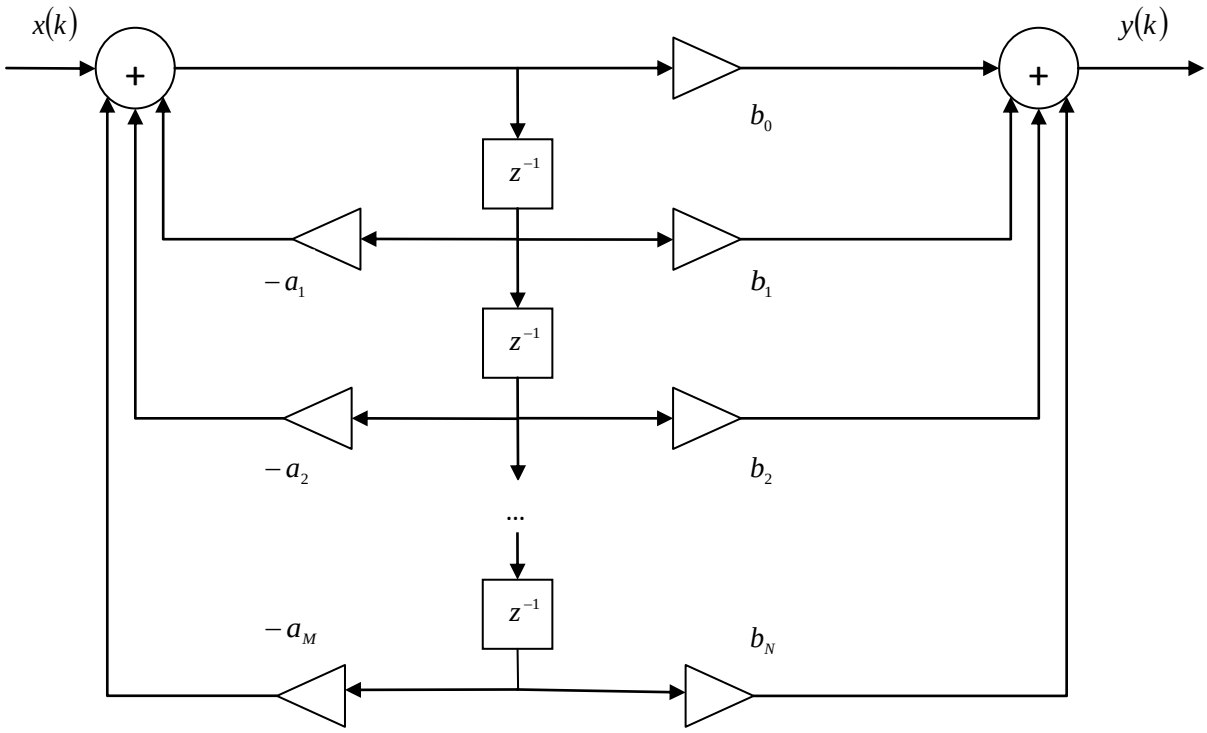


Рисунок 1.5. – Каноническая форма БИХ-фильтра

КИХ-фильтр в знаменателе своей передаточной функции имеет некую константу и работает в соответствии с уравнением, задающим свертку (1.5):

$$y_m = \sum_{k=0}^{N} b_k x_{m-k} \tag{1.5}$$

Обобщенная реализация прямой формы КИХ-фильтра с числом звеньев N представлена на рисунке 1.6.

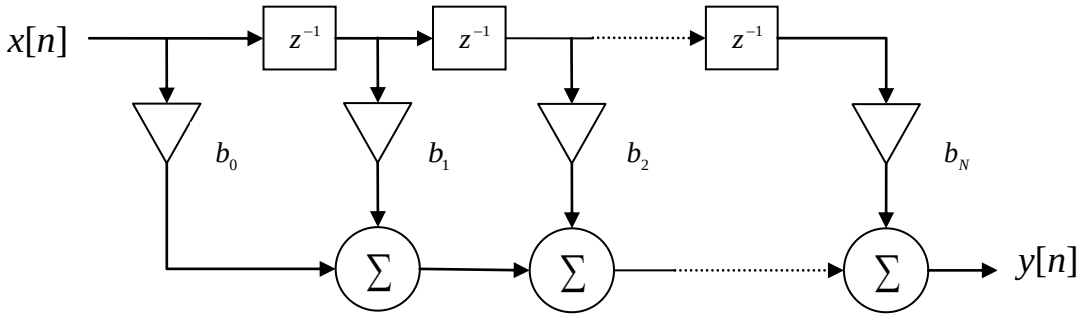


Рисунок 1.6. – Прямая форма КИХ-фильтра

Выбор между КИХ- и БИХ-фильтрами зависит от относительных преимуществ обоих типов [62]. Однако, также следует отметить, что каждый из рассмотренных видов цифровых фильтров имеет свои преимущества, но, тем не менее не лишен недостатков в реализации.

1. Ких-фильтры могут иметь строго линейную фазовую характеристику. Следовательно, фильтр не вводит фазового искажения в сигнал, что важно во многих сферах, например, передаче данных, цифровой обработке изображений. Фазовая характеристика БИХ-фильтров нелинейна, особенно на краях полос.

2. Ких-фильтры реализованы нерекурсивно, т.е. они всегда устойчивы. Гарантировать устойчивость БИХ-фильтров удается не всегда.

3. Для реализации фильтров используется ограниченное число битов. Практические последствия этого (например, шум округления и ошибки квантования) значительно менее существенны для КИХ-фильтров, чем для БИХ-фильтров.

4. Чтобы получить конечную импульсную характеристику с помощью фильтров с резкими срезами характеристики, потребуется больше коэффициентов, чем для получения бесконечной импульсной характеристики. Следовательно, для реализации предложенной спецификации амплитудной характеристики с КИХ необходимо больше вычислительной мощности и памяти, чем для реализации ее с БИХ. Впрочем, эффективность КИХ-реализаций можно значительно повысить, сыграв на вычислительной скорости БПФ и обработке при нескольких скоростях.

5. Аналоговые фильтры легко преобразовать в эквивалентные цифровые БИХ-фильтры, удовлетворяющие сходным спецификациям. Для получения КИХ-фильтров такое преобразование невозможно, поскольку для них не существует аналоговых прототипов. Однако, получать произвольные частотные характеристики на КИХ-фильтрах легче.

6. Синтез КИХ-фильтров алгебраически сложнее, если не использовать компьютерную поддержку разработки.

7. БИХ-фильтры рекуррентны. Это означает, что, пропустив через фильтр один и тот же сигнал, в итоге получаются разные результаты. Этот факт влечет за собой ряд ограничений по применению БИХ-фильтров.

Таким образом, перед проектированием цифрового фильтра необходимо сначала сформулировать требования к желаемым характеристикам фильтра, в зависимости от поставленной задачи, по которым затем будут рассчитываться параметры фильтра.

Реализовать цифровой фильтр можно используя аппаратный или программный способ проектирования [12]. Аппаратный способ состоит в реализации фильтра на элементах интегральных схем. Программная реализация осуществляется с помощью программируемых логических интегральных схем (ПЛИС) и с помощью цифровых процессоров обработки сигналов (ЦПОС). Высокая тактовая частота, большее число аппаратных интерфейсов по сравнению с ЦПОС, гибкость при разводке платы, позволяющая подавать данные произвольной разрядности, а также низкая стоимость – все эти преимущества

ПЛИС во многом обуславливают популярность в применении [74]. Однако, существуют сложные алгоритмы, выполнение которых с помощью ПЛИС невыгодно или невозможно. В таких случаях применяются сигнальные процессоры цифровой обработки сигналов (ЦОС).

1.5.2 Математическая модель КИХ-фильтра с применением системы остаточных классов

Как было отмечено выше, решение определенного класса задач накладывает желаемые характеристики, которыми должна обладать модель цифрового фильтра. В цифровых системах видеонаблюдения должны быть использованы фильтры, которые менее чувствительны к ошибкам квантования. В этом смысле хорошо подходит математическая модель КИХ-фильтра. Линейная структура КИХ-фильтра с применением системы остаточных классов дает возможность ускорить и упростить ряд вычислений [62].

Пусть система остаточных классов задана набором модулей $p_1, p_2, ..., p_n$. Тогда формула (1.6) для расчета КИХ-фильтра с применением СОК примет следующий вид:

$$\left|y_m\right|_{p_i} = \left|\sum_{k=0}^{N} \left|b_k\right| \cdot \left|x_{m-k}\right|_{p_i}\right|_{p_i} \quad (1.6)$$

Полученная система будет являться совокупностью КИХ-фильтров на основе СОК. Блок схема ее интерпретирующая представлена на рисунке 1.7.

Достоинством фильтра является выполнения алгебраических операций параллельно, достигаемое за счет применения СОК. При этом КИХ-фильтр обеспечивает снижение энергопотребление в системе в целом. Более подробно о данном преимуществе говорится в работах [64, 66,78].

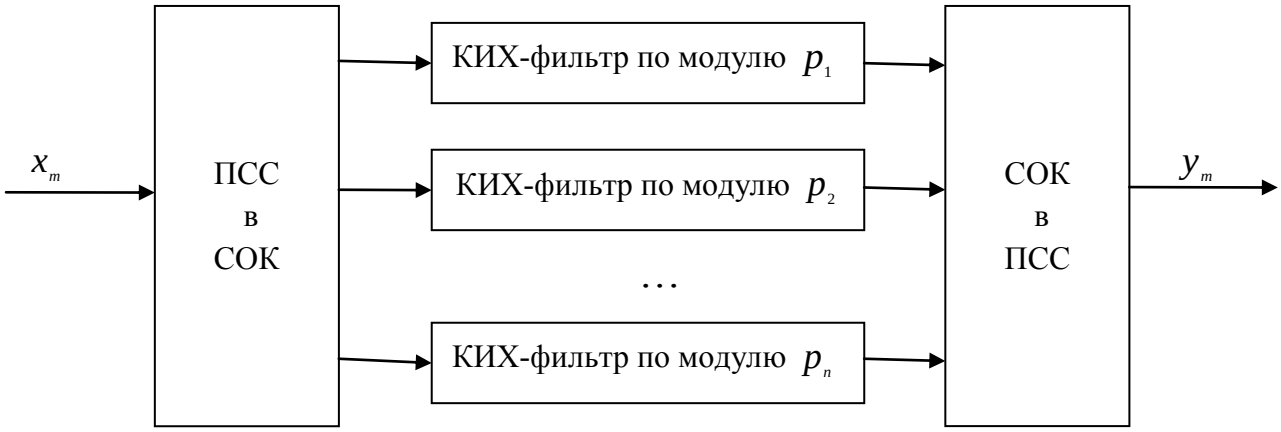


Рисунок 1.7 – Общая структура КИХ-фильтра в СОК

1.5.3 Математическая модель алгоритма быстрого преобразования Фурье

Одним из наиболее распространенных методов в цифровой обработке сигналов являются методы на основе дискретного преобразования Фурье (ДПФ). ДПФ представляет собой процесс разложения дискретного сигнала x_m с периодом N в ряд Фурье. Дальнейшая обработка данных ДПФ происходит путем замены значений сигнала x_m его коэффициентами разложения или другими словами, спектром сигнала [1]. Однако, ДПФ имеет достаточно высокую вычислительную сложность, по этой причине существует ряд алгоритмов способных сократить вычислительные затраты. Одним из таких алгоритмов является быстрое преобразование Фурье (БПФ). БПФ сокращает количество умножений в вычислительной структуре за счет возможности группировать повторяющиеся слагаемые с одинаковыми множителями. Особенностью БПФ является возможность работать с сигналом лишь тогда, когда длина анализируемого сигнала является степенью двойки [1,12]. Помимо высокой вычислительной скорости БПФ также является достаточно точным и алгоритмом с малым количеством ошибок округления.

Рассмотрим преобразования Фурье длины N . В этом случае ДПФ записывается в виде:

$$x_m = \sum_{k=0}^{N} C_k W_N^{km} \tag{1.7}$$

где C_k – обобщенный спектр сигнала, $W_N^{km} = \cos\dfrac{2\pi k(m+\varphi_k)}{N}$ – обобщенная синусоида. Для ДПФ с основанием 2 с применением соответствующих преобразований для четных и нечетных индексов получим:

$$x_m = \sum_{r=0}^{N/2} C_r^1 W_{N/2}^{rm} + W_N^m \sum_{r=0}^{N/2} C_r^2 W_{N/2}^{rm} \tag{1.8}$$

где $W_N^m = \cos\dfrac{2\pi(m+\varphi_k)}{N}$, C_r^1 и C_r^2 представляют собой искомый спектр C_k.

БПФ обладает свойством линейности, поэтому оно хорошо подходит для реализации с использованием системы остаточных классов. В этом случае, формула (1.8) для СОК с основаниями $p_1, p_2, ..., p_n$ примет вид [56]:

$$\left|x_m\right|_{p_i} = \left|\sum_{r=0}^{N/2}\left|C_r^2\right|_{p_i} \cdot \left|W_{N/2}^{rm}\right|_{p_i}\right|_{p_i} + \left|W_N^m\right|_{p_i} \cdot \left|\sum_{r=0}^{N/2}\left|C_r^2\right|_{p_i} \cdot \left|W_{N/2}^{rm}\right|_{p_i}\right|_{p_i}\Bigg|_{p_i} \tag{1.9}$$

где $i = 1,2,...,n$.

Таким образом, применение СОК позволяет свести вычисления к n параллельным вычислительным каналам для соответствующего модуля p_i.

1.5.4 Математическая модель алгоритма вейвлет-преобразования

Для проведения Фурье-преобразования используются гармонические функции, которые хорошо локализованы в частотной области, но не локализованы во временной [41,42]. Это является существенным недостатком преобразования Фурье, которое можно преодолеть, использовав вейвлет-преобразование, предоставляющее как частотную так и временную информацию об обрабатываемом сигнале [55]. Исследование по проблемам применения вейвлет-преобразования в ЦОС отражены в работах [61,82,96]. Наиболее актуальными в применении вейвлетами являются вейвлеты Хаара [16,17] и вейвлеты Добеши [15,60].

В основе вейвлет-преобразования лежит использование двух непрерывных и интегрируемых по всей оси функций вейвлет-функции $\psi(t)$ и масштабирующей или скейлинг-функции $\varphi(t)$, таких что [16,48]:

$$\int_{-\infty}^{\infty}\psi(t)dt=0,\quad \int_{-\infty}^{\infty}\varphi(t)dt=1. \tag{1.10}$$

Вейвлет-функция также должна обладать свойством смещения во времени и масштабируемости:

$$\psi(t,a,b)=\psi(a,b,t)=a^{-\frac{1}{2}}\psi_0\left(\frac{t-b}{a}\right). \tag{1.11}$$

Прямое непрерывное вейвлет-преобразование сигнала $s(t)$ задается вычислением вейвлет-коэффициентов по формуле:

$$C(a,b)=\langle s(t),\psi(a,b,t)\rangle=\int_{-\infty}^{\infty}s(t)a^{-\frac{1}{2}}\psi\left(\frac{t-b}{a}\right)dt\,. \tag{1.12}$$

Обратное непрерывное вейвлет-преобразование осуществляется по формуле восстановления во временной области [17]:

$$f=C_{\psi}^{-1}\int_{-\infty-\infty}^{\infty}\int^{\infty}\frac{dadb}{a^2}\langle f,\psi^{a,b}\rangle\psi^{a,b}\,. \tag{1.13}$$

В связи со значительным неудобством вычисления интегралов, на практике величины a и b обычно задаются дискретно [41]. В этом случае реализуется дискретное вейвлет-преобразование. В численном и функциональном анализе дискретные вейвлет-преобразования (ДВП) относятся к вейвлет-преобразованиям, в которых вейвлеты представлены дискретными сигналами (выборками) [17] в следующем виде:

$$S(t)=\sum_{i=1}^{N-1}S(i\Delta t)\delta(t-i\Delta t), \tag{1.14}$$

Следует отметить, что математические алгоритмы вейвлет-преобразований для вычисления большого количества интегралов используют более быстрые математические процедуры. Вычисления коэффициентов для преобразований такого вида требует выполнения двух видов алгебраических операций: сложения

и умножения. Указанные операции могут быть эффективно реализованы с применением системы остаточных классов.

1.6 Применение нейронных сетей в обработке изображений

Как говорилось в пункте 1.2, одним из методов распознавания образов является метод, основанный на использовании архитектуры нейронных сетей. Этот метод несет больший интерес в решении задач цифровой обработки изображений, и также является перспективной технологией в интернет и IT-индустрии. Столь повышенный интерес обусловлен значительным развитием проектов и приложений, использующих технологию нейросетей. Среди основных областей применения нейронных сетей – прогнозирование, принятие решений, распознавание образов, оптимизация, анализ данных и др [28,29,50].

Главной особенностью нейронных сетей (или другими словами нейрокомпьютеры, нейроморфные системы, многослойные самонастраивающиеся сети, модели распределенной параллельной обработки) является использование большого числа простейших нелинейных вычислительных элементов (называемых нейронами), которые организованы в виде сетей, напоминающих предположительный способ соединения нейронов в человеческом мозге [40]. История ведет их развитие с начала 1943 года, с работы Мак-Каллака и Питтса, которые впервые предложили модель нейрона в виде двоичного порогового устройства. На данный момент, искусственные нейронные сети (ИНС) являются не только инструментом решения задач распознавания образов, но получили применение в исследованиях по ассоциативной памяти, сжатию изображений и т.д [46,86,87].

Как было сказано выше, нейронные сети (НС) состоят из элементов, называемых формальными нейронами, которые сами по себе очень просты и связаны с другими нейронами. Простейшая сеть состоит из группы нейронов, образующих слой, как показано в правой части рисунка 1.8 [51,70].

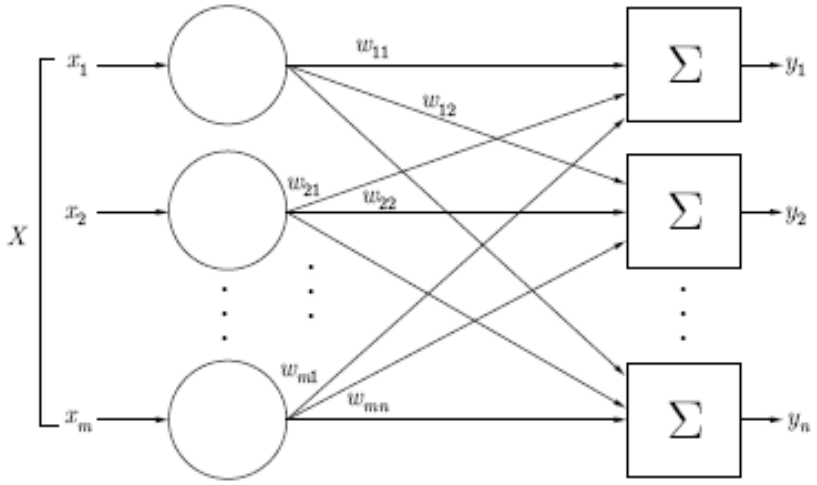


Рисунок 1.8 – Пример однослойной искусственной нейронной сети

Каждый нейрон преобразует набор сигналов, поступающих к нему на вход X в выходной сигнал Y. При этом, именно связи между нейронами, кодируемые весами, играют ключевую роль. Удобно считать веса элементами матрицы W. Матрица имеет m строк и n столбцов, где m – число входов, а n – число нейронов. Например $\omega_{2,3}$ – это вес, связывающий второй вход с третьим нейроном. Таким образом, вычисление выходного вектора N, компонентами которого являются выходы нейронов, сводится к матричному умножению вида [40]:

$$N = XW \tag{1.15}$$

где N и X – векторы-строки.

Однако, более крупные и сложные нейронные сети обладают, как правило, большими вычислительными возможностями. Примером такой нейронной сети являются многослойные нейронные сети.

Следует отметить, что одним из преимуществ НС (а также недостатком при реализации их на последовательной архитектуре) является то, что все элементы могут функционировать параллельно, тем самым существенно повышая эффективность решения задач, особенно в обработке изображений. Кроме того, что НС позволяют эффективно решать многие задачи, они предоставляют мощные гибкие и универсальные механизмы обучения, что является их главным

преимуществом перед другими методами [46,86,87] (вероятностные методы, линейные разделители, решающие деревья и т.п.). Обучение избавляет от необходимости выбирать ключевые признаки, их значимость и отношения между признаками. Но, тем не менее, выбор исходного представления входных данных (вектор в n-мерном пространстве, частотные характеристики, вейвлеты и т.п.), существенно влияет на качество решения исходной задачи [13]. Кроме того, еще одной важной особенностью НС является то, что они обладают хорошей обобщающей способностью (лучше, чем у решающих деревьев [50]), т.е. могут успешно распространять опыт, полученный на конечном обучающем наборе, на все множество образов.

1.6.1 Архитектура многослойных нейронных сетей

Как было сказано выше, одной из разновидностей НС являются многослойные нейронные сети (МНС). Эта сеть была предложена в работах Розенблатта и является наиболее часто используемой. Архитектура такой сети состоит из последовательно соединенных слоев, где нейрон каждого слоя своими входами связан со всеми нейронами предыдущего слоя, а выходами следующего [70]. НС с двумя решающими слоями может с любой точностью аппроксимировать любую многомерную функцию. НС с одним решающим слоем способна формировать линейные разделяющие поверхности, что сильно сужает круг задач ими решаемых. НС с нелинейной функцией активации и двумя решающими слоями позволяет формировать любые выпуклые области в пространстве решений, а с тремя решающими слоями – области любой сложности, в том числе невыпуклой. При этом МНС не теряет своей обобщающей способности. Обучаются МНС при помощи алгоритма обратного распространения ошибки, являющегося методом градиентного спуска в пространстве весов с целью минимизации суммарной ошибки сети. Ошибки, точнее величины коррекции весов распространяются в обратном направлении от входов к выходам, сквозь веса, соединяющие нейроны. Примером применения

простейшей однослойной НС [40] (называемой автоассоциативной памятью) служит обучение такой сети для решения задач восстановления подаваемого изображения. Подавая на вход тестовое изображение, и вычисляя качество реконструированного изображения, можно оценить насколько сеть распознала входное изображение. Достоинство этого метода заключается в возможности сети восстанавливать искаженные и зашумленные изображения, однако для решения более сложных задач применение такой сети не подходит [38].

Таким образом, многослойная нейронная сеть может моделировать функцию практически любой степени сложности, причем число слоев и число элементов в каждом слое определяют сложность функции. Единообразия в решения вопроса относительно того, как считать число слоев в такой сети нет, поэтому выделяют два подхода: считать число слоев, включая несуммирующий входной слой; считать только слои, выполняющие суммирование. Следует отметить, что определение числа промежуточных слоев и числа элементов в них является важным вопросом при конструировании сети [31]. На рисунке 1.9 приведена двухслойная нейронная сеть, при этом согласно второму подходу, вход распределительного слоя считается нулевым слоем.

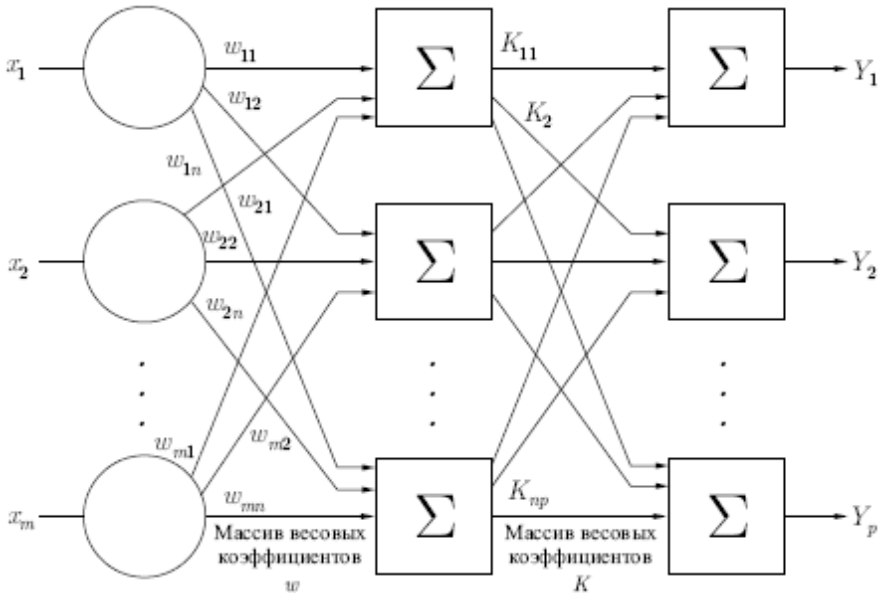


Рисунок 1.9 – Пример двухслойного персептрона

Многослойные сети не могут привести к увеличению вычислительной мощности по сравнению с однослойной сетью, если активационная функция между слоями линейна. Вычисления выхода (OUT) слоя для многослойной сети заключается в умножении входного вектора на первую весовую матрицу с последующим умножением (если отсутствует нелинейная активационная функция) результирующего вектора на вторую весовую матрицу [30,31]:

$$OUT = (XW_1)W_2 \tag{1.16}$$

По свойству ассоциативности умножения матриц, получаем :

$$(XW_1)W_2 = X(W_1W_2) \tag{1.17}$$

что показывает, что двухслойная линейная сеть эквивалентна одному слою с весовой матрицей, равной произведению двух весовых матриц. Это говорит о том, что любая многослойная линейная сеть может быть заменена эквивалентной однослойной сетью [70]. Однако, как было сказано, однослойные сети весьма ограничены по своим вычислительным возможностям, поэтому для решения возможностей сетей по сравнению с однослойной сетью необходима нелинейная активационная функция.

Среди многослойных нейронных сетей можно выделить четыре наиболее значимых и важных класса нейронных сетей [9,70]:

1. Сети прямого распространения – все связи направлены строго от входных нейронов к выходным. Такие сети еще называют многослойным персептроном, по аналогии с обычным персептроном Розенблатта, в которой только один скрытый слой.

2. Реккурентные нейронные сети или сети обратного распространения – сигнал в таких сетях с выходных нейронов или нейронов скрытого слоя частично передается обратно на входы нейронов входного слоя.

3. Радиально базисные функции – вид многослойной нейронной сети, имеющий скрытый слой из радиальных элементов и выходной слой из линейных элементов. Сети этого типа довольно компактны и быстро обучаются. Радиально

базисная сеть обладает следующими особенностями: один скрытый слой, только нейроны скрытого слоя имеют нелинейную активационную функцию и синаптические веса входного и скрытого слоев равны единицы.

4. Самоорганизующие карты или сеть Кохонена. Такой класс многослойных нейронных сетей, как правило, обучается без учителя и успешно применяется в задачах распознавания. Сети такого класса способны выявлять новизну во входных данных: если после обучения сеть встретится с набором данных, непохожим ни на один из известных образцов, то она не сможет классифицировать такой набор и тем самым выявит его новизну. Сеть Кохонена имеет всего два слоя: входной и выходной, составленный из радиальных элементов.

Однако, не смотря на свои преимущества и вычислительные возможности, при обработке изображения многослойная нейронная сеть имеет ряд недостатков. В классической МНС межслойные нейронные соединения полносвязны, и изображение представлено в виде одномерного вектора, хотя оно двумерно. Для устранения этих недостатков используется архитектура сверточной нейронной сети (СНС) [40,70].

1.6.2 Анализ архитектур сверточных нейронных сетей

Как говорилось выше, определенная специфика решаемой задачи требует наличия специальной структуры нейронной сети. Одной из таких задач является рассматриваемая задача распознавания образов, в частности распознавание изображений, которая использует специальную архитектуру нейронной сети – сверточную нейронную сеть. Данная архитектура была предложена Яном Лекуном и нацелена на эффективное распознавание изображений, которые являются входными данными для этих сетей. Свое название сверточная нейронная сеть (СНС) получила согласно названию операции свертки, на которой эта сеть базируется. Суть свертки состоит в том, что каждый фрагмент (значение пикселя) изображения умножается на матрицу (ядро) поэлементно, после чего

результат суммируется и записывается в аналогичную позицию выходного изображения.

Чаще всего СНС представляет собой чередование сверточных слоев (convolution layers), субдискретизирующих слоев (sumsampling layers) и при наличии полносвязных слоев (fully-connected layer) на выходе. Все три вида слоев выполняют разные функции и могут чередоваться в произвольном порядке [70].

Каждый слой состоит из пластин сгруппированных нейронов. Группировка в виде пластин позволяет обрабатывать входные сигналы с сохранением топологических отношений. В противном случае, топологические отношения могут потеряться если подать вход в виде вектора, а не матрицы.

Основной особенностью СНС является понятие разделяемых весов, которое означает, что часть нейронов некоторого слоя может использовать одни и те же синаптические веса. Такие нейроны объединяются в карты признаков (feature maps). При этом, все нейроны в рамках одной карты признаков связаны с одной и той же частью нейронов предыдущего слоя (либо с одной и той же частью входов сети в случае, если данная карта признаков находится на первом слое), причем количество нейронов (соответственно входов сети) в этой части совпадает с количеством нейронов в самой карте признаков. В процессе работы сети, каждый из нейронов этой карты признаков выполняет операцию свертки (convolution) определенной области нейронов предыдущего слоя (либо входов сети), общей для всех нейронов в рамках этой карты признаков.

Полносвязный слой СНС представляет собой слой, в котором каждый нейрон соединен со всеми нейронами на предыдущем уровне так, что каждая связь имеет свой весовой коэффициент. Пример полносязного слоя, где w_{ij} – весовые коэффициенты, $f(\cdot)$ – функция активации, представлен на рисунке 1.10.

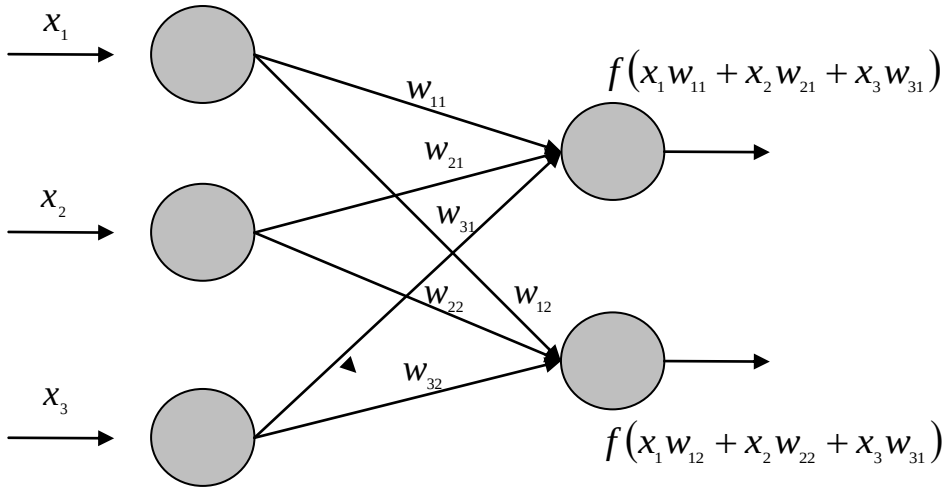


Рисунок 1.10. Полносвязный слой

Сверточный слой представляет собой слой, в котором нейрон соединен с ограниченным количеством нейронов предыдущего уровня, что отличает его от полносвязного слоя [40,70]. Сверточный слой аналогичен применению операции свертки, где используется лишь матрица весов небольшого размера (ядро свертки), которую двигают по всему обрабатываемому слою. Еще одной особенностью сверточного слоя является то, что он немного уменьшает размер изображения за счет краевых эффектов. Рисунок 1.11 иллюстрирует пример работы сверточного слоя с ядром свертки размера 3×3.

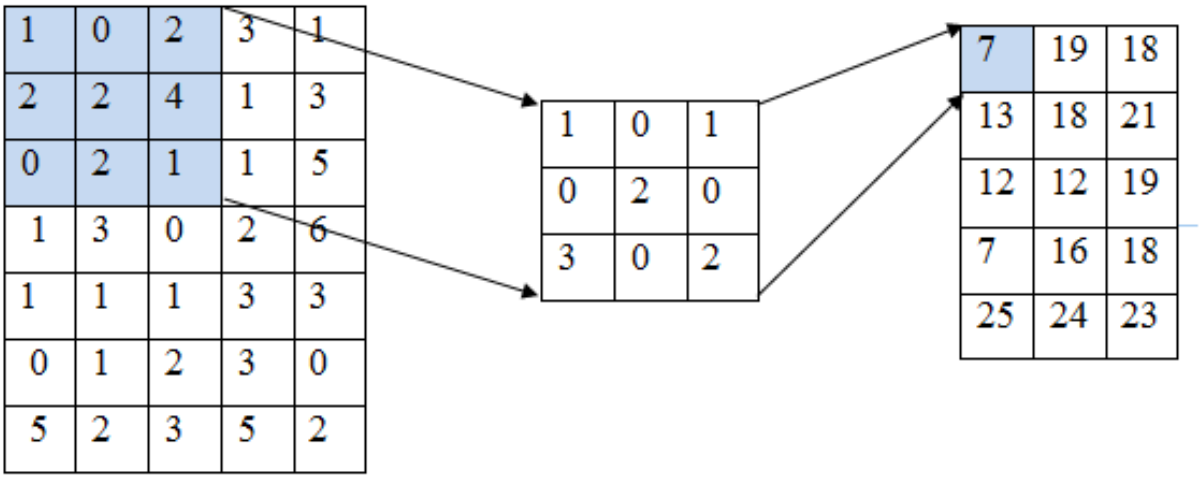


Рисунок 1.11. Сверточный слой

Еще одним видом слоя, который встречается в СНС является субдискретизирующий слой. Такой вид слоя выполняет уменьшение размерности (обычно в несколько раз). Уменьшение может происходить разными методами, но чаще всего используют метод, основанный на выборе максимального элемента (max-pooling). Такой подход разделяет карту признаков на ячейки, на которых выбираются максимальные значения. Рисунок 1.12 иллюстрирует пример работы субдискретизирующего слоя на основе метода выбора максимального элемента [70].

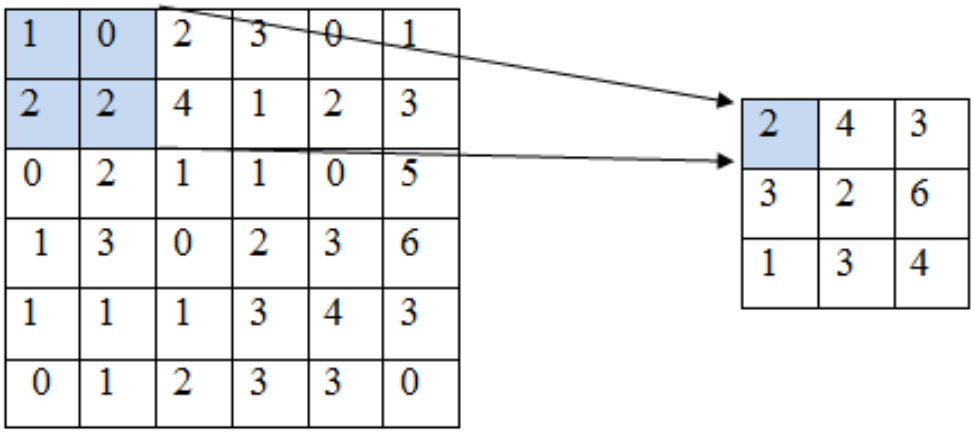


Рисунок 1.12. Субдискретизирующий слой

Все три вида слоев могут чередоваться в произвольном порядке, что позволяет составлять карты признаков из карт признаков, а это на практике означает способность распознавания сложных иерархий признаков. Более поздние слои извлекают более общие характеристики, меньше зависящие от искажений изображения. В классической архитектуре сверточная нейронная сеть состоит из двух слоев: слоя свертки и слоя усреднения.

Таким образом, идея сверточных нейронных сетей заключается в чередовании сверточных и субдискретизирующих слоев, обеспечивающих частичную устойчивость к изменениям масштаба, смещениям, поворотам,

искажениям. Для обучения такая сеть использует стандартные методы, чаще всего метод обратного распространения ошибки.

Преимущества использования сверточной нейронной сети перед другими методами решения задачи распознавания образов представлены ниже [70,113]:

– Один из лучших алгоритмов по распознаванию и классификации изображений.

– По сравнению с полносвязной нейронной сетью имеет гораздо меньшее количество настраиваемых весов. Это связано с тем, что для всего изображения используется только одно ядро весов, вместо того, чтобы использовать для каждого пикселя входного изображения свои персональные коэффициенты. Данный подход подталкивает нейросеть при обучении к обобщению демонстрируемой информации, а не попиксельному запоминанию каждой показанной картинки в весовых коэффициентов, как это делает перцептрон.

– Удобное распараллеливание вычислений, а следовательно, возможность реализации алгоритмов работы и обучения сети на графических процессорах.

– Относительная устойчивость к повороту и сдвигу распознаваемого изображения.

– Обучение при помощи классического метода обратного распространения ошибок.

Недостаток рассматриваемого метода заключается в том, что он имеет много варьируемых параметров сети, и становится затруднительно решить, для какой задачи и вычислительной мощности какие настройки нужны. К варьируемым параметрам можно отнести: количество слоев, размерность ядра свертки для каждого из слоев, количество ядер для каждого из слоев, шаг сдвига ядра при обработке слоя, необходимость слоев субдискретизации, степень уменьшения ими размерности, функция по уменьшению размерности (выбор максимума, среднего и т.д.), передаточная функция нейронов. Все эти параметры

существенно влияют на результат, но выбираются исследователями эмпирически. Существуют несколько прекрасно работающих конфигураций сетей, но нет правил, по которым нужно делать выбор для новой задачи.

Сравнение МНС и СНС [116] показало существенные преимущества последней как по скорости, так и по надежности классификации. Полезным свойством СНС является и то, что характеристики, формируемые на выходах верхних слоев иерархии, могут быть применимы для классификации по методу ближайшего соседа, причем СНС может успешно извлекать такие характеристики и для образов, отсутствующих в обучающем наборе. Также, в отличии от МНС для СНС характерны более быстрая скорость обучения и работы. В [116] показаны результаты тестирования СНС на базе данных ORL, содержащие изображения лиц с небольшими изменениями освещения, масштаба, пространственных поворотов, положения и различными эмоциями. Результаты показывают приблизительно 98% точности распознавания, причем для известных лиц, предъявлялись варианты их изображений, отсутствующие в обучающем наборе. Такой результат позволяет сделать вывод о том, что архитектура СНС является перспективой для дальнейших разработок в области распознавания изображений.

1.7 Обоснование целесообразности применения СОК для решения задач цифровой обработки изображений

Основными проблемами цифровой обработки изображений, возникающими в цифровых системах видеонаблюдения, являются качество обработки подаваемых на вход изображений, скорость передачи данных и в свою очередь, энергетическая эффективность таких систем. Точность и быстродействие, определяемые необходимостью проведения большого количества вычислений в процессе функционирования цифровых приложений, оказывают существенное влияние на качество принимаемой информации. Основными путями решения

указанных проблем является модернизация цифровых систем видеонаблюдения за счет использования более эффективных способов вычислений [43,44,121].

С учетом основополагающих требований предъявляемых к построению высокопроизводительных вычислительных средств, в том числе применимых в цифровых системах видеонаблюдения, подтверждается основной метод решения задачи повышения скорости обрабатываемых цифровых данных, а именно метод, позволяющий строить структуру вычислительного устройства такой системы с максимальным распараллеливанием выполнения арифметических операций [67,78]. Этот метод в свою очередь решает ряд задач, которые ставятся перед вычислительным устройством:

– внедрение эффективных алгоритмических и аппаратных структур параллельного типа;

– обеспечение высокой степени интеграции и унификации арифметического устройства;

– применение перспективных средств контроля ошибок;

– использование вариантов машинной арифметики, наилучшим образом приспособленных для высокоскоростных реализаций вычислительных процессов, требующих больших объемов вычислений.

Применение привычной двоичной системы счисления в ходе выполнения арифметических операций над большим объемом данных влечет за собой ряд сложностей, вызванных наличием межразрядных связей. Указанный недостаток, накладывает ограничения на способы реализации арифметических операций, тем самым усложняет аппаратуру и ограничивает быстродействие системы [23,79,80]. Поэтому целесообразно применение такой арифметики, в которой бы поразрядные связи при вычислениях отсутствовали либо были сведены к минимуму. Арифметикой, обладающей указанными свойствами является непозиционная система счисления – система остаточных классов (СОК). Таким образом, поиск путей решения проблемы повышения производительности привели к мысли о независимой параллельной обработки данных и

следовательно, о замене привычной двоичной системы счислений системой остаточных классов.

В указанной системе числа представляются своими остатками от деления на выбранную систему оснований, и все рациональные операции могут выполняться параллельно над цифрами каждого разряда в отдельности [2]. Однако, столь удобной в одном отношении системе остаточных классов присущ ряд недостатков в других отношениях: ограниченность действия этой системы полем целых положительных чисел, трудность определения соотношений чисел по величине, определения выхода результата операции из диапазона и т.д. В свою очередь, перечисленные недостатки требуют эффективных путей их преодоления [20,52,58,107,108].

В СОК числа представляются в базисе взаимно-простых чисел, называемых модулями $\beta = \{p_1,...,p_k\}$, $НОД(p_i, p_j) = 1$, для $i \neq j$. Произведение всех модулей СОК $P = \sum_{i=1}^{k} p_i$ называется динамическим диапазоном системы. Любое целое число $0 \le X < P$ может быть единственным образом представлено в СОК в виде вектора $\{x_1, x_2,..., x_k\}$, где $x_i = |X|_{p_i} = X \bmod p_i$ [110].

Динамический диапазон СОК обычно делится на две примерно равные части, таким образом, чтобы примерно половина диапазона представляла положительные числа, а остальная часть диапазона – отрицательные. Таким образом, любое целое число, удовлетворяющее одному из двух соотношений:

$$-\frac{P-1}{2} \le X \le \frac{P-1}{2}, \text{ для нечетных } P, \qquad (1.18)$$

$$-\frac{P}{2} \le X \le \frac{P}{2}, \text{ для четных } P,$$

может быть представлено в СОК.

Операции сложения, вычитания и умножения в СОК определяются формулами

$$A \pm B = (\alpha_1, \alpha_2,..., \alpha_n) \pm (\beta_1, \beta_2,..., \beta_n) = ((\alpha_1 \pm \beta_1) \bmod p_1, (\alpha_2 \pm \beta_2) \bmod p_2,..., (\alpha_n \pm \beta_n) \bmod p_n); \qquad (1.19)$$

$$A \times B = (\alpha_1, \alpha_2, ..., \alpha_n) \times (\beta_1, \beta_2, ..., \beta_n) = ((\alpha_1 \times \beta_1) \bmod p_1, (\alpha_2 \times \beta_2) \bmod p_2, ..., (\alpha_n \times \beta_n) \bmod p_n). \quad (1.20)$$

Равенства (1.19) – (1.20) показывают параллельную природу СОК, свободную от поразрядных переносов. Эти операции носят название модульных, так как для их выполнения в СОК достаточно одного такта обработки численных значений.

Для преобразования чисел из двоичной позиционной системы счисления в СОК используем алгоритм, основанный на применении распределенной арифметики [59]. K -битное число X разобьем на отдельные форматы, для каждого из которых отводится заранее известное количество B -двоичных разрядов. Тогда n -битное двоичное число может быть выражено как комбинация $\frac{n}{B}$ – взвешенных (позиционных) форматов размерностью B бит (разрядов). При этом позициям каждого формата присваивается определенный вес 2^j , где $j = 0, B, 2B, ..., MB$.

$$X = \sum_{j=0}^{M} \left(\sum_{i=0}^{B-1} x_i 2^i \right) 2^j , \quad (1.21)$$

где B – количество разрядов выбранного формата; M – степень формата; x_i – коэффициент 0 или 1; $j = 0, B, 2B, ..., MB$ – позиция формата; i – позиция разряда в формате. Если развернуть выражение (1.21), то

$$X = \left(x_0 2^0 + x_1 2^1 + ... + x_{B-1} 2^{B-1} \right) 2^0 + \left(x_0 2^0 + x_1 2^1 + ... + x_{B-1} 2^{B-1} \right) 2^B + ...$$
$$... + \left(x_0 2^0 + x_1 2^1 + ... + x_{B-1} 2^{B-1} \right) 2^{M \times B} . \quad (1.22)$$

Из выражения (1.22) видно, что число X может быть представлено двоичным числом ширины $M \times B$ разрядов. Преобразование числа из двоичного позиционного кода в модулярный код осуществляется с помощью модульного суммирования остатков по модулю m_i [2]:

$$X = \left| \sum_{j=0}^{M} \left(\left| \sum_{i=0}^{B-1} x_i 2^i \right|_{m_i} \right) 2^j \right|_{m_i} . \quad (1.23)$$

Если в качестве примера входного числа взять 32-разрядное число и число разрядов в формате взять равным 8 бит, тогда входное число может быть выражено как комбинация четырех байтов:

$$X[n] = x[n;31...24]2^{24} + x[n;23...16]2^{16} + x[n;15...8]2^{8} + x[n;7...0]2^{0}. \quad (1.24)$$

Остаток числа $X[n]$ по модулю p_i может быть представлен в виде

$$X_i[n] = \left|x[n;31...24]2^{24}\right|_{p_i} + \left|x[n;23...16]2^{16}\right|_{p_i} + \left|x[n;15...8]2^{8}\right|_{p_i} + \left|x[n;7...0]2^{0}\right|_{p_i}. \quad (1.25)$$

Структурная схема преобразователя числа из двоичной формы в СОК, соответствующая выражению (1.25), приведена на рисунке 1.13 [59]. Она состоит из просмотровых таблиц LUT1-LUT4 и трех нейронных сетей конечного кольца по модулю m_i.

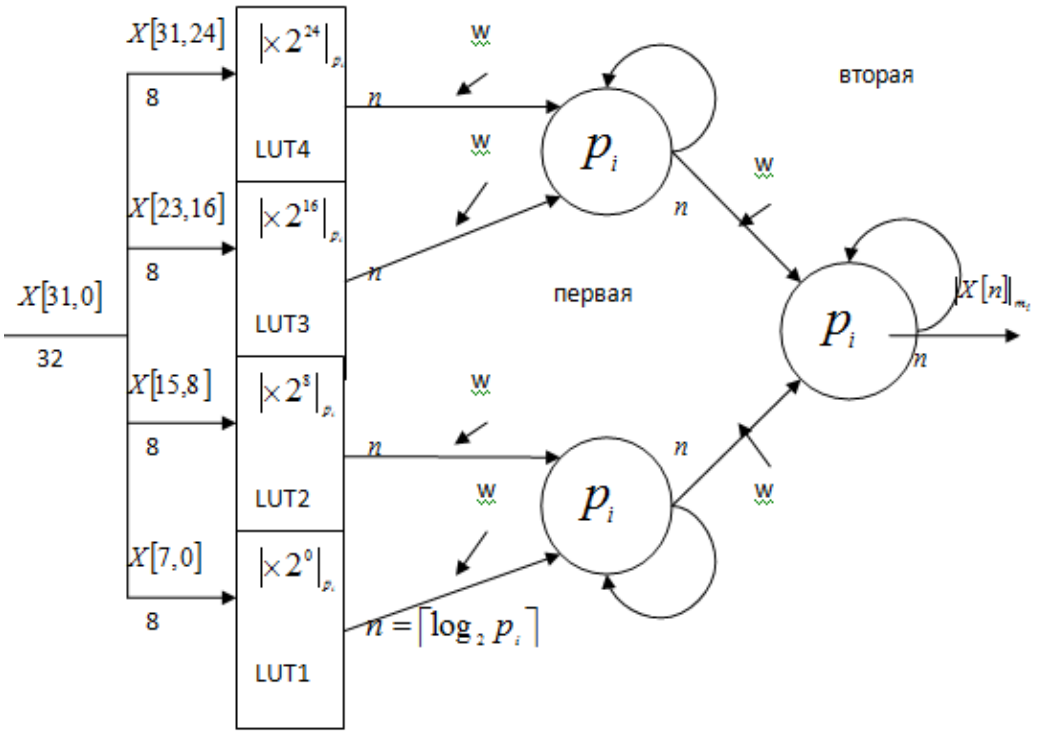


Рисунок 1.13. Схема преобразователя числа из двоичной формы в СОК.

Каждый член выражения (1.25), с учетом веса 2^j, может быть предварительно вычислен и сохранен в просмотровых таблицах. Организация просмотровых таблиц определяется двоичным представлением модуля p_i и выражается формулой $n = \lceil \log_2 p_i \rceil$. Для реализации каждого байта используются LUT с организацией $n \times 2^4$. На выходе каждой просмотровой таблицы формируются остатки по модулю p_i, которые поступают на вход нейронных

сетей конечного кольца, состоящих из двух ступеней. В первой ступени используются две сети, а во второй одна нейронная сеть. Первая ступень нейронных сетей конечного кольца обеспечивает суммирование по модулю p_i выходных значений просмотровых таблиц, а вторая – суммирует остатки, сформированные в результате работы первой ступени. На выходе второй ступени нейронной сети конечного кольца формируется остаток исходного двоичного числа $\left|X[n]\right|_{p_i}$.

Восстановление числа x по остаткам $\{x_1, x_2, ..., x_k\}$ основано на Китайской Теореме об Остатках (КТО) [2]

$$X = \left| \sum_{i=0}^{k} \left| P_i^{-1} \right|_{p_i} x_i \right|_{p_i} P_i \Bigg|_P , \tag{1.26}$$

где $P_i = \frac{P}{p_i}$. Элемент $\left|P_i^{-1}\right|_{p_i}$ означает мультипликативный обратный для P_i, по модулю p_i.

Существует, однако, и другой способ преобразования числа из СОК в позиционную форму, основанный на применении обобщенной позиционной системы счисления [110]. Число $X < P$ имеет вид $\{x_1', x_2', ..., x_k'\}$, $0 < x_i' \le p_i$ в обобщенной позиционной системе счисления, если:

$$X = x_1' + x_2' p_1 + x_3' p_1 p_2 + ... + x_k' \prod_{i=1}^{k-1} p_i , \tag{1.27}$$

где $x_i' \in [0, p_i)$ цифры числа X в обобщенной позиционной системе счисления, и:

$$\begin{gathered} x_1' = x_1 \bmod p_1 \\ x_2' = (x_2 - x_1')c_{12} \bmod p_2 \\ x_3' = ((x_3 - x_1')c_{13} - x_2')c_{23} \bmod p_3 \\ \dots \\ x_k' = (..((x_k - x_1')c_{1k} - x_2')c_{2k} - ... - x_{k-1}')c_{k-i,k} \bmod p_k . \end{gathered} \tag{1.28}$$

Константы c_{ij} являются мультипликативными обратными элементами для p_i по модулю p_j для всех $1 \le i \le j \le k$, то есть $c_{ij} \cdot p_i \equiv 1 \bmod p_j$ для $1 \le i \le n$, и могут быть вычислены, например, с помощью алгоритма Евклида.

Как говорилось выше, для обратного преобразования числа из СОК в позиционную систему счисления требуется применение КТО.

В [21] приводится модификация Китайской теоремы об остатках (КТОм), которая заключается в следующем:

Если обе стороны уравнения (1.27) поделить на P, то мы имеем следующее уравнение:

$$\tilde{X} = \frac{X}{P} = \left| \sum_{i=1}^{n} \left(\frac{\left| P_i^{-1} \right|_{p_i}}{p_i} \right) \cdot x_n \right|_1 = \left| \sum_{i=1}^{n} k_i x_i \right|_1 \tag{1.29}$$

где $k_i = \left(\left| P_i^{-1} \right|_{m_i} \right) / p_i$, $i = 1,2,\ldots,n$ это константы, которые могут быть вычислены заранее. Значение $\tilde{X}$ может рассматриваться в качестве позиционного представления числа X, которые связаны между собой соотношением:

$$X = \tilde{X} P \tag{1.30}$$

В работе [22] показано, что аппаратная реализация КТОм возможна на практике, когда значения k_i аппроксимированы константами $k_i^{'}$, в соответствии с уравнением

$$k_i^{'} = \left\lceil \left(\frac{\left| P_i^{-1} \right|_{p_i}}{p_i} \right) \cdot 2^N \right\rceil, i = 1,2,\ldots,n \tag{1.31}$$

где $\lceil \cdot \rceil$ функция округления до ближайшего целого вверх. Параметр N КТОм вычисляется из уравнения

$$N = \left| \log_2 P + \log_2 \sum_{i=1}^{n} (p_i - 1) \right| - 1 \tag{1.32}$$

При этом константы $k_i^{/}$ используются для вычисления $X^{/}$ по формуле (1.33), которая также применяется для выполнения обратного преобразования $X^{/}$ с помощью выражения $X = \lfloor (X^{/} P) / 2^N \rfloor$, где $\lfloor \cdot \rfloor$ это функция округления до ближайшего целого вниз.

$$X^{/} = \left| \sum_{i=1}^{n} k_i^{/} x_i \right|_{2^N} \tag{1.33}$$

В работе [22] также показано, что модифицированная КТО обеспечивает снижение задержки для процедуры обратного преобразования, по сравнению с традиционной КТО и MRC.

Преимущества представления чисел в СОК могут быть представлены следующим образом [56,72]:

1. Так как в СОК отсутствует распространение переноса между арифметическими блоками и числа большой размерности представляются в виде небольших остатков, это приводит к ускорению в обработке данных.

2. При представлении данных с использованием СОК, числа большой размерности кодируются в набор небольших остатков, соответственно уменьшается сложность арифметических устройств в каждом канале модуля, что облегчает и упрощает работу вычислительной системы.

3. СОК является непозиционной системой без отсутствия зависимости между своими арифметическими блоками, следовательно, ошибка в одном канале не распространяется на другие, что в свою очередь облегчает процесс обнаружения и исправления ошибок.

Таким образом, применение СОК позволяет упростить и уменьшить архитектуру вычислительных электронных устройств, за счет чего повышается не только скорость, но и энергетическая эффективность продуктов.

Однако, такие операции как деление, сравнение двух чисел, обнаружение знака являются трудоемкими и дорогими в СОК. Для этих проблемных операций были предложены многие решения. Большинство из них, состоит в преобразовании остатка в двоичную систему (обратное преобразование). С другой стороны, выбор правильного набора модулей является еще одним важным вопросом для построения эффективной СОК с достаточным динамическим диапазоном [66,84,102,103,104,105].

Подводя некоторые итоги, можно отметить, что система остаточных классов позволяет существенно улучшить параметры вычислительного аппарата в цифровой системе видеонаблюдения по сравнению с вычислительным аппаратом, построенным на той же физико-технологической базе, но в позиционной системе

счисления, а также получить новые более прогрессивные конструктивные и структурные решения. Таким образом, система остаточных классов, обладающая всеми указанными достоинствами в применении, является наиболее подходящей системой при разработке фильтров для объектов в цифровых системах видеонаблюдения.

1.8 Математическая постановка задачи исследования

Анализ проблем работы систем цифрового видеонаблюдения показал, что основной задачей, стоящей перед такими системами, является задача обработки изображений большой размерности в режиме реального времени. Однако, возникает противоречие между практической потребностью обработки изображений высокого разрешения в режиме реального времени и ограниченностью аппаратных и энергетических ресурсов современных мобильных устройств. Таким образом, необходимо обеспечить повышение скорости выполнения арифметических вычислений за счет разработки новых математических моделей цифровых фильтров, алгоритмы которых используют методы, сокращающие время их работы.

Наиболее популярными из современных разрешающих способностей дисплеев и видеостандартов, являются: 720 HD (1280×720) точек; Full HD (1920×1080) точек; Ultra HD 4k (3840×2160) точек или Ultra HD 8k (7680×4320) точек [21].

С точки зрения математической постановки задач распознавания образов. постановку диссертационного исследования можно сформулировать следующим образом. Пусть U – множество образов в конкретной задаче распознавания. Тогда x – это отдельный образ из этого множества, который может характеризоваться бесконечным числом признаков. Для формирования алфавита признаков выберем некоторое подмножество признаков X (конечное), которое называется пространством признаков.

Пусть x – элемент пространства X, соответствующий образу $x \in U$, а $H : U \to X$ оператор отображающий x в x. При этом $X = P(U)$ пространство признаков. Предположим, что во множестве образов U нас интересуют некоторые подмножества – классы. Множество классов $\Omega = \{\omega_1, \omega_2, ..., \omega_m\}$ является конечным и классы образуют полную группу подмножества из U, т.е. $\bigcup_{i=1}^{m} \varpi_i = U$ для всех $i \neq j$.

Тогда, для того, чтобы классифицировать образ $x \in U$ по классам $\omega_1, ..., \omega_m$ из пространства признаков необходимо найти индикаторную функцию $\tilde{g}$: $X \to Y$, $Y = \{y_1, ..., y_m\}$, которая ставит в соответствие каждому вектору $x = Px \in X$ метку $y_i \in Y$ того класса ϖ_i, которому принадлежит соответствующий образ, т.е. $\tilde{g}(x) = y_i$, если $x = Px$, $x \in \varpi_i$. На этапе обучения системе распознавания доступна информация о классах в виде некоторого множества пар (x_j, y_j) (прецеденты), $j = 1, ..., N$, где $x_j = Px_j$, $y_j = g(x_j) \in Y$, где $\Xi = \{x_1, ..., x_N\}$ множество обучающей выборки. Тогда по множеству прецедентов $(\Xi, Y) = \{(x_j, y_j) : j = 1, ..., N\}$ требуется найти оптимальную аппаратную реализацию, которая позволит минимизировать время работы алгоритма распознавания и аппаратные затраты FPGA, измеряемые в количестве используемых LUT-таблиц и Слайсов.

С математической точки зрения, задачей исследования является разработка системы распознавания образов на основе сверточной нейронной сети, использующей вычисления СОК. Важнейшим параметром вычислительной системы, построенной с использованием СОК, является выбор количества модулей СОК, их вида и диапазона, который обеспечивают выбранные модули. Наиболее эффективным модулем СОК с точки зрения производительности является модуль вида 2^n, несколько менее эффективными являются модули вида $2^n - 1$. Следующим по эффективности являются модули вида $2^n + 1$ [124]. Остальные модули являются наименее эффективными и называются модулями общего вида.

Пусть $p_1, p_2, ..., p_n$ модули СОК, тогда $P = p_1 p_2 ... p_n$ динамический диапазон СОК. Оценка качества и эффективности систем цифрового видеонаблюдения основана на анализе времени выполнения алгоритма T , измеряемой в наносекундах (нс), количества LUT-таблиц, которые при использовании FPGA можно оценить на основе задействованных для реализации алгоритма основных структурных блоков платы – блоков Слайс, объединяющих различные по функционалу вычислительные и логические единицы. В случае использования для реализации различных вариантов модулей СОК $p_1, p_2, ..., p_n$ будут изменяться характеристики работы устройства в целом. Следовательно, в данном случае можно говорить о зависимости скорости работы и занимаемой площади от количества и вида выбранных модулей СОК. Основной задачей данного исследования является выбор рационального набора модулей СОК, обеспечивающего максимально быстрое выполнение аппаратных процессов распознавания образов встраиваемой системы видеонаблюдения.

При этом важно учесть не только снижение скорости работы T , но и сохранение или снижение занимаемой площади устройства S , которая также зависит от набора модулей СОК.

На основе проведенного анализа сформулируем научную задачу исследования в соответствии с темой и целью диссертационной работы.

Целью диссертационного исследования является разработка математических моделей, методов и алгоритмов реализации повышения скорости работы цифровых фильтров обработки изображений систем видеонаблюдения на основе интеграции системы остаточных классов и искусственных нейронных сетей.

Объект исследования – цифровые системы видеонаблюдения.

Предмет исследования – математические методы, модели и алгоритмы вычислительных средств объектов цифровой обработки изображений.

Научная задача – разработка новых математических моделей цифровых фильтров изображений, численных методах вычисления немодульных операций, а

также комплекса программ, применение которых позволит увеличить скорость работы системы.

1.9 Выводы по первой главе

Основные результаты настоящей главы состоят в следующем:

1. Установлено, что актуальной проблемой цифровой обработки изображений в современных системах видеонаблюдения является задача распознавания образов. Качество решения данной задачи зависит от визуального качества изображения, а также методов и алгоритмов его улучшения.

2. Анализ современных методов цифровой обработки изображений, применяемых для решения задачи распознавания образов, показал, что перспективным инструментом для решения данной задачи являются сверточные нейронные сети, использующие фильтры цифровой обработки изображений в своей архитектуре.

3. Математической моделью фильтров цифровой обработки изображений является операция свертки, которая представляет собой поэтапное суммирование произведений отсчетов сигнала или пикселей на коэффициенты фильтра. Использование только этих операций сложения и умножения обосновывает применение системы остаточных классов для реализации фильтров цифровой обработки изображений, так как основным достоинством модулярной арифметики является способность быстрого и параллельного выполнения этих операций на арифметическом уровне.

4. Сформулирована основная задача исследования, которая заключается в разработке высокопроизводительных моделей, методов и алгоритмов решения задачи распознавания образов, основанных на применении сверточных нейронных сетей с использованием вычислений в системе остаточных классов.

ГЛАВА 2. РАЗРАБОТКА МАТЕМАТИЧЕСКИХ МОДЕЛЕЙ, МЕТОДОВ И АЛГОРИТМОВ ЦИФРОВОЙ ОБРАБОТКИ ИЗОБРАЖЕНИЙ В СОК

2.1 Архитектурная организация процесса цифровой обработки изображений с вычислениями в СОК

Ранее было сказано, что цифровую фильтрацию определяет используемый в процессе цифровой фильтр. Говоря о процессе цифровой фильтрации (обработки) изображений следует отметить, что в данном случае цифровой фильтр позволяет накладывать на изображение различные эффекты, например: размытие, резкость, деформация, шум и т.д [36]. При этом под цифровым фильтром понимается математический алгоритм обработки изображения [11]. Большая группа цифровых фильтров имеет один и тот же алгоритм, но эффект, накладываемый фильтром на изображение, зависит от коэффициентов, используемых в алгоритме. Чаще всего в процессе цифровой обработки изображений используют фильтры с конечной импульсной характеристикой, основанные на теории линейных систем и применении двумерных сверток [62].

Под сверткой понимается способ представления какого-либо векторного значения скалярным значением. Другими словами, свертка – это математическая операция, заключающаяся в интегральном преобразовании функций f и g, в результате которой порождается третья функцию. Если $f, g: R^d \to R$ – функции, интегрируемые относительно меры Лебега на пространстве R^d, тогда их сверткой называется функция $f * g: R^d \to R$, определенная формулой (2.1):

$$(f * g)(x) \overset{def}{=} \int_{R^d} f(y)g(x-y)dy = \int_{R^d} f(x-y)g(y)dy \qquad (2.1)$$

Применительно к обработке изображений векторное значение представляет собой цвет группы пикселей, а скалярное значение, получаемое на основе свертки, представляет собой цвет пикселя, получаемого в результате применения к исходному изображению какого-либо эффекта.

Таким образом, чтобы применить фильтр к изображению необходимо пройтись по изображению точка за точкой. При этом, каждый из пикселей рассматривается вместе с матрицей, центральным элементом которой он является. Перемножаем соответствующие значения двух матриц и их сумму присваиваем рассматриваемой точке. Схематично процесс свертки (фильтрации) изображения показан на рисунке 2.1.

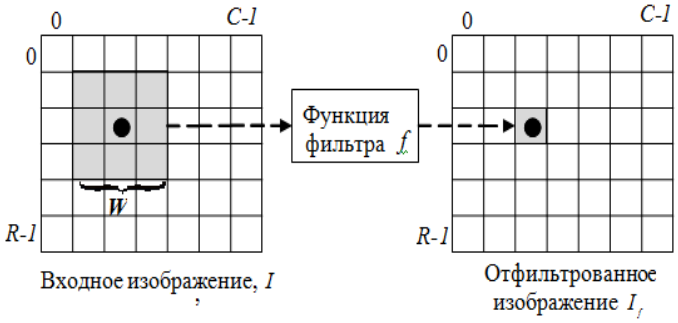


Рисунок 2.1. Схема процесса цифровой фильтрации (свертки) изображения

Поскольку, математическая свертка включает в себя только операции сложения и умножения и работает с дискретными величинами, то целесообразно перейти к выражению расчета операции свертки по формуле (2.2):

$$C_{new}[i][j] = \sum_{k=0}^{m}\sum_{l=0}^{n} a_{k,l} C_{old}\left[i - \frac{m}{2}\right]\left[j - \frac{n}{2}\right] \tag{2.2}$$

где m и n – это константы фильтра, которые задают двумерный размер фильтра, a – коэффициенты фильтра, определяющие тот эффект, который накладывает фильтр. Масочная фильтрация изображения может быть представлена в виде (2.3):

$$I_f(x,y) = \sum_{i=0}^{d-1}\sum_{j=0}^{d-1} W_{i,j} I(x+i, y+j), \tag{2.3}$$

где I_f - обработанное изображение, а

$$W = \begin{bmatrix} w_{0,0} & \cdots & w_{0,d-1} \\ \vdots & \vdots & \vdots \\ w_{d-1,0} & \cdots & w_{d-1,d-1} \end{bmatrix} \tag{2.4}$$

(5) – маска фильтра, размерностью $d \times d$.

Обработка цифрового изображения с использованием СОК состоит из трех этапов. На первом этапе происходит перевод значений пикселей из представления ПСС в СОК. После чего происходит обработка изображения цифровым фильтром (иными словами свертка). Цифровой фильтр проходит по изображению точка за точкой, в результате чего перемножаются соответствующие значения двух матриц, а их сумма присваивается рассматриваемой точке. Происходит эта операция по каждому модулю отдельно и в параллельном режиме. На третьем этапе происходит обратное преобразований значений из СОК в представление ПСС.

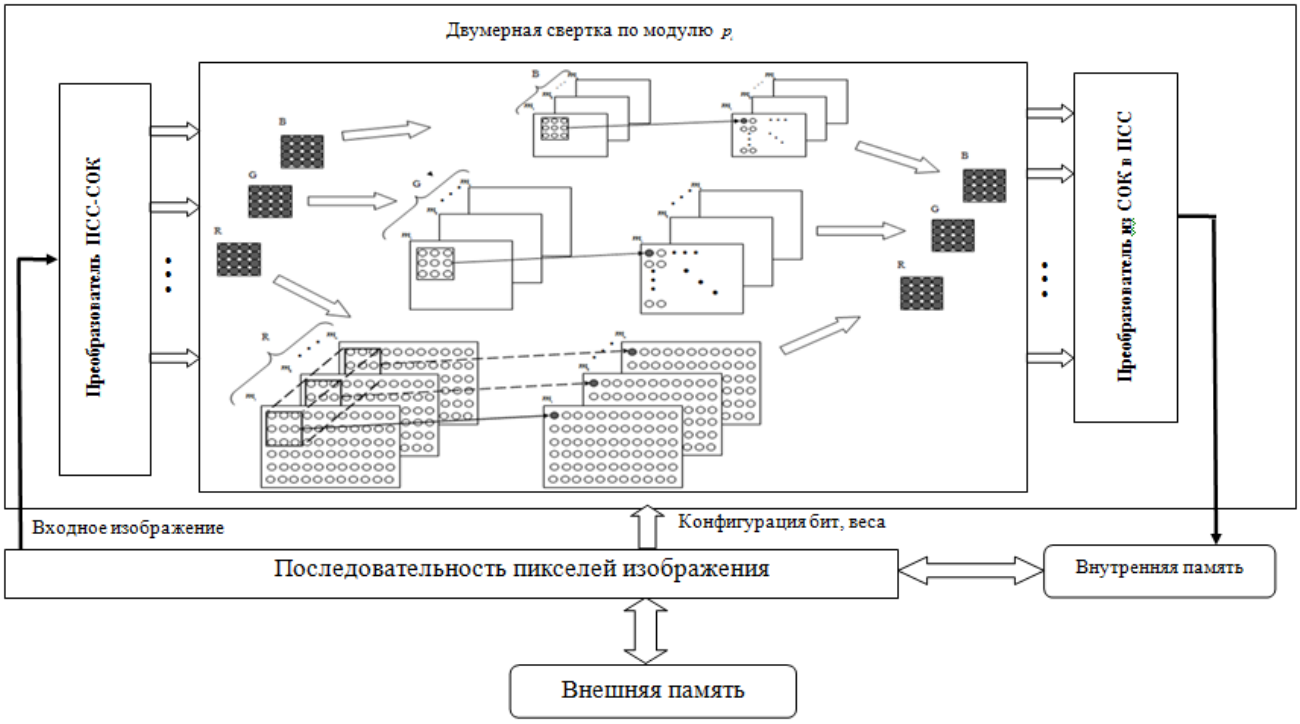


Рисунок 2.2. Архитектурная организация процесса цифровой обработки изображений с вычислениями в СОК

На рисунке 2.2. показана схема функционирования СНС на основе вычислений в СОК. Схема состоит из блока, в котором происходит преобразования из ПСС в СОК, блока двумерной свертки, блока, в котором происходит обратная операция преобразования из СОК в ПСС, а также блоков внутренней и внешней памяти. Особое место в представленной схеме занимает

блок двумерной свертки, поскольку он характеризуется набором цифровых фильтров, которые применяются в ней. Более подробно данная схема будет рассмотрена в следующей главе, а также представлена ее модифицированная версия.

В данной главе будут рассмотрены принципы работы цифрового фильтра на конкретных изображениях, а также показано каким образом ограничения, накладываемые на динамический диапазон системы счисления, могут приводить к некорректным результатам работы цифровых фильтров.

2.2 Математические модели фильтров для цифровой обработки изображений

Любой цифровой фильтр на основе двумерной свертки характеризуется размером группы пикселей, т.е. размером фильтра, а также своей импульсной характеристикой. В ЦОИ импульсная характеристика фильтра это изображение, получаемое в результате обработки черного изображения, в центре которого располагается белая точка. Конечность импульсной характеристики определяется конечным размером группы пикселей, используемых в фильтре. В свою очередь, импульсная характеристика зависит от размера фильтра, который определяют коэффициенты фильтры [1,12]. Коэффициенты фильтра представляют собой некоторые скалярные значения, на которые умножаются значения цветов пикселей из группы, соответствующей размеру фильтра. Обработка изображения с применением такого рода фильтров описывается следующей формулой [11]:

$$C_{new}[i][j] = \sum_{k=0}^{m}\sum_{l=0}^{n} a_{k,l} C_{old}\left[i - \frac{m}{2}\right]\left[j - \frac{n}{2}\right] \qquad (2.5)$$

где m и n – это константы фильтра, которые задают двумерный размер фильтра, a – коэффициенты фильтра, определяющие тот эффект, который накладывает фильтр. Следует отметить, что размер фильтра всегда является нечетным. Нечетный размер маски фильтра обусловлен с одной стороны тем, что фильтрация изображения является пространственной фильтрацией, а такая фильтрация требует работы с ядром массива изображения. Нечетный размер

маски фильтра, в свою очередь, хорошо устанавливает взаимосвязь между соседними пикселями изображения. С другой стороны, размер маски фильтра прямо пропорционально влияет на количество математических операций, требуемых в процессе свертки изображения. По этой причине, чаще всего в процессе фильтрации используют размеры маски 3×3 или 5×5 [11,122].

Таким образом, основной задачей при разработке цифровых фильтров с конечной импульсной характеристикой является расчет коэффициентов фильтра. Рассмотрим типовые примеры фильтров, используемые в ЦОИ.

В соответствии с поставленной задачей фильтрации различают [122]:

1. Сглаживающие фильтры. Простым примером сглаживающего фильтра (фильтр уменьшения резкости) является фильтр радиуса r, который задается при помощи матрицы размера $(2r+1)\times(2r+1)$, все значения которой равны $\frac{1}{(2r+1)^2}$, а сумма значений равна единице. Такой фильтр является двумерным аналогом низкочастотного одномерного П-образного фильтра скользящего среднего. При фильтрации с таким ядром значение пикселя заменяется усредненным значением пикселей в квадрате со стороной $2r+1$ вокруг него. Примером такого фильтра является биномиальный, Гауссов или простой фильтр.

Биномиальные фильтры содержат значения дискретного биномиального распределения. Эти фильтры строятся с помощью последовательной свертки с маской 1/2[1 1], что эквивалентно вычислительной схеме треугольника Паскаля (таблица 2) [14].

Таблица 2.1 – Схема треугольника Паскаля, где R – порядок бинома, f – масштабный множитель 2^{-R}, σ^2 – дисперсия

R	f		σ^2
0	1	1	0
1	1/2	1 1	1/4
2	1/4	1 2 1	1/2
3	1/8	1 3 3 1	3/4
4	1/16	1 4 6 4 1	1

Двумерные биномиальные фильтры могут быть составлены из горизонтального и вертикального одномерных фильтров:

$$B^R = B_x^R B_y^R \tag{2.6}$$

Пример 2.1. Предположим, что $R = 4$, тогда одномерные биномиальные фильтры имеют вид:

$$B_x^4 = \frac{1}{16}[1 \quad 4 \quad 6 \quad 4 \quad 1], \; B_y^4 = \frac{1}{16}\begin{bmatrix} 1 \\ 4 \\ 6 \\ 4 \\ 1 \end{bmatrix}.$$

Составим двумерный биномиальный фильтр следующим образом:

$$B^4 = B_x^4 B_y^4 = \frac{1}{16}[1 \quad 4 \quad 6 \quad 4 \quad 1] \times \frac{1}{16}\begin{bmatrix} 1 \\ 4 \\ 6 \\ 4 \\ 1 \end{bmatrix} = \frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}.$$

Простой сглаживающий фильтр заменяет значение каждого пикселя исходного изображения средним значением соседних пикселей, включая себя. Это фильтр основан на ядре, которое представляет форму и размер учитывающейся окрестности [22].

$$F_1 = \frac{1}{9}\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

Одно из возможных применений такого фильтра – шумоподавление, т.е. решение задачи восстановление исходного изображения, к пикселям которого добавлен случайный шум. При этом, шум меняется независимо от пикселя к пикселю и, при условии, что математическое ожидание значения шума равно нулю, шумы соседних пикселей будут компенсировать друг друга. Чем больше окно фильтрации, тем меньше будет усредненная интенсивность шума, однако при этом будет происходить и существенное размытие значащих деталей изображения. Образом белой точки на черном фоне при фильтрации таким

фильтром будет равномерно серый квадрат [122]. Работа фильтра F_1 продемонстрирована на рисунке 2.3.

а)  б) 

Рисунок 2.3. а) Изображение «девушка» до обработки; б) Изображение «девушка» после обработки сглаживающим фильтром F_1

Более эффективного шумоподавления можно добиться, применив Гауссовский фильтр, задающейся формулой:

$$F_{gauss}(i,j) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{i^2+j^2}{2\sigma^2}\right) \tag{2.7}$$

где, σ – степень размытия изображения. Влияние пикселей друг на друга в таком фильтре уменьшается с расстоянием, что и обуславливает эффективность в его применении. Гауссовская фильтрация также является сглаживающей. Образом белой точки на черном фоне будет симметричное размытое пятно, с убыванием яркости от середины к краям [118]. Двумерная функция Гауссова (рисунок 2.4) фильтра выглядит следующим образом:

$$G(x,y) = \frac{1}{\sqrt{2\pi\sigma^2}}e^{-(x^2+y^2)/2\sigma^2}. \tag{2.8}$$

2. Контрастоповышающие фильтры. Если сглаживающие фильтры снижают локальную контрастность изображения, размывая его, то контрастоповышающие фильтры (фильтры повышения резкости) производят обратный эффект. Говоря о контрастоповыющих фильтрах можно сказать, что они являются фильтрами высоких пространственных частот. Ядро такого фильтра

имеет значение, большее 1, в точке $(0,0)$. Общая сумма всех значений равна 1. Примером такого фильтра является:

$$F_2 = \begin{pmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{pmatrix}$$

Фильтр F_2 известен еще как фильтр Лапласса. Эффект повышения резкости достигается за счет того, что фильтр подчеркивает разницу между интенсивностями соседних пикселей, удаляя эти интенсивности друг от друга. Причем, эффект резкости будет тем сильнее, чем больше значение центрального члена ядра. Признаком применения такого фильтра является получение на изображении заметных светлых и менее заметных темных ореолов границ [122]. На рисунке 2.5 приведен результат работы фильтра повышения резкости.

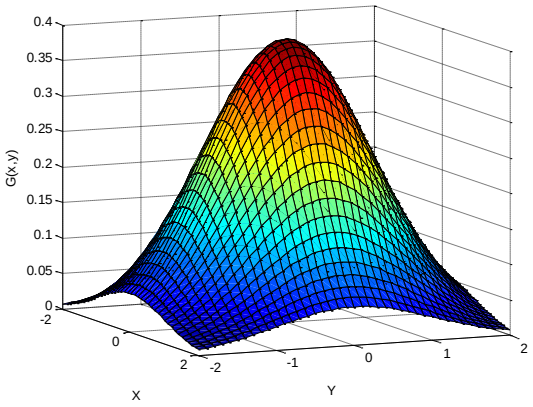


Рисунок 2.4. – График двумерной функции фильтра Гаусса [3]

а)  б) 

Рисунок 2.5. а) Изображение «девушка» до обработки; б) Изображение «девушка» после обработки фильтром повышения резкости F_2

3. Разностные фильтры. Данные фильтры являются линейными фильтрами, которые задаются дискретными аппроксимациями дифференциальных операторов, на основе метода конечных разностей. Разностные фильтры играют важную роль во многих приложениях и чаще всего применяются для поиска границ на изображении.

Простейшим дифференциальным оператором является взятие производной $\frac{\partial}{\partial x}$ по координате x. Данный оператор определен для непрерывных функций. Существует множество способов определить аналогичный оператор для дискретных изображений при помощи линейного фильтра. Наиболее известными примерами разностных фильтров являются фильтры Превитта (Prewitta) и Собеля (Sobel) [118,122].

Пример фильтра Превитта (вертикальный):

$$F_3 = \begin{pmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{pmatrix}$$

Пример фильтра Собеля (вертикальный):

$$F_4 = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}$$

Аналогичные фильтры, которые приближают оператор производной $\frac{\partial}{\partial y}$ по координате y, получается путем транспонирования матриц.

Выделение контуров основывается на дифференцировании в той или иной форме. Для первой частной производной в направлении x можно использовать одну из аппроксимаций, представленных в таблице 2.2 для $\frac{\partial g(x_1, x_2)}{\partial x_1}$, где нижний индекс обозначает центральный пиксель асимметричных масок [118]. Контуры дают пересечения нулевого уровня во вторых производных, следовательно,

вторые производные по всем направлениям можно просто сложить для образования оператора Лапласа:

$$L = D_x^2 + D_y^2. \tag{2.9}$$

Таблица 2.2. – Дискретные разности первого порядка

Разность	Обозначение	Формула	Маска фильтра
Левая	${}^{-}D_x$	$\frac{g(x_1,x_2)-g(x_1-\Delta x_1,x_2)}{\Delta x_1}$	$[1. \quad -1]$
Правая	${}^{+}D_x$	$\frac{g(x_1+\Delta x_1,x_2)-g(x_1,x_2)}{\Delta x_1}$	$[1 \quad -1.]$
Центральная	D_{2x}	$\frac{g(x_1+\Delta x_1,x_2)-g(x_1-\Delta x_1,x_2)}{2\Delta x_1}$	$\frac{1}{2}[1 \quad 0 \quad -1]$

Дифференциальные операторы второго порядка можно получить с помощью двукратного применения операторов первого порядка:

$$D_x^2 = {}^{-}D_x \, {}^{+}D_x. \tag{2.10}$$

В пространственной области это означает

$$[1. \quad -1] \times [1 \quad -1.] = [1 \quad -2 \quad 1].$$

Таким образом, из выражений (8) и (9) следует, что дискретный оператор Лапласа имеет маску фильтра [14]:

$$L = [1 \quad -2 \quad 1] + \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}.$$

Существует много других способов построения дискретной аппроксимации для оператора Лапласа. Интересной возможностью являются биномиальные маски:

$$L' = 4(B^R + I). \tag{2.11}$$

Пример 2.2. Пусть $R = 4$, тогда, исходя из выражения (2.11), оператор Лапласа с помощью биномиальной маски можно получить следующим образом:

$$L' = 4\left(B^4 - I\right) = 4\left(\frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix} - \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}\right) =$$

$$= \frac{1}{64}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & -220 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

В отличии от сглаживающих контрастоповышающих фильтров, не меняющих среднюю интенсивность изображения (сумма элементов ядра равна единице), применение разностных фильтров дает изображение со средним значением пикселя близким к нулю (сумма элементов ядра равна нулю). Вертикальным перепадам (границам) исходного изображения соответствуют пиксели с большими по модулю значениями на результирующем изображении. Поэтому разностные фильтры хорошо подходят для решения задачи нахождения границ изображения [118]. На рисунке 2.6 и 2.7 приведены результаты работы разностных фильтров Превитта и Собеля.

а)  б) 

Рисунок 2.6. а) Изображение «девушка» до обработки; б) Изображение «девушка» после обработки разностным фильтром Превитта F_3

а)  б) 

Рисунок 2.7. а) Изображение «девушка» до обработки; б) Изображение «девушка» после обработки разностным фильтром Собеля F_4

Аналогично выше рассмотренным фильтрам, используя метод конечных разностей можно составить фильтры для других дифференциальных операторов. Например, важный для многих приложений дифференциальный оператор Лапласа (лапласиан) $\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$ можно приблизить для дискретных изображений фильтром с матрицей:

$$F_5 = \begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

Матрица вида F_5 является одним из вариантов представления фильтра Лапласа.

Фильтры увеличения резкости являются высокоимпульсными фильтрами. Они используется, чтобы подчеркнуть высокочастотные составляющие, представляющие детали изображения без устранения низкочастотных компонентов [122]:

$$\begin{aligned} H &= A \times orig - l = \\ &= (A-1) \times orig + (orig - l) = \\ &= (A-1) \times orig + h \end{aligned}, \qquad (2.12)$$

где A - коэффициент усиления, $orig$ - исходное изображение, l - фильтр низких частот и h - фильтр высоких частот.

Пример 2.3. Пусть $A = 1$, а l - дискретный оператор Лапласа, тогда

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Пример 2.4. Пусть $A = 2$, а h - высокочастотный фильтр выделения контуров размерностью 3×3, тогда

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

В результате применения дискретного фильтра Лапласа большие по модулю значения соответствуют как вертикальным, так и горизонтальным перепадам яркости. Таким образом, фильтр F_5 является фильтром, находящим границы любой ориентации [118,122]. На рисунке 2.8 приведен результат работы разностного фильтра.

а)  б) 

Рисунок 2.8. а) Изображение «девушка» до обработки; б) Изображение «девушка» после обработки разностным фильтром Лапласа F_5

Однако, применение такого фильтра имеет существенные недостатки. Поскольку, нахождение границ на изображении может производиться путем применения этого фильтра и взятия всех пикселей, модуль значения которых превосходит некоторый порог, то из этого следует проблема неопределенности в выборе величины порога. Для разных частей изображения приемлемый результат обычно получается при существенно разных пороговых значениях. Еще одним

недостатком разностных фильтров является повышенная чувствительность к шумам изображения [11].

Более полный список распространенных в применении фильтров приведен в таблице 2.3.

Таблица 2.3. Матрицы коэффициентов фильтров изображений

Название фильтра	Матрица коэффициентов
Градиентный фильтр Превитта (вертикальный)	$\begin{pmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{pmatrix}$
Градиентный фильтр Превитта (горизонтальный)	$\begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{pmatrix}$
Градиентный фильтр Собеля (вертикальный)	$\begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}$
Градиентный фильтр Собеля (горизонтальный)	$\begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}$
Фильтр высоких частот Лапласа (3×3)	$\begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix}$
Фильтр высоких частот Лапласа (5×5)	$\begin{pmatrix} -1 & -3 & -4 & -3 & -1 \\ -3 & 0 & 6 & 0 & -3 \\ -4 & 6 & 20 & 6 & -4 \\ -3 & 0 & 6 & 0 & -3 \\ -1 & -3 & -4 & -3 & -1 \end{pmatrix}$
Фильтр низких частот Гаусса (3×3)	$\frac{1}{16}\begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}$

Фильтр низких частот Гаусса (5×5)	$\frac{1}{571}\begin{pmatrix} 2 & 7 & 12 & 7 & 2 \\ 7 & 31 & 52 & 31 & 7 \\ 12 & 52 & 127 & 52 & 12 \\ 7 & 31 & 52 & 31 & 7 \\ 2 & 7 & 12 & 7 & 2 \end{pmatrix}$
Фильтр высоких частот (5×5)	$\begin{pmatrix} -1 & -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 & -1 \\ --1 & -1 & 24 & -1 & -1 \\ -1 & -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 & -1 \end{pmatrix}$
Фильтр повышения резкости	$\begin{pmatrix} -1 & -1 & -1 \\ -1 & 16 & -1 \\ -1 & -1 & -1 \end{pmatrix}$

Многообразие получаемых результатов при применении различных фильтров цифровой обработки изображений дает выбирать соответствующую модель фильтра в зависимости от поставленной решаемой задачи [25]. Численная реализация процесса фильтрации будет показана в пункте 2.2.

2.3 Разработка архитектурной организации процесса фильтрации изображения

Численная организация процесса фильтрации напрямую связана с его структурной схемой. Поэтому, говоря о фильтрации изображения особое место должна занимать схема этого процесса, которая должна включать в себя поэтапно каждый процесс.

Для выполнения операций в СОК необходимо первоначально преобразовать числа из позиционной системы счисления в систему остаточных классов, т.е. применить операцию прямого преобразования ПСС-СОК [2]. Таким образом, схема, на основе которой должно проходить это преобразование должна сначала получить значения пикселей, после этого преобразовать их в представление СОК, а затем выполнять цифровую обработку изображений, а именно цифровую

фильтрацию. После прохождения описанной операции к числам, представленным в СОК, должна быть применена обратная операция для получения информации в ПСС. Исходя из этого, в [121] авторы представляют операцию фильтрации изображения как процесс, состоящий из трех этапов. Каждый из этих этапов описывается следующим образом:

1. Преобразование значений пикселей из позиционной системы счислению в систему остаточных классов. Этот этап включает в себя решение нескольких задач: чтение значения пикселей входного цифрового изображения в виде десятичных или двоичных чисел; выбор соответствующего набора модулей СОК, который удовлетворяет ограничению, накладываемому на динамический диапазон системы; создание справочной LUT-таблицы для более быстрой обработки; присваивание новых значений пикселей в соответствии с СОК, полученных на этом этапе. Диаграмма, иллюстрирующая описанный процесс в [121] представлена на рисунке 2.9.

2. Операция фильтрации изображения с вычислениями в СОК.

3. Обратное преобразование полученных значений пикселей из СОК в ПСС. Этап включает в себя: получение значений пикселей в виде остатков в СОК; определение числа остатков; преобразование каждого остатка в свою первоначальную форму с применением LUT-таблиц. На рисунке 2.10 представлена схема процесса преобразования из ПСС в СОК.

Особое место в описанном процессе прямого и обратного преобразований из ПСС в СОК и обратно отводится просмотровым LUT-таблицам, так как они повышают работу всей системы в целом. Используются таблицы во время перехода значений пикселей из ПСС в СОК и обратно, что естественно приводит к более быстрому преобразованию значений. Покажем свою обобщенную модифицированную схему, придерживаясь которой можно получить обработанное цифровое изображение (рисунок 2.11).

Отличительной особенностью предлагаемой схемы является то, что в ней учтен подготовительный этап фильтрации, в процессе которого происходит выбор маски фильтра и выбор набора модулей СОК, который занимает одно из важных

мест, в процессе обработки изображений. Причем эти операции зависят от изначально поставленной задачи обработки. В [121] выбор набора модулей СОК происходит уже после чтения значений пикселей изображений, что может привести к получению некорректных результатов работы системы, обработки изображений [63,84].

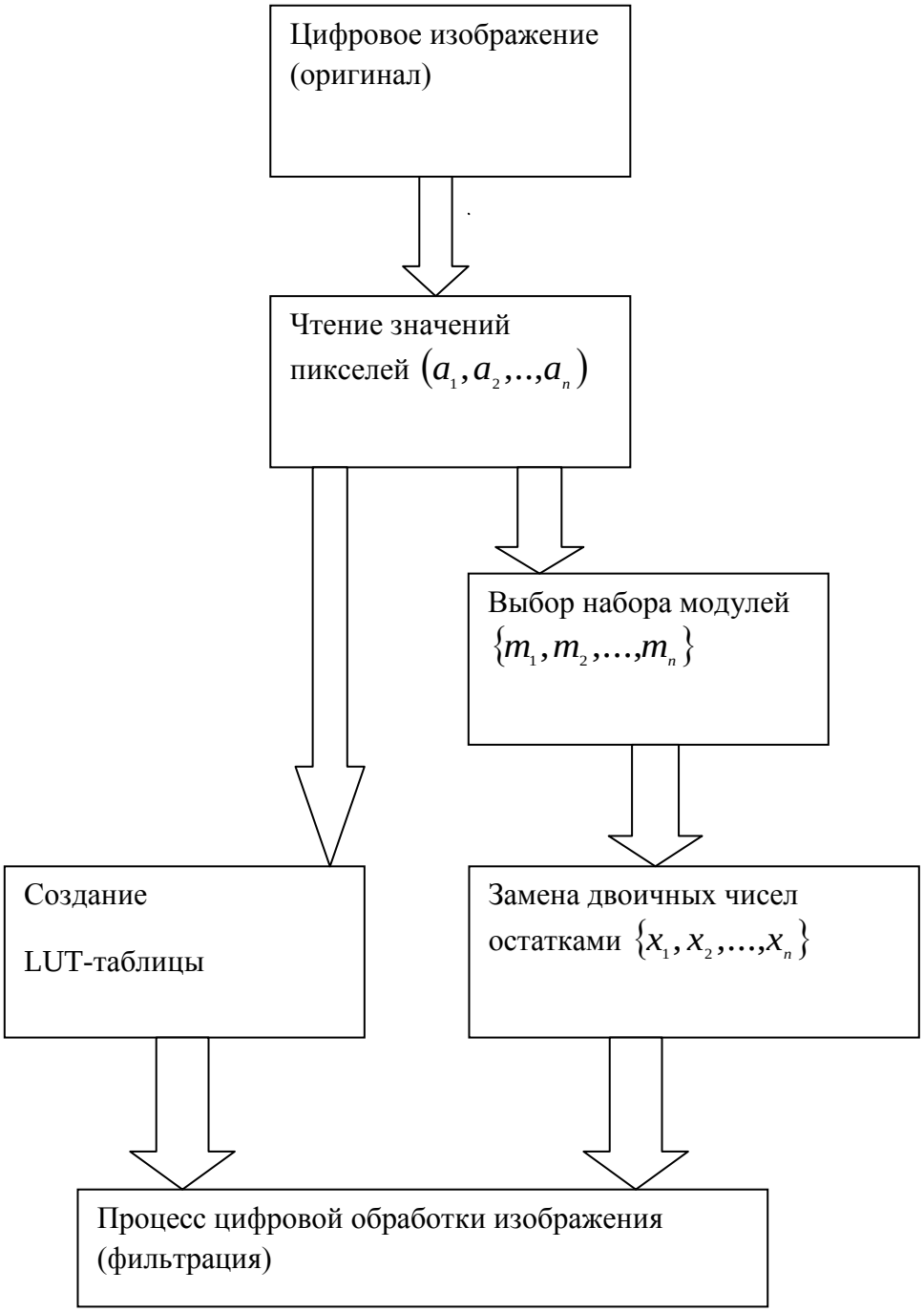


Рисунок 2.9. Схема прямого преобразователя ПСС-СОК согласно [121]

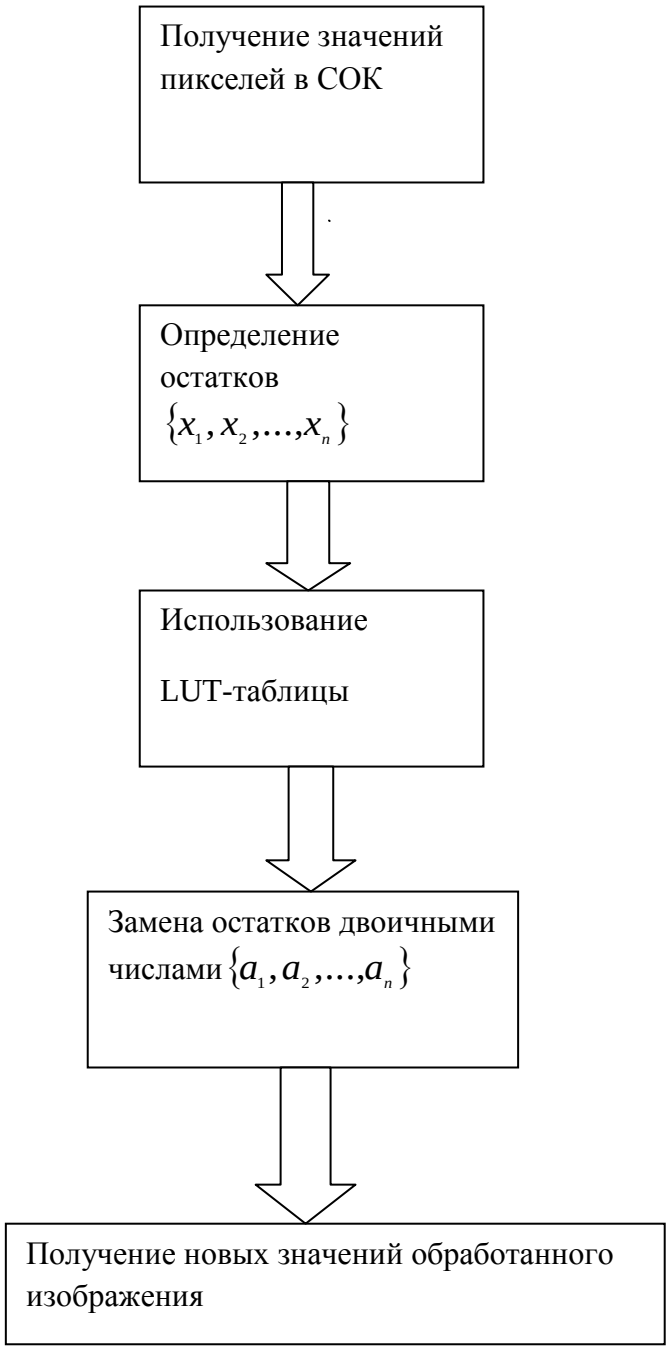


Рисунок 2.10. Схема обратного преобразователя СОК-ПСС согласно [121]

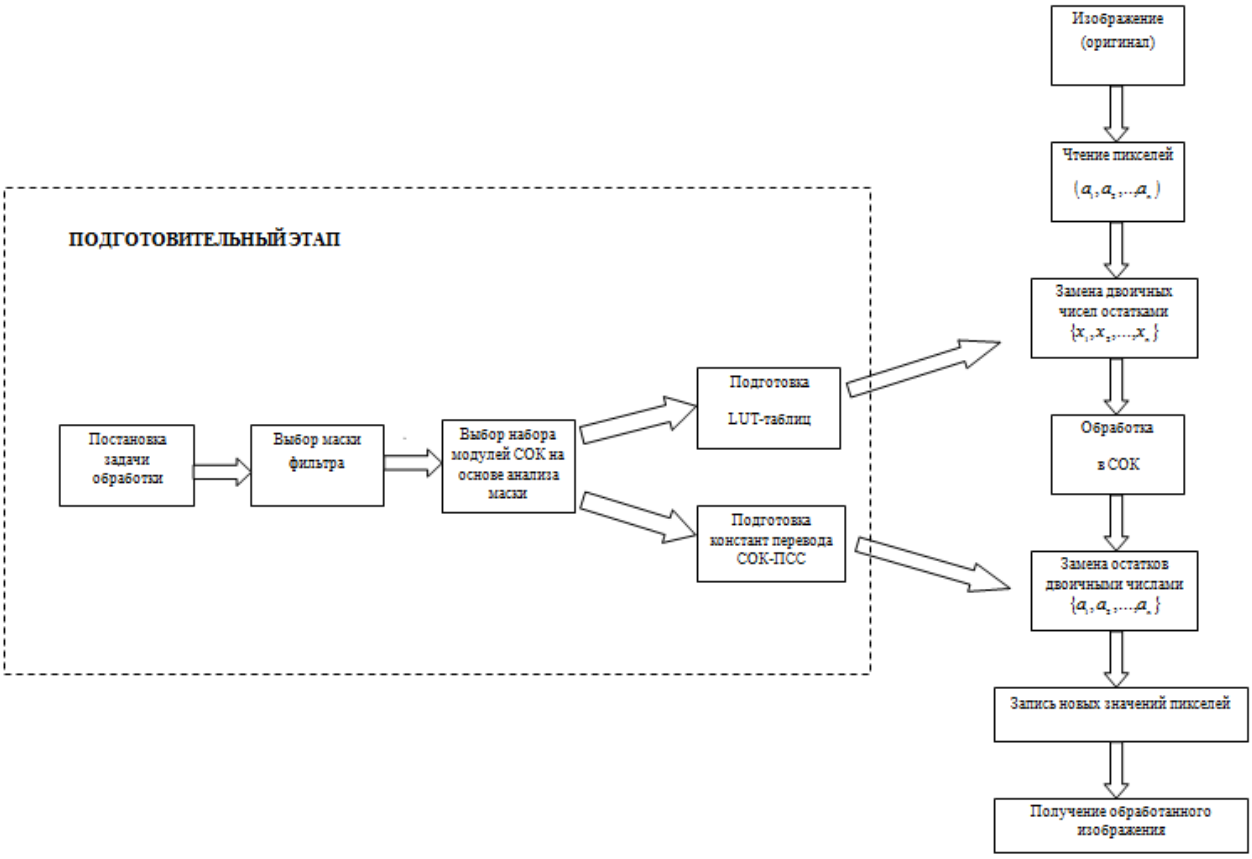


Рисунок 2.11. Предлагаемая схема процесса фильтрации изображения

2.4 Разработка численной реализации метода фильтрации изображения с вычислениями в СОК

Рассмотрим более подробно влияние цифровой фильтрации на изображение на примере известных фильтров Привита, Собеля, Гаусса, Лапласиана. Фильтрация будет проводиться в пространственной области с применением изображения в оттенках серого. Для представления одного изображения будем использовать двумерную матрицу размером $M \times N$. Значение каждого элемента из этой матрицы показывает степень яркости, при этом каждый элемент принимает 8 битное значение, которое может варьироваться в пределах $0-255$. В математическом пакете MATLAB такое представление изображения запишется следующим образом [11]:

$$F(x,y)=\begin{bmatrix} F(1,1) & F(1,2) & \cdots & F(1,N) \\ F(2,1) & F(2,2) & \cdots & F(2,N) \\ \vdots & \vdots & \vdots & \vdots \\ F(M,1) & F(M,2) & \cdots & F(M,N) \end{bmatrix} \quad (2.13)$$

Сочетание функции и маски изменений уровней серого, записанного в виде формулы (4.1) называется фильтром. Для того чтобы применить линейный фильтр к массиву изображения необходимо перемножить все коэффициенты изображения на соответствующие прилегающие элементы из фильтра (маски) и суммировать все полученные значения [2]. В случае выполнения фильтрации в пространственной области, маска накладывается на соответствующий участок изображения, двигается на нем, а в качестве вычисляемых значений используются соответствующие значения пикселей, которые совпали при накладывании маски [18]. Таким образом, после такой операции получается новое изображение в оттенках серого, вычисленное в соответствии с предложенной маской. Математическая интерпретация описанной операции соответствует формуле (2.14):

$$g(x,y)=\sum_{s=-a}^{a}\sum_{t=-b}^{b} w(s,t)f(x+s,y+t), \quad (2.14)$$

где a и b – положительные числа; $f(x,y)$ – функция для заданного фильтра; $w(s,t)$ – функция для значений выходного изображения.

Предположим, что коэффициенты маски размером 3×3 можно представить в виде:

$$\begin{bmatrix} w(-1,1) & w(-1,0) & w(-1,-1) \\ w(0,1) & w(0,0) & w(0,-1) \\ w(1,1) & w(1,0) & w(1,-1) \end{bmatrix}$$

Коэффициенты значений пикселей имеют следующие значения:

$$\begin{bmatrix} f(x-1,x+1) & f(x-1,y) & f(x-1,y-1) \\ f(x,y+1) & f(x,y) & f(x,y-1) \\ f(x+1,y+1) & f(x+1,y) & f(x+1,y-1) \end{bmatrix}$$

Перемножим две представленные матрицы коэффициентов между собой и проссумируем полученные значения [11,118]:

$$\sum_{s=-1}^{1}\sum_{t=-1}^{1} w(s,t) f(x+s, y+t) \tag{2.15}$$

Таким образом, процесс фильтрации можно поделить на 3 этапа:

1. Расположение заданной маски на фрагменте изображения.
2. Умножение коэффициентов фильтра на смежные коэффициенты изображения.
3. Нахождение общей суммы полученных значений.

Все множество фильтров, которые используются для получения улучшения качества цифрового изображения можно поделить на две категории: низкочастотные и высокочастотные фильтры. Обработка цифрового изображения низкочастотным фильтром, который проходит через полосу низкочастотных пикселей и изменяет высокочастотные пиксели, может привести к скрытым изображениям. Фильтр высоких частот, который проходит через высокочастотные пиксели и вносит изменения в полосе низких частот пикселей, приводит к появлению шума на краях изображения [118]. Наиболее известными фильтрами, применяемыми для решения обозначенных проблем, которые так же применяются для решения задач повышения или понижения резкости изображения, обнаружения границы, шумопонижения и т. д. являются следующие фильтры:

Фильтры Превитта для обнаружения границы:

$$P_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}, \quad P_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

Фильтры Собеля для обнаружения границы:

$$P_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}, \quad P_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Поскольку операция фильтрации изображения согласно формуле (2.14) требует большого количества выполнения арифметический операций сложения и умножения значений пикселей с соответствующей маской, то можно сделать вывод, что для операций, связанных с цифровой обработкой изображений необходимо повышение скорости выполнения этих операций. Кроме того, увеличение скорости работы системы приведет к понижению энергопотребления цифрового приложения. Подходящим решением проблемы является применение непозиционной системы счисления. Помимо свойства высокого параллелизма, которым обладает СОК, архитектура такой системы может использовать в своей работе комбинационные схемы, а также просмотровые таблицы (LUT-таблицы). Эти преимущества, заключающиеся в выработке достаточно высокой скорости работы системы и малой потребляемой мощности, подчеркивают преимущества применения СОК в приложениях цифровой обработке изображений [2,56].

В процессе фильтрации изображений будем использовать Китайскую Теорему об Остатках, а также Модифицированную Китайскую теорему об остатках. Применив КТО в совокупности с LUT-таблицами можно выполнять операции над числами с наименьшими потерями информации [69]. Применение модифицированной КТО поможет уменьшить динамический диапазон системы и как следствие повысить эффективность реализации.

Таким образом, применяя модифицированную КТО получим снижение динамического диапазона:

$$P = \prod_{i=1}^{p} p_i \tag{2.16}$$

Функция в этом случае примет вид:

$$X = x_1 + p_1 \left| \sum_{i=1}^{m} w_i x_i^{'} \right|_{p_2 \cdots p_m}, \text{ где } m > 1. \tag{2.17}$$

$$w_1 = \frac{N_1 \left| N_1^{-1} \right|_{p_1^{-1}}}{p_1}$$

$$w_i = N_i / p_1, \text{ для } i = 2,3,..,m,\ x_1^{'} = x_1 \text{ и}$$

$$x_1^i = \left| N_1^{-1} x_1 \right|_{p_1},\ i = 2,3,..,m.$$

В пункте 2.5 будет описан актуальный вопрос, возникающий при цифровой обработки изображений, а именно выбор корректного набора модулей. Поэтому, исходя из результатов полученных ранее, применим, предлагаемый нами в пункте 2.5 набор модулей и для решения данной задачи. Предлагаемый набор модулей имеет вид: $\{5,7,9,16\}$.

Промоделируем работу данного набора модулей на примере конкретного изображения. В качестве входного изображения будем рассматривать изображение «девушка» в оттенках серого. Входные значения пикселей в этом изображении представлены в виде:

$$\begin{bmatrix} 252 & 254 & 250 & 250 \\ 250 & 253 & 254 & 251 \\ 254 & 253 & 252 & 251 \\ 254 & 252 & 253 & 254 \end{bmatrix}$$

Маска значений для перемещения четырех средних значений пикселей изображения приведена ниже. Представленная маска фильтра известна как фильтр Лапласа для повышения резкости:

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Преобразуем все имеющиеся значения пикселей изображения и значения маски фильтра в значения, представленные в СОК. В качестве выбранного набора модулей будем использовать набор $\{5,7,9,16\}$. Динамический диапазон набора $P = 5 \cdot 7 \cdot 9 \cdot 16 = 5040$ подходит для цифровых изображений, представленных в оттенках серого. Тогда для выбранных значений пикселей получим:

$$\begin{bmatrix} \{2,0,0,12\} & \{4,2,2,14\} & \{0,5,7,10\} & \{0,5,7,10\} \\ \{0,5,7,10\} & \{3,1,1,13\} & \{4,2,2,14\} & \{1,6,8,11\} \\ \{4,2,2,14\} & \{3,1,1,13\} & \{2,0,0,12\} & \{1,6,8,11\} \\ \{4,2,2,14\} & \{2,0,0,12\} & \{3,1,1,13\} & \{4,2,2,14\} \end{bmatrix}$$

Значения маски фильтра перепишутся в виде:

$$\begin{bmatrix} \{4,6,8,15\} & \{4,6,8,15\} & \{4,6,8,15\} \\ \{4,6,8,15\} & \{4,2,0,9\} & \{4,6,8,15\} \\ \{4,6,8,15\} & \{4,6,8,15\} & \{4,6,8,15\} \end{bmatrix}.$$

Учитывая использование LUT-таблицы в обратном преобразовании при переходе от системы остаточных классов к позиционной системе счисления необходимо построить таблицу поиска для значений пикселей цифровых изображений (таблица 2.4).

Таблица 2.4. Значения пикселей изображения, представленные в СОК и ПСС

Значения пикселей в наборе модулей $\{5,7,9,16\}$	Значения пикселей в позиционной системе счисления
$\{0,5,7,10\}$	250
$\{1,6,8,11\}$	251
$\{2,0,0,12\}$	252
$\{3,1,1,13\}$	253
$\{4,2,2,14\}$	254

Рассмотрим более подробно процесс фильтрации значений изображения. В качестве обрабатываемого массива данных будем использовать массив того же размера 3×3, что и маска фильтра. При умножение значений изображений на соответствующие значения фильтра получим:

$$\begin{bmatrix} \{2,0,0,12\} & \{4,2,2,14\} & \{0,5,7,10\} \\ \{0,5,7,10\} & \{3,1,1,13\} & \{4,2,2,14\} \\ \{4,2,2,14\} & \{3,1,1,13\} & \{2,0,0,12\} \end{bmatrix} \times \begin{bmatrix} \{4,6,8,15\} & \{4,6,8,15\} & \{4,6,8,15\} \\ \{4,6,8,15\} & \{4,2,0,9\} & \{4,6,8,15\} \\ \{4,6,8,15\} & \{4,6,8,15\} & \{4,6,8,15\} \end{bmatrix} =$$

$$\begin{bmatrix} \{8,0,0,180\} & \{16,12,16,210\} & \{0,30,56,150\} \\ \{0,30,56,150\} & \{12,2,0,117\} & \{16,12,16,210\} \\ \{16,12,16,210\} & \{12,6,8,195\} & \{8,0,0,180\} \end{bmatrix}.$$

Далее сложим все соответственные коэффициенты друг с другом, после чего получим остаток каждого по модулю. Полученный результат $\{3,6,6,2\}$ соответствует значению первого пикселя изображения, представленного в СОК.

Проведем аналогичную операцию для оставшихся значений пикселей, после чего получим:

$$\begin{bmatrix} \{3,6,6,2\} & \{2,6,2,0\} \\ \{0,3,3,15\} & \{2,2,4,7\} \end{bmatrix}.$$

Для перевода найденных значений из представления в СОК обратно в ПСС применим КТО. Поскольку найденный динамический диапазон СОК равен $P = 5 \cdot 7 \cdot 9 \cdot 16 = 5040$, тогда по формуле $P_i = \frac{P}{p_i}$ найдем диапазоны для каждого из четырех модулей $\{5,7,9,16\}$ и остатки при делении:

$$P_1 = 1008, \qquad 1008 \bmod 5 = 3$$

$$P_2 = 720, \qquad 720 \bmod 7 = 6$$

$$P_3 = 560, \qquad 560 \bmod 9 = 2$$

$$P_4 = 315 \qquad 315 \bmod 16 = 11$$

Применяя обратную операцию, поучаем следующие значения:

$$3 \cdot x_1 = 1 \bmod 5 \quad \rightarrow \quad x_1 = 2$$

$$6 \cdot x_2 = 1 \bmod 7 \quad \rightarrow \quad x_2 = 6$$

$$2 \cdot x_3 = 1 \bmod 9 \quad \rightarrow \quad x_3 = 5$$

$$11 \cdot x_4 = 1 \bmod 16 \quad \rightarrow \quad x_4 = 3$$

Найдем массив B_i:

$$B_1 = P_1 \cdot x_1 = 1008 \cdot 2 = 2016;$$

$$B_2 = P_2 \cdot x_2 = 720 \cdot 6 = 4320;$$

$$B_3 = P_3 \cdot x_3 = 560 \cdot 5 = 2800;$$

$$B_4 = P_4 \cdot x_4 = 315 \cdot 3 = 945.$$

Тогда полученное значение пикселя после обратной операции преобразования будет записано как:

$$A = 3 \cdot 2016 + 6 \cdot 4320 + 6 \cdot 2800 + 2 \cdot 945 = 50658,$$

$$50658 \bmod 5040 = 258.$$

Проведя аналогичные операции для каждого из значений получим преобразованные значения пикселей из СОК в ПСС:

$$\begin{bmatrix} 258 & 272 \\ 255 & 247 \end{bmatrix}.$$

Все вычисления производились с использованием пакета прикладных программ MATLAB [95]. Результатами моделирования являются, полученные гистограммы для цифровых изображений для входного изображения и изображения, полученного после фильтрации (рисунки 2.12 и 2.13 соответственно). По горизонтальной оси каждого графика отложены значения уровней яркости r_k, а по вертикальной – значения гистограммы $h(r_k) = n_k$. Тем самым, эти графики выражают зависимости $h(r_k) = n_k$ от r_k или $p(r_k) = n_k / n$ от r_k (в случае, если значения гистограммы нормализованы) [11]. Из этих графиков видно, что ненулевые уровни гистограммы покрывают широкую часть диапазона яркостей, а также, что распределение значений пикселей не слишком отличается от равномерного, за исключением числа пиков, возвышающихся над остальными значениями. Интуитивно можно сделать вывод, что изображение, распределение значений элементов которого близко к равномерному и занимает весь диапазон возможных значений яркостей, будет выглядеть высококонтрастным и будет содержать большое количество полутонов [121].

Таким образом, основываясь только на информации, содержащейся в гистограмме исходного изображения, можно построить функцию преобразования, которая позволит автоматически добиваться такого эффекта.

Таким образом, предлагаемая система счисления хорошо подходит для приложений, использующих цифровую фильтрацию, поскольку, она эффективно выполняет операции сложения, умножения, широко использующихся в данном случае. Кроме того цифровая фильтрация на основе СОК вызывает повышенную скорость работы, снижает потребляемую мощность и уменьшает количество чипов в своей конструкции. Результаты моделирования в Matlab показали способность предлагаемой конструкции для фильтрации данных изображений.

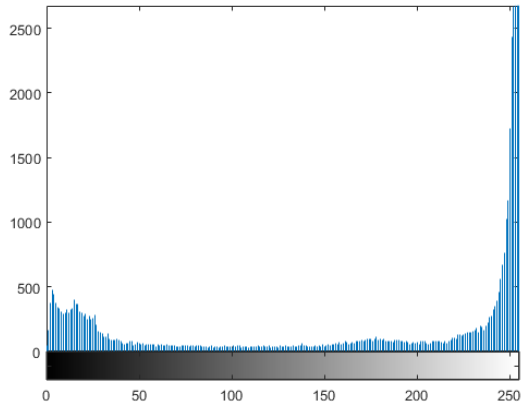
а)  б) 

Рисунок 2.12. а) Входное изображение «девушка» до обработки фильтром; б) гистограмма значений изображения.

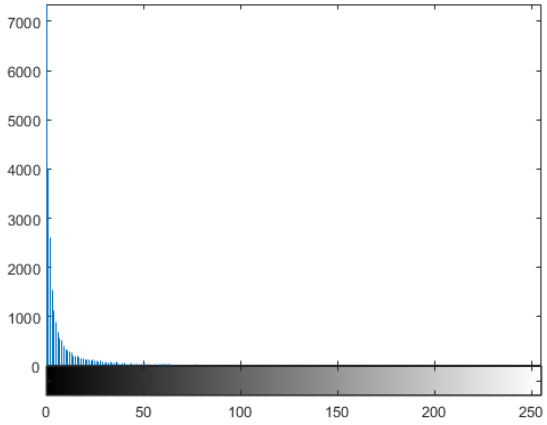
а)  б) 

Рисунок 2.13. а) Обработанное фильтром изображение «девушка»; б) гистограмма значений изображения.

2.5 Разработка метода повышения производительности цифровых фильтров на основе использования СОК

Как было указано в пункте 1.3.1, цифровые фильтры подразделяются на нерекурсивные – фильтры с конечной импульсной характеристикой (КИХ-фильтры) и рекурсивные – фильтры с бесконечной импульсной характеристикой (БИХ-фильтры) [1]. КИХ-фильтры используют меньше аппаратных затрат, что

делает их проектирование проще по сравнению с БИХ-фильтрами. На сегодняшний день цифровая фильтрация с использованием КИХ-фильтров применяется практически везде, где требуется высокоэффективная обработка сигналов, в частности в спектральном анализе, обработке изображений, обработке звука, обработке видео и во многих других приложениях [117].

Любой КИХ-фильтр, можно задать следующим выражением

$$y_n = \sum_{k=0}^{N} b_k x_{n-k}, \tag{2.18}$$

где x_n – входная последовательность сигнала, b_k – коэффициенты фильтра, N – порядок фильтра, y_n – последовательность сигнала, полученного на выходе фильтра [49].

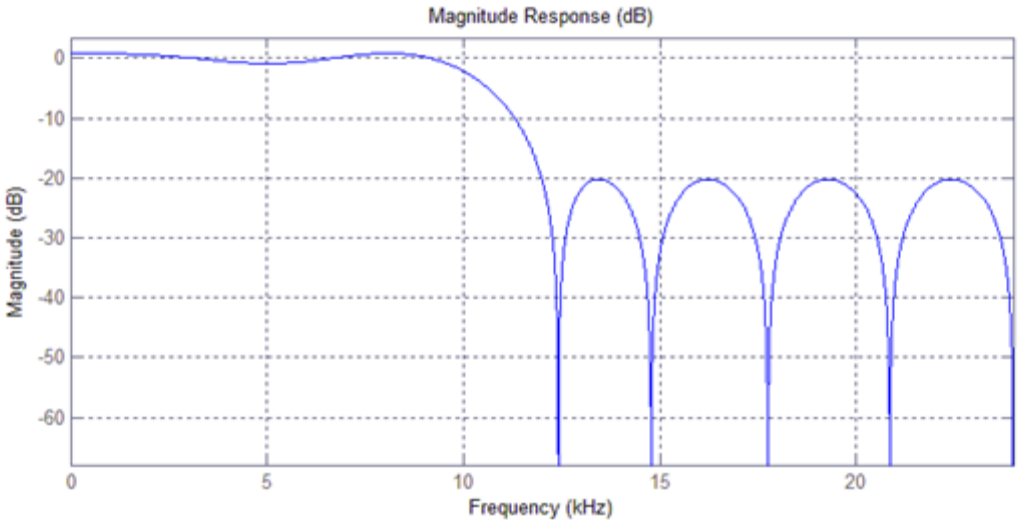


Рисунок 2.14. – АЧХ низкочастотного фильтра 15-го порядка.

Из формулы (6) видно, что для работы КИХ-фильтра используются только операции сложения и умножения, что позволяет при его реализации использовать положительные качества СОК.

Рассмотрим низкочастотный КИХ-фильтр 15-го порядка, АЧХ которого представлена на рисунке 2.14. Проектирование фильтра было осуществлено в MATLAB®, с использованием встроенного приложения Filter Design and Analysis. Фильтр, изображенный на рисунке 2.14 построен с использованием алгоритма

Паркса-МакКлеллана [95]. Коэффициенты фильтра в формате двойной точности и в целочисленном формате данных приведены в таблице 2.5.

Таблица 2.5. Коэффициенты КИХ-фильтра 15-го порядка

Коэффициенты фильтра	Числовой формат представления	
	Двойная точность (64 бита)	Целое число (12 бит)
$b_0 = b_{15}$	$-0{,}04842854521480$	-100
$b_1 = b_{14}$	$0{,}031641772569045$	64
$b_2 = b_{13}$	$0{,}066305238639482$	135
$b_3 = b_{12}$	$0{,}016109250214128$	32
$b_4 = b_{11}$	$-0{,}076170627278596$	-156
$b_5 = b_{10}$	$-0{,}041780626517982$	-86
$b_6 = b_9$	$0{,}183762552952241$	376
$b_7 = b_8$	$0{,}417189375297071$	854

Для получения целочисленных значений данного фильтра была использована команда

```
num2int(quantizer([12,11]),z)
```

Таким образом, полученные целочисленные значения коэффициентов фильтра являются 12-битными числами, из которых 11 бит – величина числа, 1 бит – знак. Эти целочисленные значения в дальнейшем могут быть представлены в СОК, путем простого вычисления остатков от деления на модули СОК.

Так как, исходные коэффициенты фильтра являются числами двойной точности длиной 64 бита, то переход от такого представления к 12-битному влечет за собой ошибку округления. На рисунке 2.15 показана АЧХ этой ошибки. Из рисунка видно, что максимум ошибки примерно на 25 Дб ниже, чем область запирания фильтра.

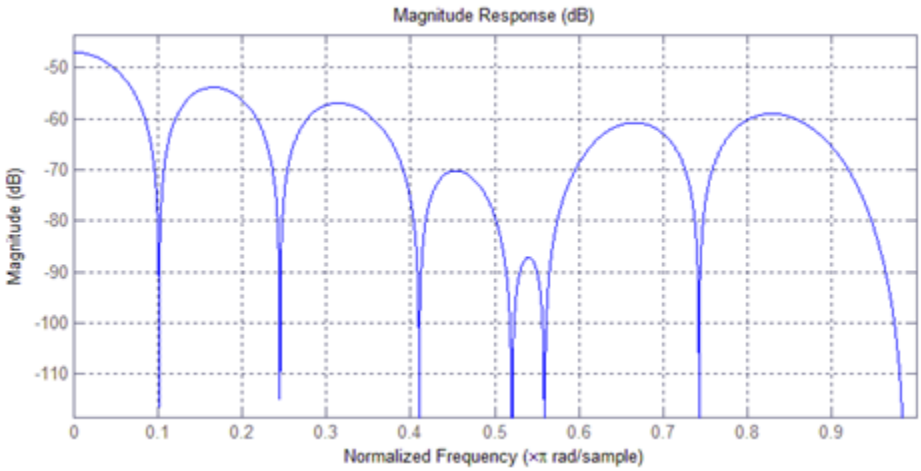


Рисунок 2.15. – АЧХ ошибки округления КИХ-фильтра.

Для равномерного распределения последствий из-за возникновения ошибок округления между фильтром и сигналом, рассмотрим случай, когда в качестве входной последовательности используется сигнал с такой же точностью, что и коэффициенты фильтра, т.е. 12 бит. Тогда диапазон работы КИХ-фильтра равен:

$$\max\left\{\left|y(n)\right|\right\} = 2^{12} \cdot \sum_{k=0}^{15} \left|b_k\right| = 4096 \cdot 3606 = 14770176 \tag{2.19}$$

Передаточная функция фильтра, описывающая связь между сигналами на входе и выходе фильтра, задается в виде

$$H(z) = b_0 + b_1 z^{-1} + \ldots + b_{15} z^{-15} = \sum_{i=0}^{15} b_i z^{-i}. \tag{2.20}$$

С учетом того, что коэффициенты построенного фильтра симметричны, выражение (2.20) можно переписать в следующем виде

$$H(z) = \sum_{i=0}^{8} b_k \left(z^k + z^{15-k}\right). \tag{2.21}$$

На рисунке 2.16 представлена схема фильтра, описанного формулой (2.21), где блок z^{-1} – это блок задержки. При реализации изображенной схемы фильтра в СОК достаточно заменить коэффициенты b_i $1 = 0,\ldots,15$ на их модулярное представление, а блоки сложения и умножения на блоки сложения по модулю и умножения по модулю, соответственно.

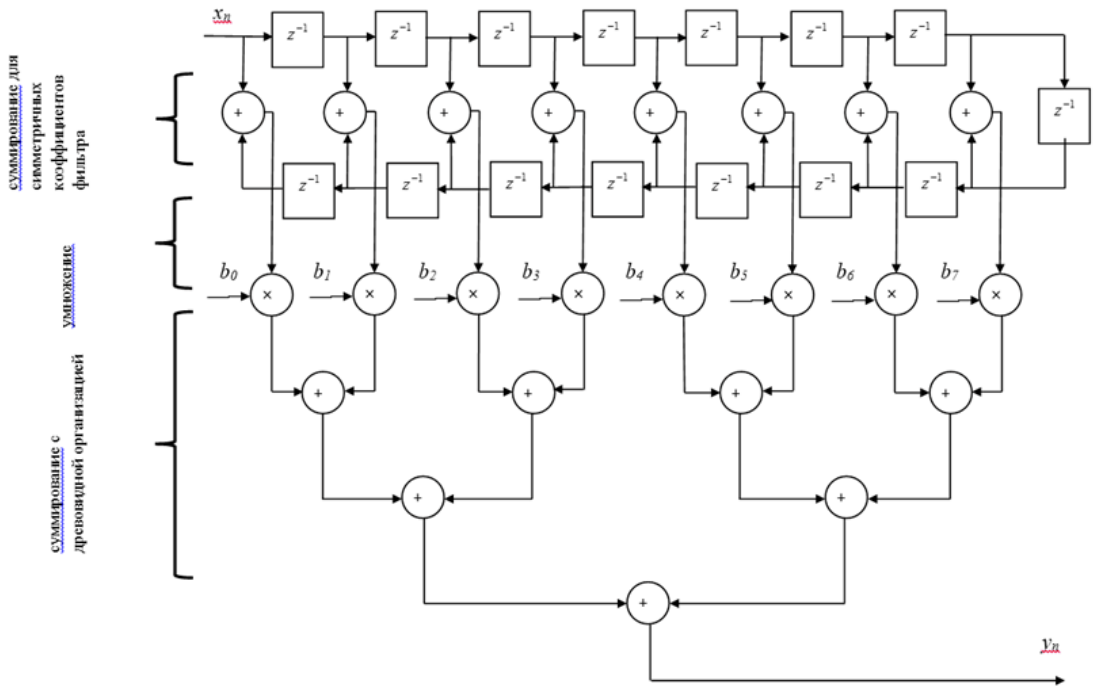


Рисунок 2.16. – Схема работы фильтра 15-го порядка

Как было сказано в пункте 1.4, выбор правильного набора модулей является еще одним важным вопросом для построения эффективной СОК с достаточным динамическим диапазоном [27,127].

Одним из самых известных наборов модулей является набор вида $\{2^n-1,2^n,2^n+1\}$ [71,114]. Этот набор известен как средство упрощения расчетов, которые необходимы для осуществления операции обратного преобразования. Тем не менее, арифметические схемы, которые используют модуль (2^n+1) имеют большую задержку среди всех трех каналов. Таким образом, для того, чтобы упростить сложность, вызванную данным модулем были предложены новые наборы модулей: $\{2^{n-1}-1,2^n-1,2^n\}$ [11] и $\{2^n-1,2^n,2^{n+1}-1\}$ [12]. Эти три набора имеют 3n-битный динамический диапазон. Так как для многих приложений ЦОС необходим больший динамический диапазон, следовательно, были предложены новые наборы модулей $\{2^n-1,2^n,2^{2n+1}-1\}$ [13] и $\{2^n-1,2^n+1,2^{2n}+1\}$ [127], которые обеспечивают 4n-битный диапазон, а также $\{2^n-1,2^{2n},2^{2n}+1\}$ [90], который дает 5n-битный динамический диапазон. Несмотря на то, что динамический диапазон стал больше, увеличилась задержка работы арифметических устройств, основанных на

этих наборах, в связи с использованием модулей с большими величинами. Чтобы устранить этот недостаток, и сохранить большой динамический диапазон были предложены наборы имеющие от четырех до пяти модулей, например, $\{2^n-1,2^n,2^n+1,2^{n+1}-1\}$ [103], $\{2^n-1,2^n,2^n+1,2^{n+1}+1\}$ [103], $\{2^n-1,2^n,2^n+1,2^{2n}+1\}$ [76], $\{2^n-1,2^n,2^n+1,2^{2n+1}-1\}$ [105], $\left\{2^n-1,2^n,2^n+1,2^n-2^{\frac{n+1}{2}}+1,2^n+2^{\frac{n+1}{2}}+1\right\}$ [92], $\{2^n-1,2^n+1,2^{2n}-1,2^{2n+1}-3\}$ [132], $\{2^n-1,2^{2n},2^n+1,2^{2n}+1\}$ [105], $\{2^n-1,2^n,2^n+1,2^{n-1}-1,2^{n+1}+1\}$ [77] и $\left\{2^n,2^{\frac{n}{2}}-1,2^{\frac{n}{2}}+1,2^n+1,2^{2n-1}-1\right\}$ [104].

Каждый из этих наборов имеет свои преимущества и недостатки. Некоторые из них предлагают большой динамический диапазон, в то время как другие имеют большую степень параллелизма. Некоторые из них могут привести к более эффективному обратному преобразованию, в то время как другие к получению более эффективного остатка арифметических устройств [73]. Как будет видно далее, выбор наиболее эффективного набора модулей играет существенную роль в скорости работы КИХ-фильтра.

Для оценки времени работы КИХ-фильтров в СОК проводился подсчет тактов синхронизации вычислительной системы, необходимых для выполнения полного цикла работы фильтра при обработке одного вновь поступившего отсчета входной последовательности. В КИХ-фильтрах в СОК используются только операции сложения и умножения, которые в свою очередь тем или иным способом (в зависимости от конкретного вида модулей СОК) сводится к комбинации базовых логических элементов. В соответствии с методом, предложенным в работе [129], использовались следующие значения количества тактов синхронизации вычислительной системы для базовых элементов.

1. Время работы двухвходовых элементов «И», «ИЛИ», «И-НЕ», «ИЛИ-НЕ» принимается равным $T_1=1$.

2. Время работы двухвходовых элементов «ИСКЛЮЧАЮЩЕЕ ИЛИ», а также мультиплексора 2:1 принимается равным $T_2=2$.

3. Время работы двоичного полусумматора принимается равным $T_3 = 2$.

4. Время работы полного двоичного сумматора принимается равным $T_4 = 4$.

Таблица 2.6. Время выполнения операций модулярного сложения и умножения для разных наборов модулей СОК

Набор модулей	Задержка, такты	
	Модулярное сложение	Модулярное умножение
$\{2^n-1, 2^n, 2^n+1\}$	$8n+11$	$16n+12$
$\{2^{n-1}-1, 2^n-1, 2^n\}$	$8n$	$16n-7$
$\{2^n-1, 2^n, 2^{n+1}-1\}$	$8n+8$	$16n+9$
$\{2^n-1, 2^n, 2^{2n+1}-1\}$	$16n+8$	$32n+9$
$\{2^n-1, 2^n+1, 2^{2n}+1\}$	$16n+11$	$32n+12$
$\{2^n-1, 2^n, 2^n+1, 2^{n+1}-1\}$	$8n+11$	$16n+12$
$\{2^n-1, 2^n, 2^n+1, 2^{n+1}+1\}$	$8n+19$	$16n+28$
$\{2^n-1, 2^{2n}-1, 2^{2n}+1\}$	$16n+11$	$32n+12$
$\{2^n-1, 2^n, 2^n+1, 2^{2n}+1\}$	$16n+11$	$32n+12$
$\{2^n-1, 2^n, 2^n+1, 2^{2n+1}-1\}$	$16n+8$	$32n+9$
$\left\{2^n-1, 2^n, 2^n+1, 2^n-2^{\frac{n+1}{2}}+1, 2^n+2^{\frac{n+1}{2}}+1\right\}$	$8n+11$	$24n+13$
$\{2^n-1, 2^n, 2^n+1, 2^{n-1}-1, 2^{n+1}+1\}$	$8n+19$	$16n+28$
$\left\{2^n, 2^{\frac{n}{2}}-1, 2^{\frac{n}{2}}+1, 2^n+1, 2^{2n-1}-1\right\}$	$16n-8$	$32n-23$
$\{2^n-1, 2^n+1, 2^{2n}-1, 2^{2n+1}-3\}$	$16n+11$	$48n+13$
$\{2^n-1, 2^{2n}, 2^n+1, 2^{2n}+1\}$	$16n+11$	$28n+12$

Как уже отмечалось ранее, выбор набора модулей играет важную роль в улучшении производительности КИХ-фильтров, построенных в СОК. Проведем анализ производительности КИХ-фильтра с использованием наиболее известных из них. В таблице 2.6 приведено время задержки при модулярном сложении и умножении, для каждого рассмотренного набора модулей.

Для каждого из приведенных в таблице 2.6 наборов модулей было подобрано наименьшее n, обеспечивающее необходимый диапазон, указанный в (7). Т.е. все наборы модулей должны быть взаимно простыми и удовлетворять

условию $P \geq 14770176$. В таблице 2.7 приведены минимальные значения n, удовлетворяющие этому условию.

Таблица 2.7. Время выполнения модульных операций для фильтрации в СОК с различными наборами модулей

№	Набор модулей	n	Десятичная запись модулей	Задержка, такты		Итого
				+	×	
1	$\{2^n-1, 2^n, 2^n+1\}$	8	{255,256,257}	75	140	440
2	$\{2^{n-1}-1, 2^n-1, 2^n\}$	9	{255,511,512}	72	137	425
3	$\{2^n-1, 2^n, 2^{n+1}-1\}$	7	{127,128,255}	64	121	377
4	$\{2^n-1, 2^n, 2^{2n+1}-1\}$	6	{63,64,8191}	104	201	617
5	$\{2^n-1, 2^n+1, 2^{2n}+1\}$	6	{63,65,4097}	107	204	632
6	$\{2^n-1, 2^n, 2^n+1, 2^{n+1}-1\}$	6	{63,64,65,127}	59	108	344
7	$\{2^n-1, 2^n, 2^n+1, 2^{n+1}+1\}$	7	{127,128,129,257}	75	140	440
8	$\{2^n-1, 2^{2n}-1, 2^{2n}+1\}$	5	{32,1023,1025}	91	187	551
9	$\{2^n-1, 2^n, 2^n+1, 2^{2n}+1\}$	5	{31,32,33,1025}	91	287	651
10	$\{2^n-1, 2^n, 2^n+1, 2^{2n+1}-1\}$	5	{31,32,33,2047}	88	169	521
11	$\left\{2^n-1, 2^n, 2^n+1, 2^n-2^{\frac{n+1}{2}}+1, 2^n+2^{\frac{n+1}{2}}+1\right\}$	5	{31,32,33,25,41}	51	133	337
12	$\left\{2^n, 2^{\frac{n}{2}}-1, 2^{\frac{n}{2}}+1, 2^n+1, 2^{2n-1}-1\right\}$	6	{64,7,9,65,2047}	88	169	521
13	$\{2^n-1, 2^n+1, 2^{2n}-1, 2^{2n+1}-3\}$	4	{15,17,254,509}	75	205	505
14	$\{2^n-1, 2^{2n}, 2^n+1, 2^{2n}+1\}$	4	{15,256,17,257}	75	124	424

В наборе модулей $\{2^n-1, 2^n, 2^n+1, 2^{n-1}-1, 2^{n+1}+1\}$ (таблица 2.6) при наименьшем необходимом $n=5$ не выполняется требование взаимной простоты. Поэтому этот набор был исключен из исследования. При реализации фильтра 15-го порядка логическая глубина выполнения операций равна 5, при этом используется одно параллельное умножение и 4 параллельных сложения (рисунок 2.16). Задержка для модулярного сложения и умножения получена с учетом данных из таблицы 2.6. В последней колонке таблицы 2.7 приведен суммарный результат количества необходимых тактов на выполнение всех модульных

операций при фильтрации в СОК. Анализ полученных результатов позволяет сделать вывод, что наиболее эффективными наборами модулей для ЦОС являются $\left\{2^n-1,2^n,2^n+1,2^n-2^{\frac{n+1}{2}}+1,2^n+2^{\frac{n+1}{2}}+1\right\}$ и $\left\{2^n-1,2^n,2^n+1,2^{n+1}-1\right\}$. Наименее эффективными наборами модулей, т.е. такими, которые будут приносить наибольшую задержку при выполнении модульных операций, являются наборы $\left\{2^n-1,2^n,2^n+1,2^{2n}+1\right\}$ и $\left\{2^n-1,2^n+1,2^{2n}+1\right\}$.

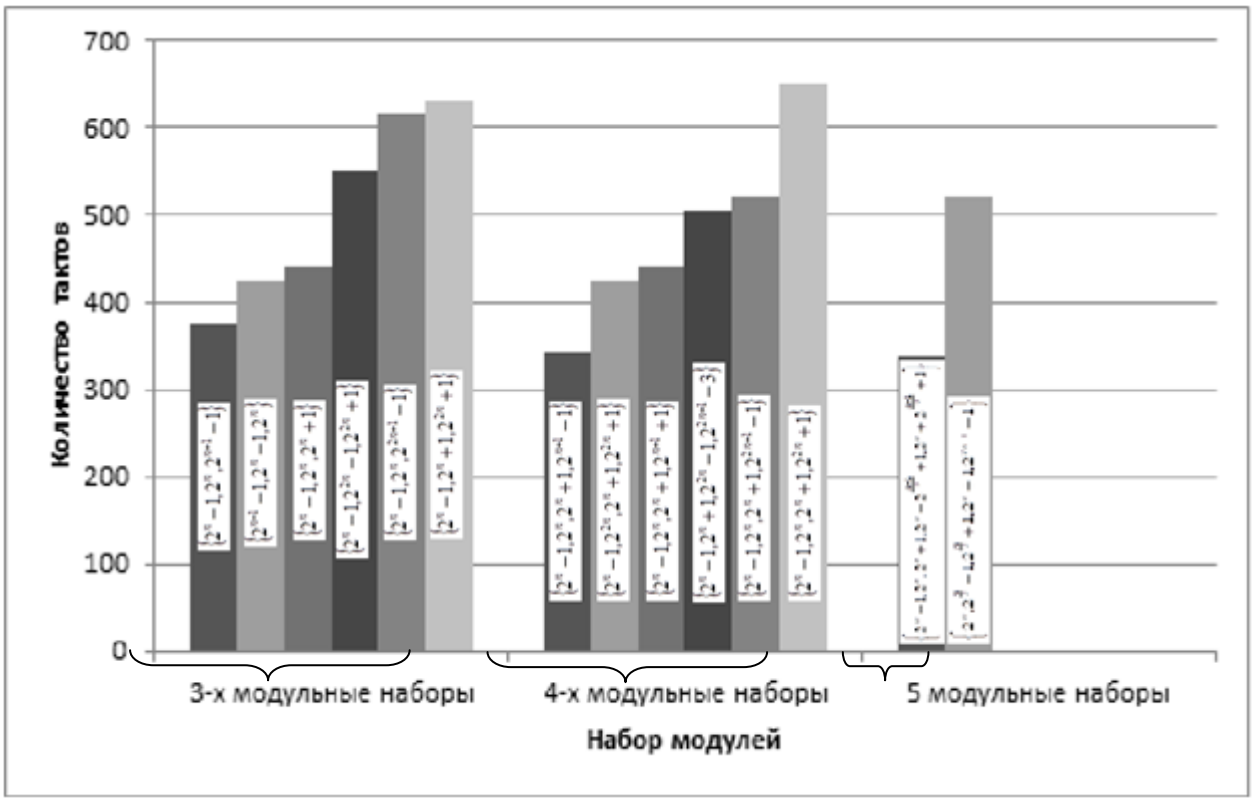


Рисунок 2.17 – Модульная задержка наборов модулей

На рисунке 2.17 представлено время работы КИХ-фильтра для рассмотренных наборов модулей. Среди всех 3-х модульных наборов наилучший результат дает набор $\left\{2^n-1,2^n,2^{n+1}-1\right\}$, его задержка составляет 377 тактов, наихудшим является набор $\left\{2^n-1,2^n+1,2^{2n}+1\right\}$ с задержкой в 632 такта. Наиболее эффективным среди 4-х модульных наборов является набор $\left\{2^n-1,2^n,2^n+1,2^{n+1}-1\right\}$ с задержкой равной 344 такта, а наименее эффективным набор $\left\{2^n-1,2^n,2^n+1,2^{2n}+1\right\}$ с задержкой в 651 такт. Для 5 модульных наборов наименьшую задержку в 337

тактов дает набор $\left\{2^n-1,2^n,2^n+1,2^n-2^{\frac{n+1}{2}}+1,2^n+2^{\frac{n+1}{2}}+1\right\}$, наибольшую в 521 такт набор $\left\{2^n-1,2^n,2^n+1,2^{2n+1}-1\right\}$.

Таблица 2.8. Сравнение времени фильтрации в СОК для разных наборов модулей

Набор модулей	Итоговая задержка, %
$\left\{2^n-1,2^n,2^n+1,2^n-2^{\frac{n+1}{2}}+1,2^n+2^{\frac{n+1}{2}}+1\right\}$	100
$\left\{2^n-1,2^n,2^n+1,2^{n+1}-1\right\}$	102
$\left\{2^n-1,2^n,2^{n+1}-1\right\}$	112
$\left\{2^{n-1}-1,2^n-1,2^n\right\}$	126
$\left\{2^n-1,2^{2n},2^n+1,2^{2n}+1\right\}$	126
$\left\{2^n-1,2^n,2^n+1\right\}$	131
$\left\{2^n-1,2^n,2^n+1,2^{n+1}+1\right\}$	131
$\left\{2^n-1,2^n+1,2^{2n}-1,2^{2n+1}-3\right\}$	150
$\left\{2^n-1,2^n,2^n+1,2^{2n+1}-1\right\}$	155
$\left\{2^n,2^{\frac{n}{2}}-1,2^{\frac{n}{2}}+1,2^n+1,2^{2n-1}-1\right\}$	155
$\left\{2^n-1,2^{2n}-1,2^{2n}+1\right\}$	163
$\left\{2^n-1,2^n,2^{2n+1}-1\right\}$	183
$\left\{2^n-1,2^n+1,2^{2n}+1\right\}$	188
$\left\{2^n-1,2^n,2^n+1,2^{2n}+1\right\}$	193

Примем время работы самого быстрого набора модулей $\left\{2^n-1,2^n,2^n+1,2^n-2^{\frac{n+1}{2}}+1,2^n+2^{\frac{n+1}{2}}+1\right\}$ равным 100%. В таблице 2.8 показано относительное время работы разных наборов модулей СОК, в сравнении с этим набором модулей. Самый эффективный набор модулей $\left\{2^n-1,2^n,2^n+1,2^n-2^{\frac{n+1}{2}}+1,2^n+2^{\frac{n+1}{2}}+1\right\}$ на 93% работает быстрее, чем самый

медленный набор $\{2^n-1,2^n,2^n+1,2^{2n}+1\}$, что подтверждает важность задачи выбора наиболее эффективного набора модулей СОК для ЦОС.

Однако, набор вида $\{2^n-1,2^n,2^n+1,2^{2n}+1\}$ содержит модули типа 2^n+1. Аппаратная реализация модулей такого типа приводит к большим трудностям при отслеживании кодовой комбинации, соответствующей числу 0 [110]. Во избежание трудностей такого рода в дальнейшем мы будем использовать СОК с модулями вида 2^n, 2^n-1 при аппаратной реализации разработанных методов и алгоритмов.

2.6 Разработка алгоритма кратномасштабного анализа сигналов в СОК

Как говорилось в главе 1, пункте 1.2, одними из наиболее распространенных методов обработки изображений являются методы на основе преобразование Фурье и вейвлет-преобразование. Рассмотрим работу дискретного вейвлет преобразования на примере цифровой обработки изображения в оттенках серого.

Как было указано в пункте 2.3, при цифровой обработке изображений изображение обычно представляется в виде прямоугольного массива целых чисел (пикселей). При этом яркость или темнота уровней серого цвета определяется степенями 2. Таким образом, чем больше число, представляющее код пикселя, тем ярче изображение в этой точке. Величина пикселей изображения кодируется 8 -битными числами, находящимися в диапазоне $[0,255]$, при этом 0 представляет черный цвет, 255 белый [91,118].

Для обработки изображений будем использовать вейвлетные фильтры. В этом случае, для обработки изображения требуется двукратное применение фильтров, так как изображение представляется в виде двумерного массива данных. Таким образом, обработка изображения происходит в два этапа. Первый этап состоит в обработке строк изображения, второй в обработке его столбцов. Обратный этап восстановления изображения также происходит в два этапа и

реализуется аналогичным образом [34,89]. На рисунке 2.18 схематично представлены этапы обработки изображений.

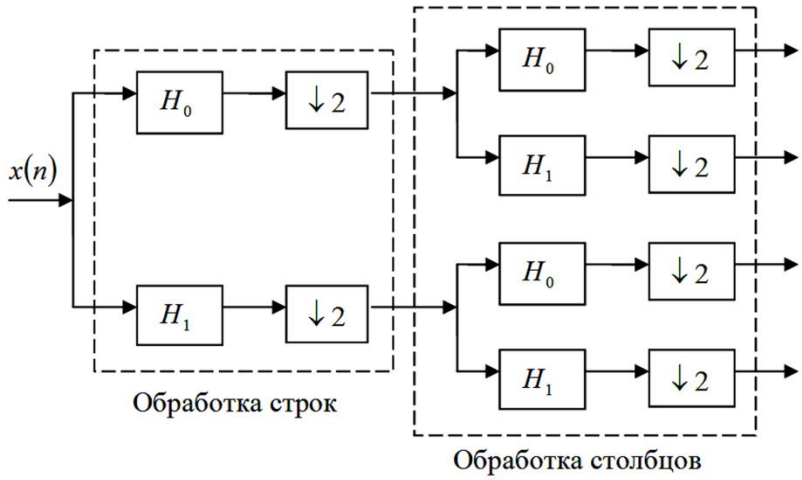
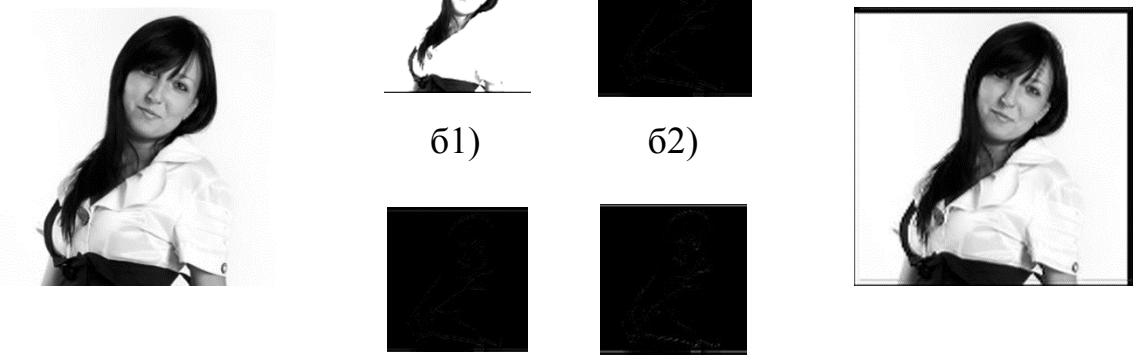


Рисунок 2.18. Структура набора фильтров обработки изображений.

Покажем работу фильтра Добеши-4 на примере. В качестве исходного изображения используем изображение «девушка», после обработки его вейвлет фильтрами получили 4 изображения, одно из которых является уменьшенной копией оригинала (рисунок 2.19). На этапе восстановления получаем изображение, совпадающее с изображением, не подвергшемся фильтрации.



а) б1) б2) б3) б4) в)

Рисунок 2.19. Результат обработки изображения «девушка» в оттенках серого а) исходное изображение; б1)-б4) результаты разложения исходного изображения; в) результат восстановления изображения

Исходные использующие коэффициенты фильтра являются числами двойной точности длиной 64 бита. Разложение и восстановления изображения с использованием фильтра Db-4 проводились в программной среде MATLAB® [48,95].

Одним из наиболее используемых и часто применяемых на практике при цифровой обработке данных является 8 битное представление информации. При 8 битном представлении данных со знаком, под величину числа отводится 6 бит, а знак числа определяется 2 старшими битами (00 – положительное, 01 и 10 – недопустимые комбинации, используемые для обнаружения ошибок, 11 – отрицательное).

Приведем пример записи числа -47 в 8-битном представлении, используя прямой, обратный и дополнительный коды записи информации. При переводе числа из десятичной в двоичную систему счисления, мы получим следующее представлении данного числа, что будет соответствовать прямому коду представления (рисунок 2.20):

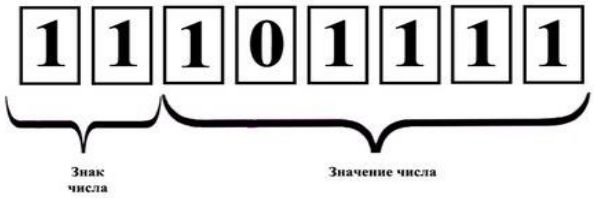


Рисунок 2.20. Схематичное представление числа -47 в записи прямого кода

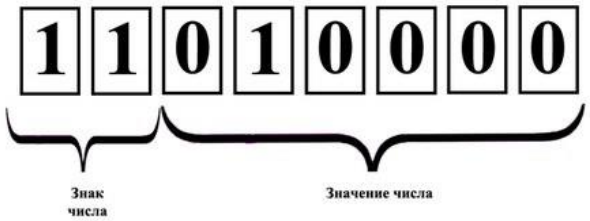


Рисунок 2.21. Схематичное представление числа -47 в записи обратного кода

Соответствующий данному числу обратный код записи будет иметь вид как на рисунке 2.21.

Дополнительный код представления числа образован путем прибавления к младшему биту в записи обратного кода 1, таким образом, получаем (рисунок 2.22):

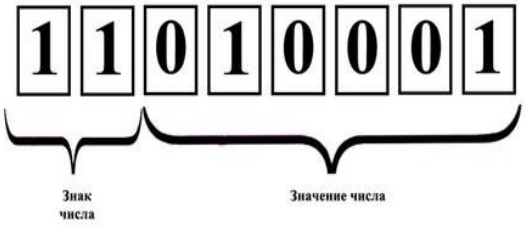


Рисунок 2.22. Схематичное представление числа -47 в записи дополнительного кода

На практике применяются обратный и дополнительный коды, что связано с сокращением числа, выполняемых операций и с количеством затраченного энергопотребления. В случае применения обратного кода имеется массив данных большей размерности и при этом большее количество операций. В дополнительном коде имеем меньший массив данных и меньшее число операций за счет неиспользования циклического переноса из старшего в младший разряд, что соответственно приводит к снижению затраченного энергопотребления.

Наиболее существенными характеристиками цифрового устройства при построении цифровых фильтров являются энергопотребление и быстродействие [22,64,101]. Динамическое потребление мощности микросхемой определяется по формуле:

$$W = n \cdot C \cdot V^2 \cdot f \qquad (2.22)$$

где n — число переключаемых узлов; C — емкость; V — разность уровней напряжения; f — частота. В формуле (1) число переключаемых узлов зависит, в том числе и от разрядности чисел, используемых во всех операциях.

Таким образом, из формулы (2.22) можно сделать вывод о прямо пропорциональном влияние количества разрядности данных на результирующую, потребляемой мощности приложением ЦОИ.

На быстродействие интегральной схемы оказывают влияние несколько факторов, которые входят в формулы (2.23) и (2.24) [74]:

$$t_{read/write} = \frac{N_{elements} \cdot N_{bytes/elements}}{a_{read/write} \cdot Throughtput_{ideal}}, \quad (2.23)$$

где $N_{elements}$ - количество передаваемых элементов, $N_{bytes/elements}$ - размер элемента данных, $Throughtput_{ideal}$ - идеальная пропускная способность при чтении/записи, $a_{read/write}$ - коэффициенты соотношения реальной и идеальной пропускной способности.

$$t_{comp} = \frac{N_{elements} \cdot N_{operations/element}}{f_{clock} \cdot Troughtput_{process}}, \quad (2.24)$$

где $N_{elements}$ - количество обрабатываемых элементов, $N_{operations/element}$ - количество операций над элементом данных, необходимое для достижения конечного результата, f_{clock} - тактовая частота, $Troughtput_{process}$ - количество операций за такт.

В формулах (2.23) и (2.24) значение $N_{elements}$ указывает на размер входных данных для формулы (2.23) и обрабатываемых данных для формулы (2.24). Одним из путей увеличения быстродействия устройства и снижения энергопотребления является уменьшение разрядности обрабатываемых коэффициентов фильтра. Далее будет показано, как влияет снижение разрядности на качество вейвлет-обработки изображения в оттенках серого.

Для моделирования обработки изображения использовались три изображения: «девушка», «дом», «тюльпаны». Для коэффициентов фильтра использовалась разрядность от 4 до 32 бит.

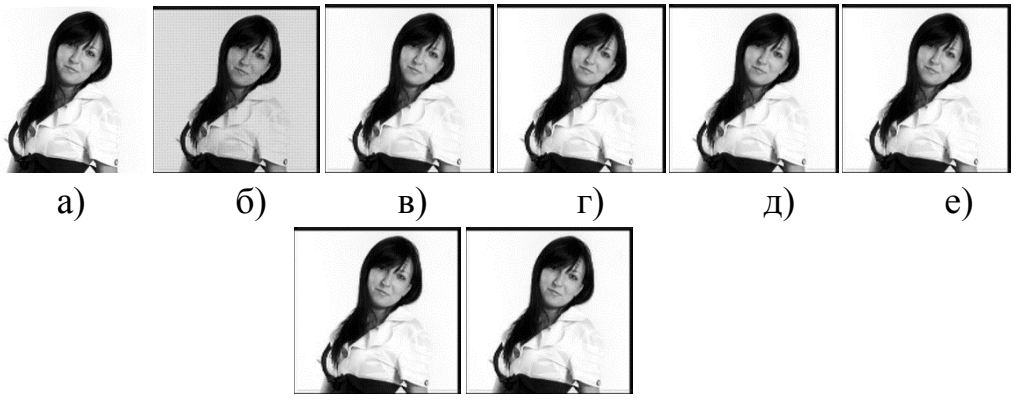
Из рисунков 2.23-2.25 видно, что с повышением разрядности коэффициентов фильтра увеличивается качество фильтруемого изображения.

Для количественной оценки качества полученных результативных изображений используем числовые характеристики PSNR и SSIM. Более подробно об этих характеристиках сказано в пункте 2.2. В таблице 1 представлены результаты оценки качества обработанных изображений с

использованием различной разрядности коэффициентов фильтра. Для рассматриваемых нами случаев фильтрации изображений с разной разрядностью коэффициентов фильтра величины PSNR [94] и SSIM [125] вычислялись между изображением, полученным при использовании традиционной двоичной системы счисления и изображением, полученным с использованием соответствующей разрядности коэффициентов: $8, 12, 16, 20, 24, 28, 32$.

В таблице 2.9 представлены результаты оценки качества обработанных изображений с использованием различной разрядности коэффициентов фильтра.

Из таблицы 2.9 видно, что уменьшение разрядности коэффициентов фильтра ведет к снижению качества обработанных изображений в оттенках серого. Таким образом, чем больше разрядность представляемых коэффициентов фильтра (в нашем случае изображения), тем лучше результат обработки изображения. Анализируя полученные результаты моделирования можно сделать вывод о том, что использование 12 битной разрядности коэффициентов фильтра дает приемлемое качество изображения, однако визуально неотличимое от оригинала. Для абсолютно точных расчетов, необходимо использовать 28 и более бит разрядности коэффициентов фильтра.

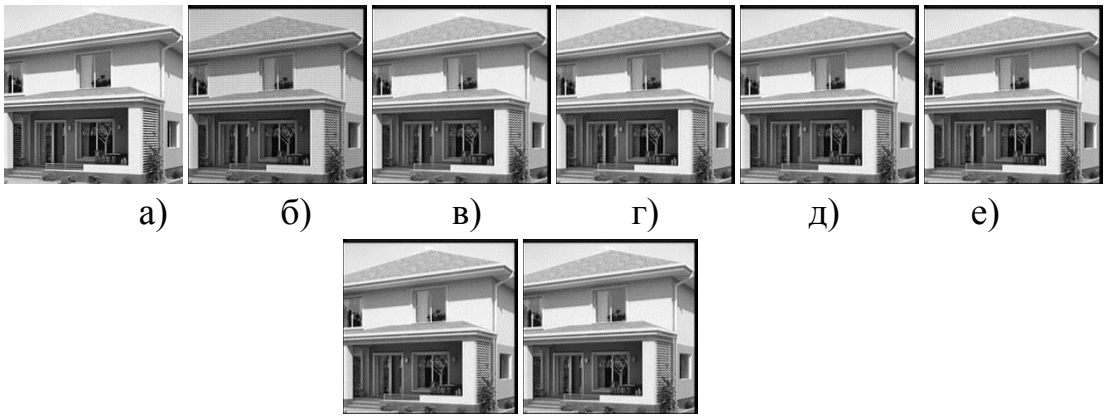


а) б) в) г) д) е)

ж) з)

Рисунок 2.23. Результаты моделирования изображения «девушка» с пониженной разрядностью коэффициентов фильтра.

а) исходное изображение; б) 8 бит; в) 12 бит; г) 16 бит; д) 20 бит; е) 24 бит; ж) 28 бит з) 32 бит

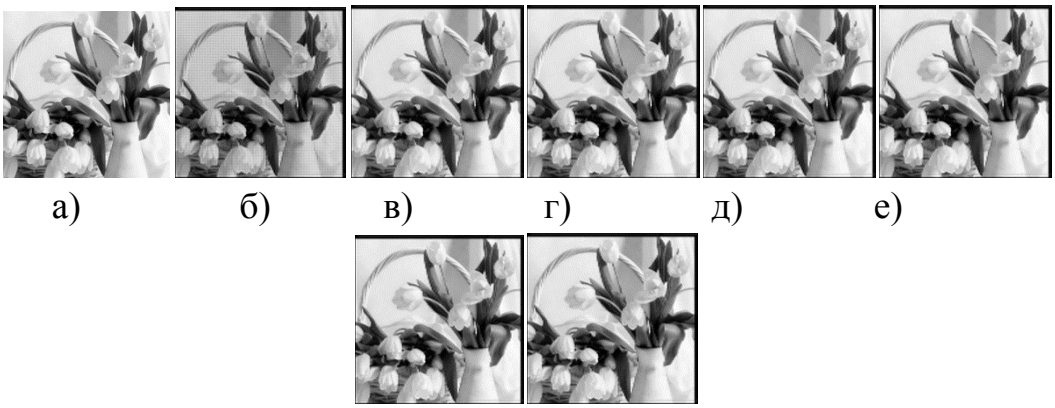


а) б) в) г) д) е)

ж) з)

Рисунок 2.24. Результаты моделирования изображения «дом» с пониженной разрядностью коэффициентов фильтра.

а) исходное изображение; б) 8 бит; в) 12 бит; г) 16 бит; д) 20 бит; е) 24 бит; ж) 28 бит з) 32 бит



а) б) в) г) д) е)

ж) з)

Рисунок 2.25. Результаты моделирования изображения «цветы» с пониженной разрядностью коэффициентов фильтра.

а) исходное изображение; б) 8 бит; в) 12 бит; г) 16 бит; д) 20 бит; е) 24 бит; ж) 28 бит з) 32 бит

Рисунок 2.26 показывает, что при увеличении разрядности коэффициентов фильтра показатель PSNR также увеличивается, что говорит об улучшении качества полученного при обработке изображения.

Таблица 2.9. Результаты моделирования обработки изображений с различной разрядностью коэффициентов фильтра.

Разрядность	Характеристика	Изображение		
		девушка	дом	цветы
8	PSNR, Дб	18,15	20,69	19,73
	SSIM	0,5398	0,7940	0,7117
12	PSNR, Дб	42,13	44,51	43,63
	SSIM	0,9903	0,9973	0,9961
16	PSNR, Дб	60,29	58,95	58,55
	SSIM	0,9995	0,9997	0,9995
20	PSNR, Дб	71,86	69,81	69,49
	SSIM	1,0000	1,0000	1,0000
24	PSNR, Дб	83,07	83,49	82,18
	SSIM	1,0000	1,0000	1,0000
28	PSNR, Дб	∞	96,49	96,49
	SSIM	1	1,0000	1,0000
32	PSNR, Дб	∞	∞	∞
	SSIM	1	1	1

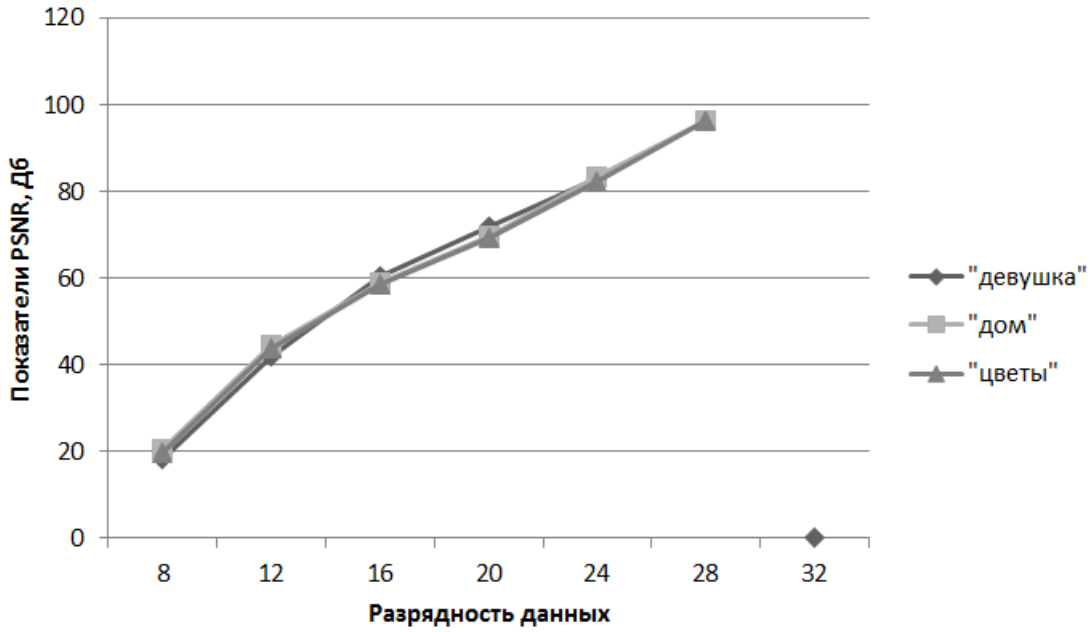


Рисунок 2.26. – Результаты PSNR изображений с уменьшением разрядности коэффициентов фильтра.

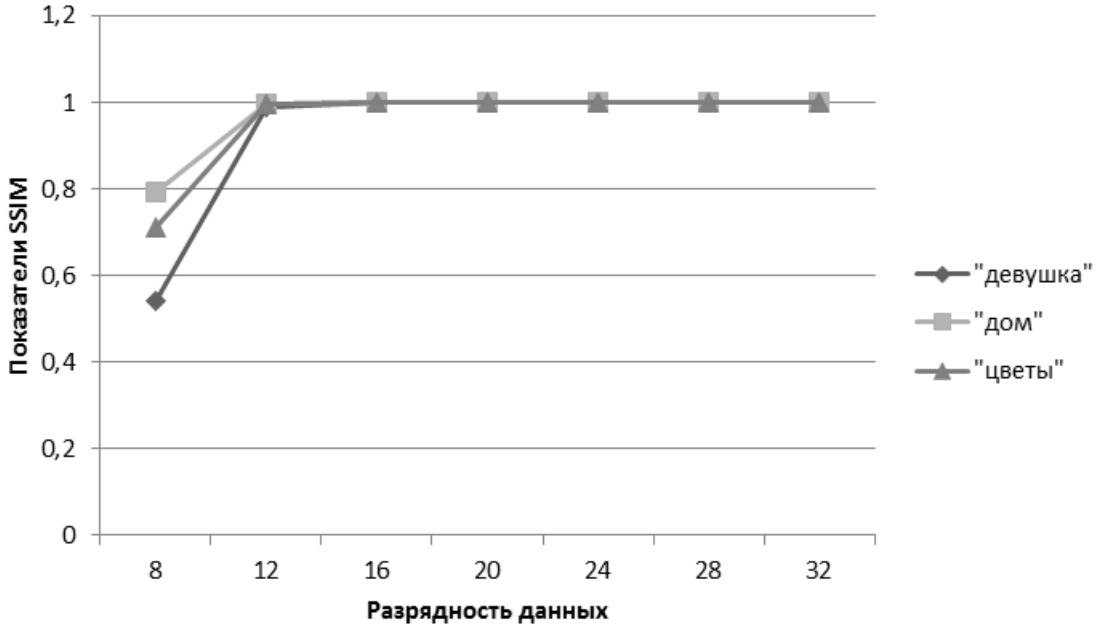


Рисунок 2.27. – Результаты SSIM изображений с уменьшением разрядности коэффициентов фильтра.

Из рисунка 2.27 видно, что при разрядности равной от 12 бит SSIM становится приблизительно равным единице, что свидетельствует о получение результирующего изображения близкого по качеству к исходному.

2.7 Разработка математических моделей ошибок цифровой обработки сигналов в СОК

Как было показано в предыдущем пункте, выбор правильного набора модулей влияет на работу цифровых фильтров. Но следует также отметить, что от выбранного набора модулей зависит и динамический диапазон работы системы, что в свою очередь может повлечь за собой некорректность полученных результатов при цифровой обработке изображений. Далее будет показано, как применение некоторых наборов модулей может приводить к существенным ошибками, искажающим изображение.

Как было сказано ранее, применение СОК является перспективным инструментом для повышения технических характеристик систем цифровой

обработки сигналов и в частности систем цифровой обработки изображений. В данном разделе будут рассмотрены изображения в оттенках серого. В таком формате изображение представляет собой прямоугольный массив целых чисел (пикселей). Количество уровней серого являются целыми степенями 2, то есть пиксель представляет яркость или темноту [118]. Таким образом, чем больше число, кодирующее пиксель, тем ярче изображение в этой точке. В стандартных приложениях обработки изображений величины пикселей изображения в оттенках серого кодируются 8-битными числами и находятся в диапазоне $[0,255]$ причем 0 обозначает черный цвет, 255 белый цвет. Если в результате обработки изображения получается отрицательное число, то оно заменяется на 0 (черный цвет). В случае получения числа, большего чем 255, оно заменяется на 255 (белый цвет). На рисунке 2.28 схематически представлен принцип отображения чисел в цвет пикселя изображения в оттенках серого.

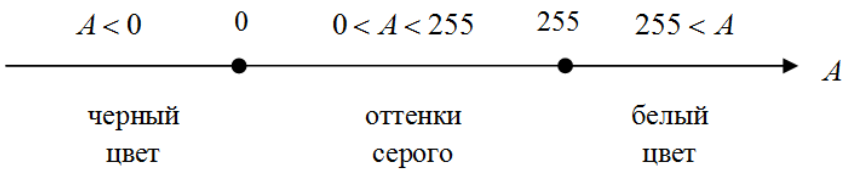


Рисунок 2.28. Зависимость цветности пикселя, представленного в оттенках серого, от величины кодового числа.

Процесс фильтрации изображения сводится к выполнению операции свертки частей изображения с элементами маски фильтра, которая представляет собой квадратную матрицу коэффициентов фильтра.

Пример 2.5. Предположим, что часть изображения в оттенках серого имеет следующие величины пикселей:

$$\begin{bmatrix} 0 & 255 & 0 \\ 255 & 0 & 255 \\ 0 & 255 & 0 \end{bmatrix}.$$

Пусть для обработки используется фильтр увеличения резкости [129]. Обоснование выбора такого вида фильтра обусловлено тем, что изображение с подчеркнутыми границами воспринимается как изображение высокого качества, чем точные репродукции. Мелкие объекты на таких изображениях обнаруживаются и опознаются быстрее и легче за счет выделенных границ и контуров при обработке такого рода фильтром.

Под границей понимается линия, разделяющая две области на изображении $L_C(x,y)$, на которой происходит скачок яркости. Граница может быть определена как геометрическое место точек, где абсолютное значение градиента функции достигает своего максимального значения, а оператор Лапласа:

$$\Delta(x,y)=\frac{\partial^2 L_C(x,y)}{\partial x^2}+\frac{\partial^2 L_C(x,y)}{\partial y^2}=0, \tag{2.25}$$

при условии, что он расположен между двумя своими экстремумами. Распределение яркости в изображении с подчеркнутыми границами вычисляется по формуле (2.26):

$$L_{CГ}(x,y)=L_C(x,y)-b\Delta(x,y) \tag{2.26}$$

Изменением величины весового коэффициента b можно влиять на степень подчеркивания границ. Операция подчеркивания границ с позиций фильтрации изображения в частотной области после выражения исходного изображения $L_C(x,y)$ через его спектр $M_C(\omega_X,\omega_Y)$примет вид (2.27):

$$L_C(x,y)=\frac{1}{4\pi^2}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}M_C(\omega_x,\omega_y)\exp\left[i(\omega_x x+\omega_y y)\right]d\omega_x d\omega_y \tag{2.27}$$

Для того чтобы написать выражение для оператора Лапласа, найдем вторые производные от $L_C(x,y)$ путем двойного дифференцирования выражения (2.27) по x и y:

$$\frac{\partial^2 L_C(x,y)}{\partial x^2}=\frac{1}{4\pi^2}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}(-1)\omega_x^2 M_C(\omega_x,\omega_y)\exp\left[i(\omega_x x+\omega_y y)\right]d\omega_x d\omega_y,$$

$$\frac{\partial^2 L_C(x,y)}{\partial y^2}=\frac{1}{4\pi^2}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}(-1)\omega_y^2 M_C(\omega_x,\omega_y)\exp\left[i(\omega_x x+\omega_y y)\right]d\omega_x d\omega_y,$$

при этом

$$\Delta(x,y)=\frac{1}{4\pi^2}\int_{-\infty-\infty}^{\infty}\int^{\infty}(-1)(\omega_x^2+\omega_y^2)M_C(\omega_x,\omega_y)\exp\left[i(\omega_x x+\omega_y y)\right]d\omega_x d\omega_y, \qquad (2.28)$$

Подставляя выражения для $L_C(x,y)$ и $\Delta(x,y)$ из (2.27) и (2.28) в формулу (2.26) получим:

$$L_{CГ}(x,y)=\frac{1}{4\pi^2}\int_{-\infty-\infty}^{\infty}\int^{\infty}M_C(\omega_x,\omega_y)\left[1+b(\omega_x^2+\omega_y^2)\right]\exp\left[i(\omega_x x+\omega_y y)\right]d\omega_x d\omega_y \qquad (2.29)$$

Сопоставляя формулы (2.27) и (2.29), видим, что исходное изображение и изображение с подчеркнутыми границами различаются своими спектрами. Спектр исходного изображения и спектр изображения с подчеркнутыми границами различаются между собой множителем $\left[1+b(\omega_x^2+\omega_y^2)\right]$, который можно рассматривать как частотную передаточную функцию линейного фильтра, подчеркивающего границы.

В другом способе подчеркивания границ достигается путем свертки исходного изображения $L_C(k,n)$ с импульсной характеристикой вида:

$$h(k,n)=\begin{Vmatrix}0 & -b & 0\\ -b & 1+4b & -b\\ 0 & -b & 0\end{Vmatrix} \qquad (2.30)$$

При этом изображение с подчеркнутыми границами вычисляется по формуле

$$L_{CГ}(k,n)=\sum_{k'=-1}^{1}\sum_{n'=-1}^{1}L_C(k+k',n+n')h(k',n')$$

Подставляя в выражение (2.30) $b=1$, получаем фильтр повышения резкости (2.31)

$$\begin{bmatrix}0 & -1 & 0\\ -1 & 5 & -1\\ 0 & -1 & 0\end{bmatrix}. \qquad (2.31)$$

Указанный фильтр известен еще как контрастоповышающий фильтр высоких пространственных частот. При использовании этого фильтра происходит повышение локальной контрастности изображения, за счет чего увеличивается его резкость. Эффект повышения резкости достигается за счет того, что фильтр подчеркивает разницу между интенсивностями соседних

пикселей, удаляя эти интенсивности друг от друга. Причем, эффект резкости будет тем сильнее, чем больше значение центрального члена ядра. Ядро фильтра имеет значение больше 1 в точке $(0,0)$ и при этом, все значения в сумме дают 1.

Применив фильтр повышения резкости, в двоичной системе счисления получим следующую величину пикселя после осуществления фильтрации:

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix} \times \begin{bmatrix} 0 & 255 & 0 \\ 255 & 0 & 255 \\ 0 & 255 & 0 \end{bmatrix} = -1020.$$

Число -1020 получено как сумма произведений соответствующих элементов матриц. Так как отрицательные величины рассматриваются как 0, то в нашем случае -1020 это черный цвет.

Как говорилось ранее, при использовании приложений цифровой обработки изображений особое внимание следует уделить выбору правильного набора модулей, который будет в достаточной мере давать динамический диапазон. В работах [121] и [130] предлагаются наборы модулей $\{5,7,8\}$ и $\{7,15,16\}$, соответственно, и утверждается о достаточности их динамического диапазона при цифровой обработке изображений. На примере покажем, что это не всегда корректно.

Пример 2.6. При использовании СОК с модулями $\{5,7,8\}$, имеем следующий результат для значений из примера 2:

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix} \times \begin{bmatrix} 0 & 255 & 0 \\ 255 & 0 & 255 \\ 0 & 255 & 0 \end{bmatrix} =$$

$$= \begin{bmatrix} \{0,0,0\} & \{4,6,7\} & \{0,0,0\} \\ \{4,6,7\} & \{0,5,5\} & \{4,6,7\} \\ \{0,0,0\} & \{4,6,7\} & \{0,0,0\} \end{bmatrix} \times \begin{bmatrix} \{0,0,0\} & \{0,3,7\} & \{0,0,0\} \\ \{0,3,7\} & \{0,0,0\} & \{0,3,7\} \\ \{0,0,0\} & \{0,3,7\} & \{0,0,0\} \end{bmatrix} = \{0,2,4\} = 100.$$

Диапазон СОК $\{5,7,8\}$ равен $P = 5 \cdot 7 \cdot 8 = 280$. Так как полученное число 100 попадает в первую половину диапазона $0 \le 100 \le 140$, то оно является положительным в данной СОК, и поэтому в дальнейшем не меняется. Число 100

дает серый цвет обработанного изображения, что не совпадает с результатом, полученным в двоичной системе счисления. Очевидна разница между двумя системами счисления, а именно значения -1020 и 100, дающие разные цвета.

Сравнение примеров 2 и 3 показывает, что набор модулей $\{5,7,8\}$ дает искаженный результат фильтрации изображения из-за недостаточного динамического диапазона.

В работе [130] используется набор модулей $\{7,15,16\}$ (специальный набор модулей $\{2^{n-1}-1,2^{n-1},2^n\}$ при $n=4$), который также считается авторами достаточным для фильтрации. Покажем на примере недостаточность динамического диапазона и для этого набора модулей.

Пример 2.7. Пусть часть изображения в оттенках серого имеет следующие величины пикселей:

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 255 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

Используя тот же самый фильтр, получим в двоичной системе счисления:

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix} \times \begin{bmatrix} 0 & 0 & 0 \\ 0 & 255 & 0 \\ 0 & 0 & 0 \end{bmatrix} = 1275.$$

Число $1275 > 255$ и поэтому рассматривается как белый цвет в стандартных приложениях обработки изображений.

При фильтрации в СОК, с набором модулей $\{7,15,16\}$, имеем:

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix} \times \begin{bmatrix} 0 & 0 & 0 \\ 0 & 255 & 0 \\ 0 & 0 & 0 \end{bmatrix} =$$

$$= \begin{bmatrix} \{0,0,0\} & \{6,14,15\} & \{0,0,0\} \\ \{6,14,15\} & \{5,5,5\} & \{6,14,15\} \\ \{0,0,0\} & \{6,14,15\} & \{0,0,0\} \end{bmatrix} \times \begin{bmatrix} \{0,0,0\} & \{0,0,0\} & \{0,0,0\} \\ \{0,0,0\} & \{3,0,15\} & \{0,0,0\} \\ \{0,0,0\} & \{0,0,0\} & \{0,0,0\} \end{bmatrix} = \{1,0,11\} = 1275.$$

Диапазон СОК $\{7,15,16\}$ равен $P = 7 \cdot 15 \cdot 16 = 1680$. Число 1275 попадает во вторую часть динамического диапазона СОК: $840 \le 1275 \le 1679$. Поэтому окончательное восстановление числа дает в результате $1275 - 1680 = -405$. Полученное отрицательное число дает черный цвет изображения, вместо белого цвета, что показывает некорректный результат фильтрации и при использовании набора модулей $\{7,15,16\}$.

Полученные результаты показывают важность выбора правильного набора модулей, обеспечивающего достаточный динамический диапазон. Использование недостаточного диапазона СОК может привести к неверному результату, поэтому следует использовать критерии определения переполнения. Мы предлагаем следующий критерий определения необходимого диапазона СОК:

$$P > 2 \cdot 255 \cdot \max\{\pi, \nu\}, \tag{2.32}$$

где $\pi = \sum_{a_{i,j} > 0} a_{ij}$ - сумма положительных коэффициентов маски фильтра; $\nu = \left| \sum_{a_{ij} < 0} a_{ij} \right|$ - модуль суммы отрицательных коэффициентов маски фильтра.

Отметим, что используя предложенный критерий можно рассчитать вероятность Q возникновения ошибки при цифровой обработки изображений. При этом вероятность будет рассчитываться по формуле (2.33):

$$Q = \begin{cases} 1, & \text{если } P \ge 2 \cdot 255 \cdot \max\{\pi, \nu\} \\ \dfrac{P}{2 \cdot 255 \cdot \max\{\pi, \nu\}}, & \text{если } P < 2 \cdot 255 \cdot \max\{\pi, \nu\} \end{cases} \tag{2.33}$$

Пример 2.8. Пусть задана маска вида: $\begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix}$. Тогда получаем значения $\pi = 8$ и $\nu = 8$. Рассчитаем значение динамического диапазона по формуле (2.15): $P = 2 \cdot 255 \cdot 8 = 4080$. Тогда для набора модулей $\{5,7,8\}$, который дает динамический диапазон равный $P = 5 \cdot 7 \cdot 8 = 280$ вероятность возникновения ошибки будет равна $100\% - 6{,}8\% = 93{,}2\%$. Для набора модулей $\{7,15,16\}$ с

динамическим диапазоном равным $P = 7 \cdot 15 \cdot 16 = 1680$ вероятность ошибки: $100\% - 41{,}2\% = 58{,}8\%$.

Практическую проверку предложенного критерия будем проводить на тех же типах фильтров, которые были рассмотрены в работах [121, 130], а именно на фильтрах повышения резкости и определения контуров изображения. Данные типы фильтров весьма широко используются на практике при цифровой обработке изображений. В то же время матрицы масок таких фильтров имеют весьма удобный вид для работы в СОК, так как в качестве элементов содержат только целые числа [118]. В дальнейшей части работы мы будем использовать два фильтра повышения резкости $\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$, $\begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix}$ и фильтр определения контуров изображения $\begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix}$ [118]. Применение предложенного критерия для этих фильтров дает следующие требования к диапазону СОК.

Для фильтра $\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$:

$$P > 2 \cdot 255 \cdot \max\{5, |-4|\} = 2550.$$

Для фильтра $\begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix}$:

$$P > 2 \cdot 255 \cdot \max\{9, |-8|\} = 4590.$$

Для фильтра $\begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix}$:

$$P > 2 \cdot 255 \cdot \max\{8, |-8|\} = 4080.$$

Для того, чтобы выполнить самое сильное из вышеперечисленных условий, а именно $P > 4590$, нет возможности использовать трехмодульные наборы СОК со всеми модулями, имеющими удобный вид 2^n или $2^n \pm 1$ и требующими не более 4 бит ($p_i \le 16$) для представления данных. Необходимо либо использовать трехмодульные наборы с 5-битными модулями $\{17,31,32\}$, $\{15,31,32\}$ либо увеличить количество модулей СОК до четырех, но при этом остаться в рамках 4-битного представления информации по модулям: $p_i \le 16$. Второй вариант представляется более предпочтительным, так как введение дополнительного модуля увеличит параллельность обработки данных, при этом время выполнения операций сложения и умножения не будет увеличено, так как они будут по-прежнему выполняться в параллельном режиме над 4-битными числами вместо обработки 5-битных чисел в трехмодульных СОК. Поэтому, в дальнейшем, для реализации фильтрации изображений с использованием указанных масок мы будем использовать набор модулей $\{5,7,9,16\}$. Все модули данного набора имеют удобный вид для практической реализации операций: 2^n, $2^n \pm 1$ [110]. Диапазон СОК $\{5,7,9,16\}$ равен $P = 5040$ и удовлетворяет предложенному критерию (6) для всех фильтров.

Покажем корректность выбора этого набора модулей на примерах, использованных выше.

Пример 2.9. Использование СОК с набором модулей $\{5,7,9,16\}$ для обработки данных из примера 4 дает следующий результат:

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix} \times \begin{bmatrix} 0 & 0 & 0 \\ 0 & 255 & 0 \\ 0 & 0 & 0 \end{bmatrix} =$$

$$= \begin{bmatrix} \{0,0,0,0\} & \{4,6,8,15\} & \{0,0,0,0\} \\ \{4,6,8,15\} & \{0,5,5,5\} & \{4,6,8,15\} \\ \{0,0,0,0\} & \{4,6,8,15\} & \{0,0,0,0\} \end{bmatrix} \times \begin{bmatrix} \{0,0,0,0\} & \{0,0,0,0\} & \{0,0,0,0\} \\ \{0,0,0,0\} & \{0,3,3,15\} & \{0,0,0,0\} \\ \{0,0,0,0\} & \{0,0,0,0\} & \{0,0,0,0\} \end{bmatrix} =$$

$$= \{0,1,6,11\} = 1275$$

Число 1275 попадает в первую часть динамического диапазона СОК: $0 \le 1275 \le 2519$ и, следовательно, является положительным (в дальнейшем не изменяется). Значению 1275 соответствует белый цвет изображения, что в точности совпадает с результатом, полученным в двоичной системе счисления.

Пример 2.10. Использование СОК с набором модулей $\{5,7,9,16\}$ для обработки данных из примера 2 дает следующий результат:

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix} \times \begin{bmatrix} 0 & 255 & 0 \\ 255 & 0 & 255 \\ 0 & 255 & 0 \end{bmatrix} =$$

$$= \begin{bmatrix} \{0,0,0,0\} & \{4,6,8,15\} & \{0,0,0,0\} \\ \{4,6,8,15\} & \{0,5,5,5\} & \{4,6,8,15\} \\ \{0,0,0,0\} & \{4,6,8,15\} & \{0,0,0,0\} \end{bmatrix} \times \begin{bmatrix} \{0,0,0,0\} & \{0,3,3,15\} & \{0,0,0,0\} \\ \{0,3,3,15\} & \{0,0,0,0\} & \{0,3,3,15\} \\ \{0,0,0,0\} & \{0,3,3,15\} & \{0,0,0,0\} \end{bmatrix} =$$

$$= \{0,2,6,4\} = 4020$$

Число 4020 попадает во вторую часть динамического диапазона СОК: $2520 \le 4020 \le 5040$, т.е. является отрицательным числом. Окончательное значение величины числа получается путем вычитания из результата полного диапазона СОК и равно $4020 - 5040 = -1020$. Числу -1020 соответствует черный цвет в изображении в оттенках серого, что в точности совпадает с результатом, полученным в двоичной системе счисления.

Далее будут описаны результаты цифровой обработки различных изображений с использованием разных фильтров и разных систем счисления.

Для моделирования были использованы СОК с модулями $\{5,7,8\}$, $\{7,15,16\}$ и $\{5,7,9,16\}$, а также традиционная двоичная система счисления. В качестве фильтров использовались следующие маски.

Фильтры повышения резкости:

$$1) \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}; \qquad 2) \begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix}.$$

Фильтр определения контуров:

$$3)\begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix}.$$

Далее по тексту эти фильтры будут обозначаться как фильтр-1, фильтр-2 и фильтр-3, соответственно.

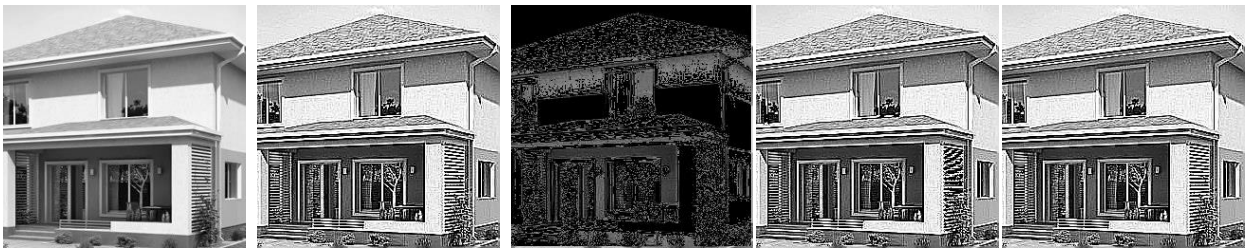
Все вычисления производились с использованием пакета прикладных программ MATLAB [95].

На рисунке 2.29 показаны результаты применения фильтра-1 к изображению «девушка». Оригинальное входное изображение в оттенках серого показано на рисунке 2.29а. Выходное отфильтрованное изображение в двоичной системе счисления показано на рисунке 2.29б. На рисунках 2.29в-2.29д показано изображение, обработанное указанным фильтром в СОК с модулями $\{5,7,8\}$, $\{7,15,16\}$ и $\{5,7,9,16\}$. Из рисунка 2.29 видно крайне низкое качество обработки изображения в СОК $\{5,7,8\}$. Данный факт объясняется слишком малым динамическим диапазоном такой СОК, что приводит к серьезным искажениям пикселей изображения. Результаты работы СОК $\{7,15,16\}$ и $\{5,7,9,16\}$ визуально неотличимы от изображения, обработанного в двоичной системе счисления.

а) б) в) г) д)

Рисунок 2.29. Результат обработки фильтром-1 изображения «девушка»: а) исходное изображение, б) изображение, обработанное в двоичной системе счисления, в) изображение, обработанное в СОК $\{5,7,8\}$, г) изображение, обработанное в СОК $\{7,15,16\}$, д) изображение, обработанное в СОК $\{5,7,9,16\}$.

На рисунке 2.30 показаны результаты применения фильтра-2 к изображению «дом». На рисунке 2.30а показано оригинальное входное изображение в оттенках серого. Отфильтрованное изображение в двоичной системе счисления представлено на рисунке 2.30б. Изображения, полученные в результате обработки указанным фильтром в СОК с наборами модулей $\{5,7,8\}$, $\{7,15,16\}$ и $\{5,7,9,16\}$ показаны на рисунках 2.30в-2.30д соответственно. Снова видно низкое качество обработки при использовании СОК с модулями $\{5,7,8\}$. Результаты обработки в СОК с модулями $\{7,15,16\}$ и $\{5,7,9,16\}$ хорошего качества, однако изображение, обработанное в СОК $\{7,15,16\}$ не идентично изображению, полученному при использовании двоичной системы счисления.

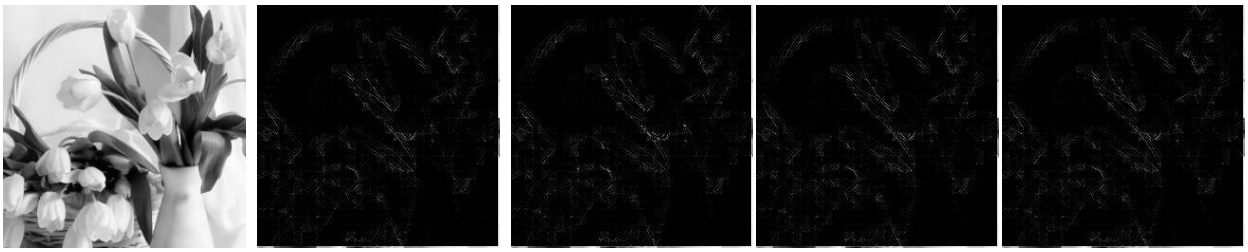


а) б) в) г) д)

Рисунок 2.30. Результат обработки фильтром-2 изображения «дом»: а) исходное изображение, б) изображение, обработанное в двоичной системе счисления, в) изображение, обработанное в СОК $\{5,7,8\}$, г) изображение, обработанное в СОК $\{7,15,16\}$, д) изображение, обработанное в СОК $\{5,7,9,16\}$.

Результаты применения фильтра-3 для обработки изображения «тюльпаны» представлены на рисунке 2.31. На рисунке 2.31а показано исходное изображение в оттенках серого. На рисунке 2.31б представлен результат фильтрации изображения в двоичной системе счисления. Рисунки 2.31в-2.31д показывают результаты фильтрации изображения в СОК с модулями $\{5,7,8\}$, $\{7,15,16\}$ и $\{5,7,9,16\}$. Визуально, изображения 2.31в-2.31д не сильно отличаются от

изображения 2.31б, однако в действительности изображения 2.31г и 2.31д тождественны изображению 2.31б, а изображение 2.31в отличаются от 2.31б.



а) б) в) г) д)

Рисунок 2.31. Результат обработки фильтром-3 изображения «тюльпаны»: а) исходное изображение, б) изображение, обработанное в двоичной системе счисления, в) изображение, обработанное в СОК $\{5,7,8\}$, г) изображение, обработанное в СОК $\{7,15,16\}$, д) изображение, обработанное в СОК $\{5,7,9,16\}$.

Сравнивая результаты работы фильтров в разных СОК можно сделать вывод, что предложенные в работах [121] и [130] наборы модулей $\{5,7,8\}$ и $\{7,15,16\}$, соответственно, могут давать искаженный результат обработки изображения. Это связанно с недостаточностью динамических диапазонов рассматриваемых СОК. Наихудший результат показывает СОК с модулями $\{5,7,8\}$, так как даже визуально качество обработки в ней изображений фильтрами повышения резкости неприемлемо на практике. С другой стороны, обработка изображения фильтром выделения контуров в этих СОК показала достаточно неплохое визуальное качество.

Для количественной оценки качества обработки изображений с использованием разных наборов модулей СОК были использованы две числовые характеристики.

1. PSNR, или пиковое отношение сигнал-шум, между двумя изображениями (оригиналом и полученным изображением). Вычисляется данная характеристика по формуле:

$$PSNR = 10\log_{10}\left(\frac{R^2}{MSE}\right), \tag{2.34}$$

где $MSE = \frac{\sum_{M,N}\left[I_1(m,n) - I_2(m,n)\right]^2}{M \cdot N}$ – среднеквадратическая ошибка сравнения качества восстановленного изображения; R – максимальное колебание входного сигнала изображения. Так как величина $PSNR$ имеет логарифмическую природу, единицей ее измерения является децибел (Дб). Чем больше величина PSNR, тем лучше качество восстановленного изображения, для тождественно равных изображений $PSNR = \infty$. При исследовании алгоритмов сжатия и очистки от шума изображений в оттенках серого практически пригодной считается величина $PSNR$, изменяющаяся в пределах от 20 Дб до 50 Дб [94]. Для рассматриваемых нами случаев фильтрации в СОК с разными наборами модулей вычислялась величина $PSNR$ между изображением, полученным при использовании традиционной двоичной системы счисления и изображением, полученным с использованием СОК.

2. SSIM, или индекс структурного сходства между двумя изображениями, определяется на основе полного сопоставления исходного и полученного изображений [125]. Данная характеристика вычисляется по формуле:

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{2.35}$$

где μ_x - среднее x, μ_y - среднее y, σ_x^2 - дисперсия x, σ_y^2 - дисперсия y, σ_{xy} - ковариация x и y, $c_1 = (k_1 L)^2$, $c_2 = (k_2 L)^2$ - две переменных, L - динамический диапазон пикселей, $k_1 = 0{,}01$ и $k_2 = 0{,}03$ константы. Величина SSIM находится в пределах между 0 и 1 и равна 1 для тождественно равных изображений. Для рассматриваемых нами случаев фильтрации в СОК с разными наборами модулей вычислялась величина $SSIM$ между изображением, полученным при использовании традиционной двоичной системы счисления и изображением, полученным с использованием СОК.

Таблица 2.10. Результаты моделирования обработки изображений в СОК с различными наборами модулей.

изображение	фильтр	модули СОК					
		$\{5,7,8\}$		$\{7,15,16\}$		$\{5,7,9,16\}$	
		PSNR, Дб	SSIM	PSNR, Дб	SSIM	PSNR, Дб	SSIM
«девушка»	фильтр-1	1.5188	0.1384	∞	1	∞	1
	фильтр-2	1.5304	0.1255	43.3496	0.9998	∞	1
	фильтр-3	29.8721	0.9806	∞	1	∞	1
«дом»	фильтр-1	4.3506	0.1010	∞	1	∞	1
	фильтр-2	4.1741	0.0535	23.2278	0.9879	∞	1
	фильтр-3	27.6603	0.9683	∞	1	∞	1
«тюльпаны»	фильтр-1	3.2489	0.0963	∞	1	∞	1
	фильтр-2	3.2068	0.0628	42.1221	0.9997	∞	1
	фильтр-3	35.2163	0.9899	∞	1	∞	1

В таблице 2.10 представлены результаты оценки качества обработанных изображений фильтрами (1-3) в СОК с использованием различных наборов модулей. Анализируя полученные результаты, можно сделать вывод, что набор модулей $\{5,7,9,16\}$ показывает абсолютно точный результат отфильтрованного изображения, во всех случаях совпадающий с результатом обработки в двоичной системе счисления. Набор модулей $\{5,7,8\}$, предложенный в работе [121], дает приемлемый результат ($PSNR > 20$ Дб) при использовании фильтра выделения контуров (фильтр-3) и плохой результат ($PSNR < 20$ Дб) при использовании фильтра-1 и фильтра-2. Применение набора модулей $\{5,7,8\}$ не гарантирует корректной работы фильтров повышения резкости и может ограниченно применяться при выделении контуров изображения. Набор модулей $\{7,15,16\}$ из работы [130] не показал ошибок при фильтрации изображений фильтром-1 и фильтром-3. При использовании фильтра-2 качество изображения ухудшалось, но

оставалось на приемлемом уровне ($PSNR > 20$ Дб) для практического применения. Следует, однако, учесть, что возникновение ошибок при использовании СОК с модулями $\{7,15,16\}$ теоретически возможно (пример 4) и при фильтрации фильтром-1 и при фильтрации фильтром-3. Отсутствие ошибок в обработке данных изображений объясняется отсутствием резких переходов в значениях соседних пикселей от 0 до 255 и обратно, чего может и не случиться при обработке каких-либо других изображений.

2.8 Выводы по второй главе

Основные результаты данной главы состоят в следующем:

1. Разработана модель нейросетевого фильтра с конечной импульсной характеристикой в системе остаточных классов. Представлены результаты исследования производительности различных наборов модулей системы остаточных классов для цифровой обработки сигналов на основе фильтров с конечной импульсной характеристикой. Показано, что наилучшую скорость фильтрации обеспечивают модули $\left\{2^n-1,2^n,2^n+1,2^n-2^{\frac{n+1}{2}}+1,2^n+2^{\frac{n+1}{2}}+1\right\}$ и $\left\{2^n-1,2^n,2^n+1,2^{n+1}-1\right\}$. Разница в производительности между наилучшим и наихудшим наборами модулей достигает 93%. Полученные результаты могут быть использованы для улучшения характеристик устройств цифровой обработки сигналов в системе остаточных классов и применены для обработки изображений.

2. Разработана новая архитектурная организация цифровой фильтрации на основе интеграции СОК и ИНС. Данная схема позволяет получать корректные результаты обработки. Установлено, что применение предложенного метода обеспечивает безошибочную обработку изображений в оттенках серого со значением вероятности наличия ошибок из-за переполнения динамического диапазона вычислительной системы, равной 1, по сравнению со значениями

вероятности равными 0,58 и 0,93 для известных наборов модулей $\{5,7,8\}$ и $\{7,15,16\}$.

3. Разработан новый численный метод решения задачи фильтрации изображений на основе вычислений в СОК. Данный метод позволяет сократить время необходимое на выполнение вычислений в ходе процесса цифровой фильтрации, за счет чего соответственно повышается скорость работы цифрового устройства.

4. Исследован вопрос о выборе корректного динамического диапазона системы остаточных классов для задач цифровой обработки изображений. Показана некорректность обработки изображений в оттенках серого с использованием предложенных другими исследователями наборов модулей СОК $\{5,7,8\}$ и $\{7,15,16\}$. Сделан вывод о недостаточности динамического диапазона для таких модулей. Предложен критерий определения достаточного динамического диапазона СОК, зависящий от коэффициентов маски фильтра. Проведено моделирование обработки конкретных изображений в оттенках серого с использованием СОК.

5. Исследован вопрос о влиянии снижения разрядности коэффициентов цифрового фильтра при вейвлет-обработке изображений в оттенках серого. Показано, что снижение разрядности коэффициентов вейвлетного фильтра ведет к снижению качества получаемого изображения при обработке. Установлено, что пригодное на практике изображение можно получить при использовании 12 и более бит представления разрядности коэффициентов, а визуально неотличимое при использовании 28 и более бит разрядности коэффициентов фильтра.

ГЛАВА 3. РАЗРАБОТКА АРХИТЕКТУР СВЕРТОЧНОЙ НЕЙРОННОЙ СЕТИ С ВЫЧИСЛЕНИЯМИ В СИСТЕМЕ ОСТАТОЧНЫХ КЛАССОВ

Перспективным инструментом в решении задачи распознавания образов, или, другими словами, классификации изображений, является сверточная нейронная сеть (СНС) [54]. Общий принцип работы такой сети заключается в том, что на начальном этапе входное изображение пропускается через серию сверточных, нелинейных слоев, слоев объединения и полносвязных слоев, а на конечном этапе генерируется результат (вывод). При этом выводом может быть класс или вероятность классов, которые лучше всего описывают изображение. Поскольку, качество работы нейронной сети прямо-пропорционально зависит от ее глубины и ширины, т. е. при увеличении количества слоев сеть может увеличить точность распознавания, то по этой причине желателен большой масштаб сверточной нейронной сети. Поэтому при решении задачи распознавания образов целесообразней рассматривать глубокую СНС.

Глубокая нейронная сеть состоит из модели многослойной нейронной сети. Глубокая сверточная нейронная сеть представляет собой сочетание двумерных сверточных слоев и глубокой нейронной сети. Эта сеть способна имитировать человеческое зрение, что в свою очередь ведет к высокой точности распознавания образов. По этой причине, алгоритмы на основе работы СНС нашли широкое применение во встраиваемых системах технического зрения, включающих в себя решение задач распознавания рукописного текста [99], распознавание лиц [116], распознавание места [113] и распознавание объектов [87].

Существуют различные подходы, методы и средства проектирования СНС с помощью которых можно получить наиболее перспективную ее архитектуру. В [109] предложен подход в реализации СНС на основе вложенной Системы Остаточных Классов (СОК) с применением известной архитектуры блока MAC (блок умножения-накопления). В [86] показано, что процесс двумерной свертки для глубокой сверточной нейронной сети занимает более 90% времени

вычислений. Таким образом, целесообразно укорить процесс свертки за счет использования массива операций MAC. Выбор применения СОК в архитектуре СНС обусловлен свойством параллельного выполнения операций, что в свою очередь хорошо сочетается с параллельной структурой СНС.

В данной главе будет рассмотрена архитектура СНС с применением модифицированного блока MAC за счет применения LUT-таблиц, а также предложена архитектура блоков СНС без применения LUT-таблиц, вложенную СОК заменит традиционная СОК с модулями специального вида.

3.1 Разработка архитектуры глубокой сверточной нейронной сети на основе вложенной системы остаточных классов

Рассмотрим глубокую сверточную нейронную сеть. Такая сеть состоит из нескольких карт признаков и классификатора. На первом этапе распознавания входного изображения, карта признаков вначале реагирует на соответствующий класс данных. После этого, классификатор выбирает соответствующие реакции из карт признаков. Как правило, классификатор работает под действием нейронной сети или при поддержки вектора машины (SVM). Типичная карта признаков состоит из двумерного сверточного слоя, функции активации и слоя подвыборки [70].

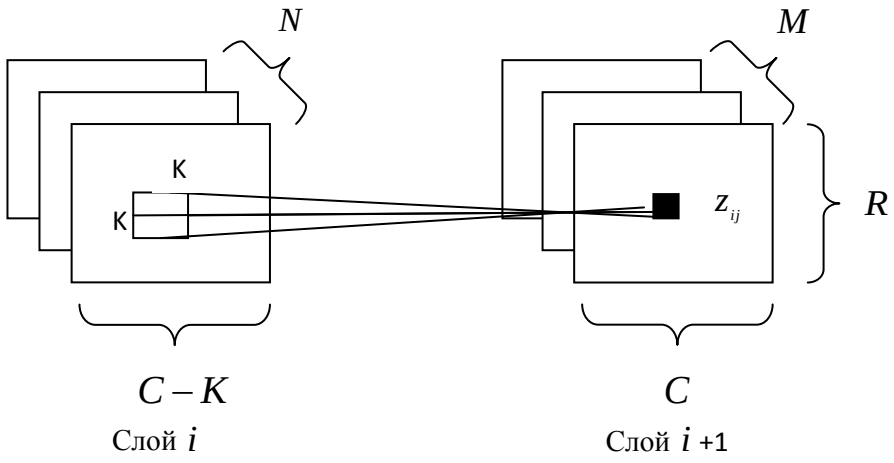


Рисунок 3.1. Двумерный сверточный нейронный слой

На рисунке 3.1 показан двумерный сверточный слой нейронной сети, где N обозначает количество входных слоев, M обозначает число выходных слоев. При этом, двумерный сверточный слой вычисляет выход путем сдвига ядра (или другими словами фильтра или нейрона) размером $K \times K$ [132].

Механизм процесса свертки изображения в формате RGB представлен на рисунке 3.2. На вход системы подается изображение, представленное в виде трех матриц. Каждая матрица соответствует определенному цвету из цветового пространства представления изображения [11].

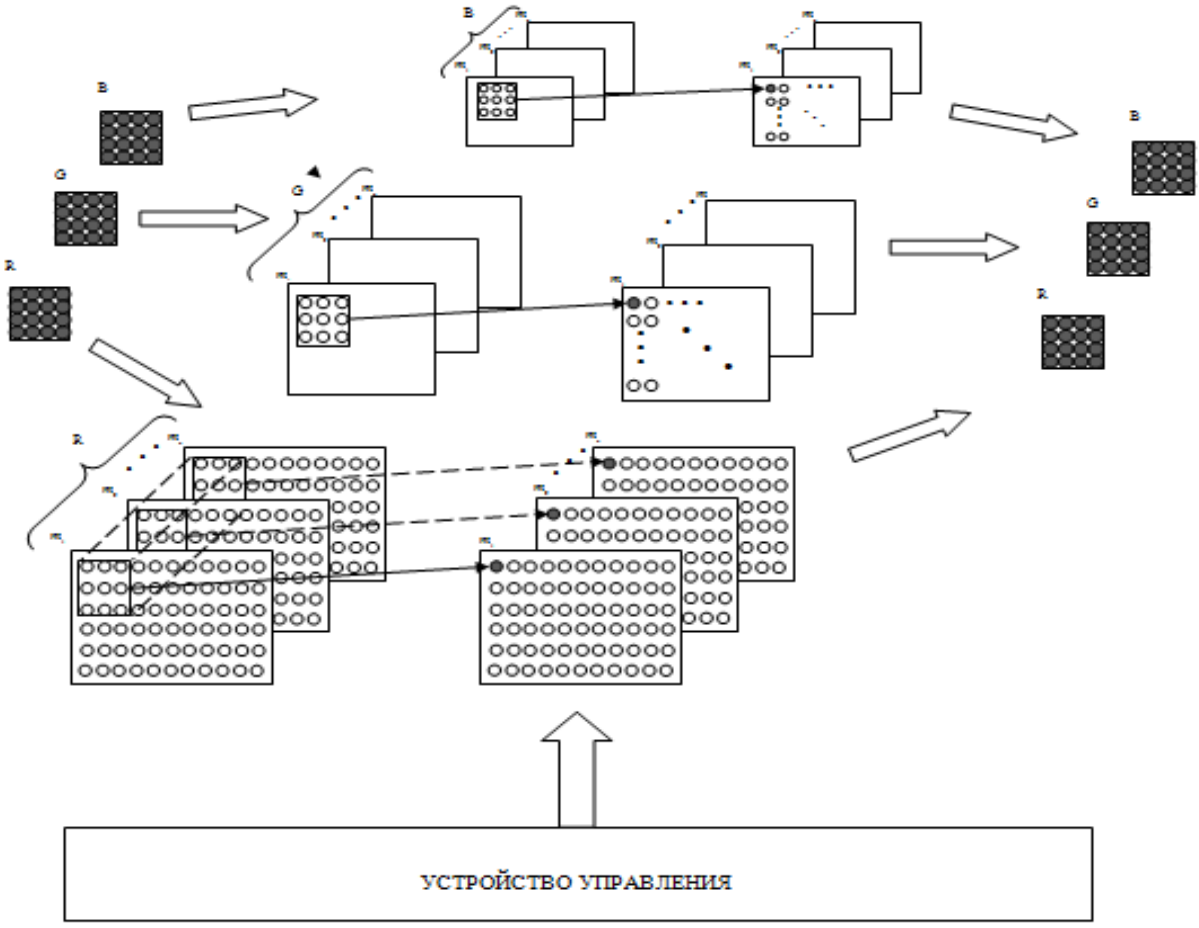


Рисунок 3.2. Архитектура процесса свертки изображения в формате RGB с вычислениями в СОК.

В данном случае это матрицы из пространства RGB, которые состоят из трех компонентов красного, зеленого и синего цвета. На следующем этапе

происходит процесс обработки каждой матрицы по всем модулям системы по средствам операции прямого преобразования ПСС-СОК. Результатом такой операции является $3n$ матриц по модулям СОК для каждого цветного компонента соответственно. Далее для каждой полученной матрицы вычисляется результат свертки этой матрицы на ядро размером 3×3. Следует отметить, что каждый процесс свертки с выбранным шагом осуществляется параллельно. Для синхронизации свертки по каждым модулям используется сигнал устройства управления. Для восстановления полученного изображения применяется операция обратного преобразования СОК-ПСС [69]. Схема, предложенная на рисунке 3.2. обладает высокой степенью параллелизма за счет использования арифметики остаточных классов.

Пример 3.1. иллюстрирует обработку изображения, представленного одной матрицей, такой случай возможен при обработке изображения в оттенках серого. Предположим, что часть изображения в оттенках серого имеет следующие величины пикселей:

$$\begin{bmatrix} 10 & 20 & 30 \\ 40 & 50 & 60 \\ 70 & 80 & 90 \end{bmatrix}$$

Пусть для обработки используется фильтр увеличения резкости [130]:

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}.$$

При использовании СОК с модулями $\{15,31,32\}$, имеем следующий результат:

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix} \times \begin{bmatrix} 10 & 20 & 30 \\ 40 & 50 & 60 \\ 70 & 80 & 90 \end{bmatrix} = \begin{bmatrix} \{0,0,0\} & \{14,30,31\} & \{0,0,0\} \\ \{14,30,31\} & \{5,5,5\} & \{14,30,31\} \\ \{0,0,0\} & \{14,30,31\} & \{0,0,0\} \end{bmatrix} \times$$

$$= \begin{bmatrix} \{10,10,10\} & \{5,20,20\} & \{0,30,30\} \\ \{10,9,8\} & \{5,19,18\} & \{0,29,28\} \\ \{10,8,6\} & \{5,18,16\} & \{0,28,26\} \end{bmatrix} = \{5,19,18\}$$

Механизм процесса свертки для значений, которые использовались выше, представлен на рисунке 3.3.

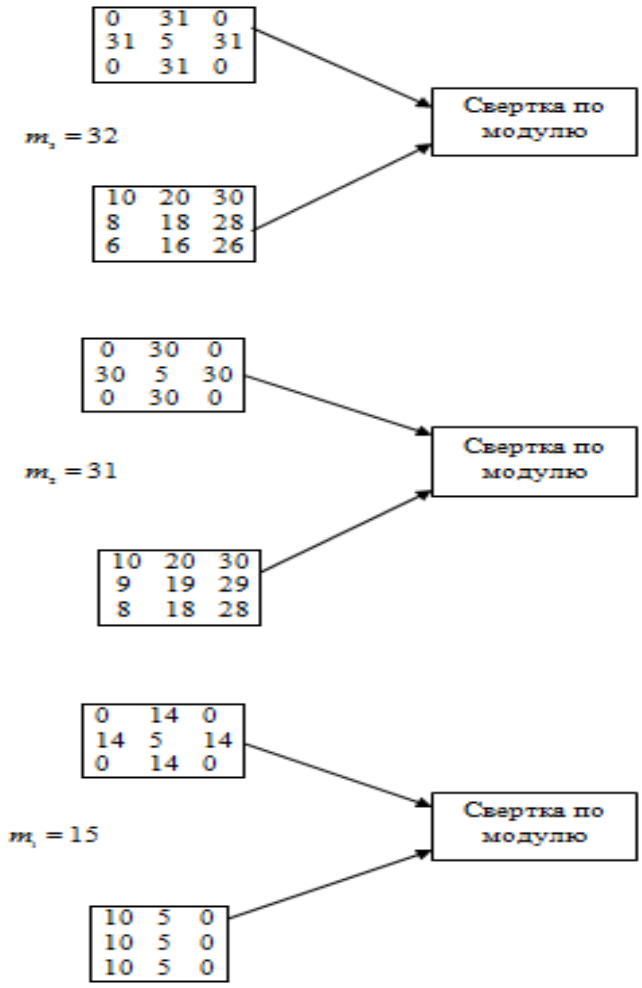


Рисунок 3.3. Процесс свертки части изображения в оттенках серого в СОК $\{15,31,32\}$.

Рисунок 3.4 представляет собой сеть ImageNet [109], которая является реальным приложением глубокой СНС. Сеть состоит из 8 слоев, внешние пять слоев состоят из двумерных сверточных слоев, а внутренние три слоя состоят из полностью подключенных нейронных сетей. При этом, СНС имеет двойные слои 1-7. Работа такой сети состоит в следующем, вначале она получает 3 входных RGB изображения размера 224 × 224 и выводит после обработки первым слоем пару слоев по 48 карт признаков каждая размером 55×55. Размер ядра 1 слоя составляет 11 × 11 и при этом, поле восприятия смещается через вход

изображения с шагом в 4 пикселя. Следующая карта признаков имеет аналогичную структуру. Шаг для слоя 2-5 составляет 1 пиксель. Выходным результатом СНС является возможность классифицировать полученные изображения в 1000 классов.

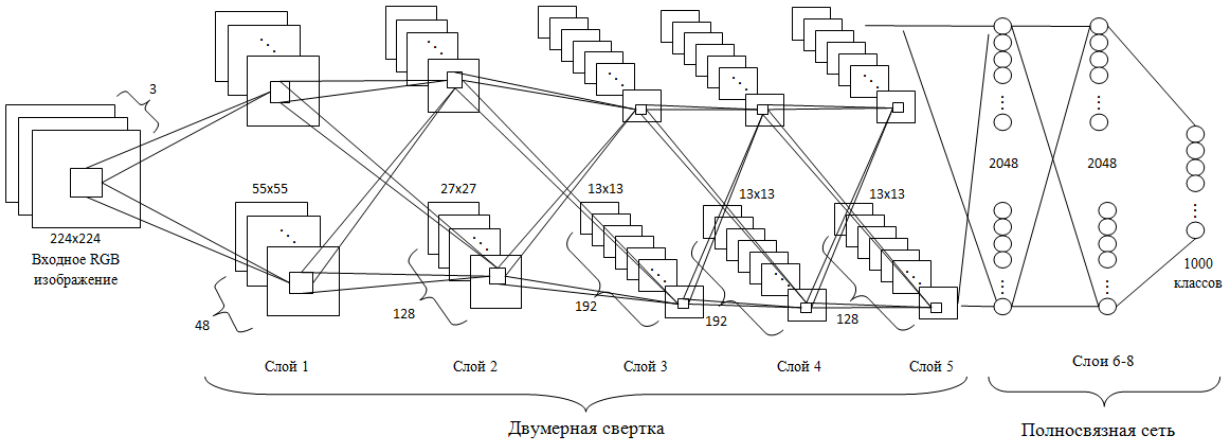


Рисунок 3.4. Пример архитектуры сверточной нейронной сети ImageNet

Параметры СНС ImageNet представлены в таблице 3.1. Из этой таблицы видно, что для одного слоя сети с ядром максимального размера 11×11 требуется 121 операция блока MAC [109].

Таблица 3.1 – Параметры для сети ImageNet

Слой	M	N	R	C	K
1	3	48	55	55	11
2	48	128	27	27	5
3	256	192	13	13	3
4	192	192	13	13	3
5	192	128	13	13	3

Для M карт признаков размером $R \times C$, выполняется операция MAC (3.1):

$$z_{ij} = \sum_{p=0}^{K-1} \sum_{q=0}^{K-1} y_{i+p,\, j+q} w_{pq} \qquad (3.1)$$

где y_{ij} – входное значение, w_{pq} – вес коэффициентов, z_{ij} – выходное значение.

Для уменьшения полосы пропускания данных согласно формуле 3.1, заменим порядок вычисления, как показано на рисунке 3.5. Двумерная свертка СНС с учетом рисунка 3.5 представлена на рисунке 3.6.

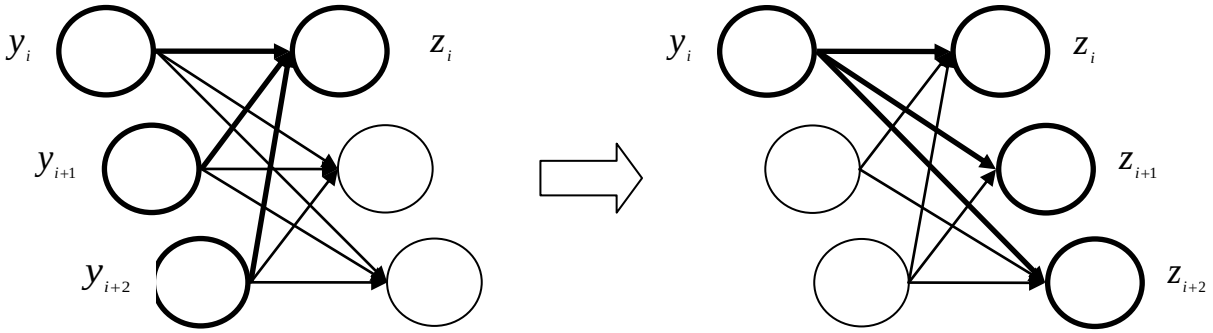


Рисунок 3.5. Замена порядка вычислений

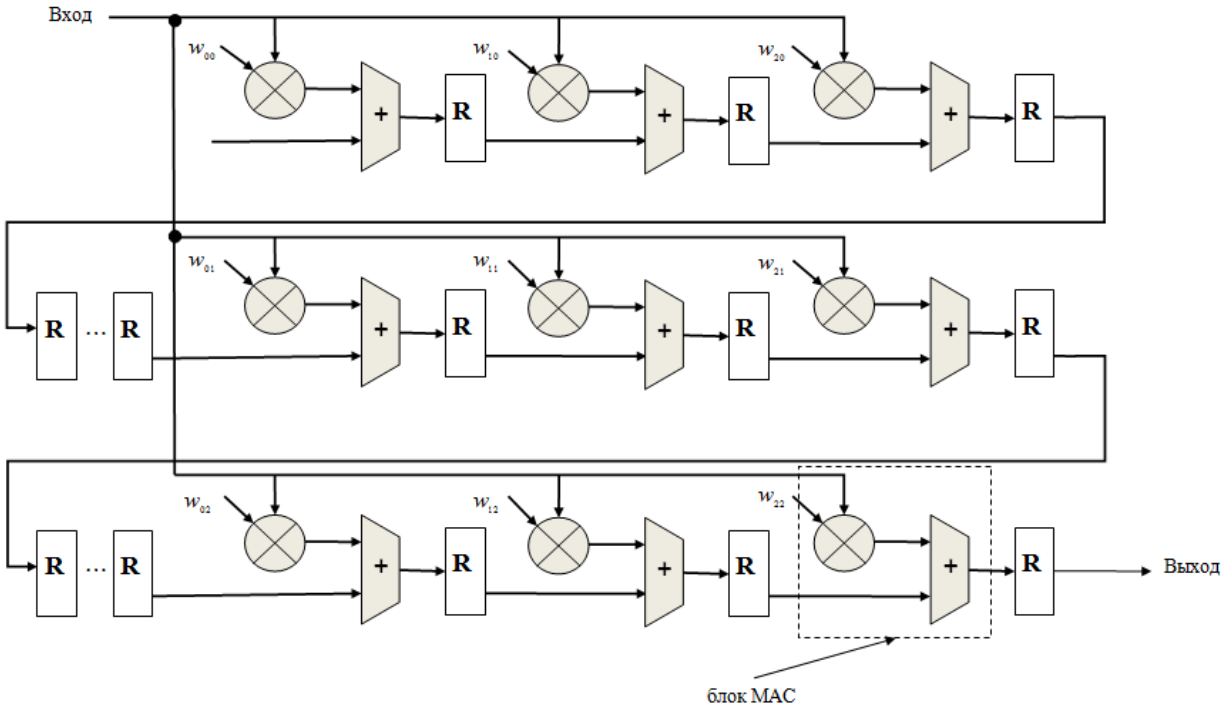


Рисунок 3.6. Процесс двумерной свертки СНС

Из таблицы 3.1, видно, что различные слои имеют разные размеры ядра (маски) K. В предложенной архитектуре мы используем мультиплексоры в устройстве двумерной свертки размером 3×3, показанном на рисунке 3.6.

Используем конфигурации бит для переключения размера ядра на 11×11, четыре параллельных ядра 5 × 5 или 16 параллельных ядра 3 × 3 (рисунок 3.7).

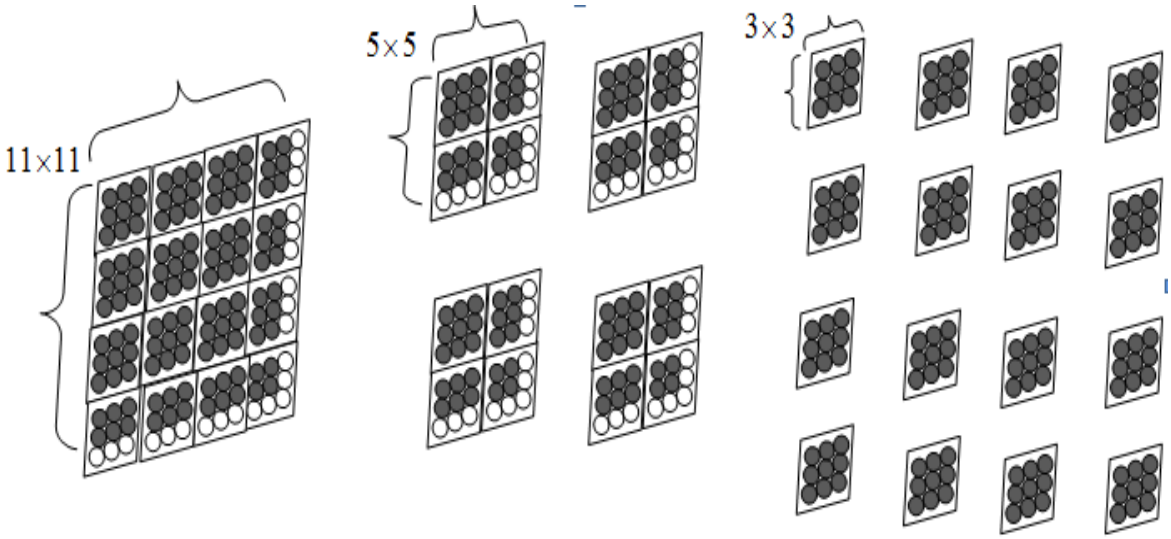


Рисунок 3.7. Реконфигурация ядра с размером 3 × 3 для схемы двумерной свертки

Рассмотрим представление 48-битной фиксированной точки, которая является самой высокой точностью обычной реализации. 48-битный размер представления данных, обусловлен тем, что 32-битное представление является в большинстве случаев недостаточным для процесса обработки данных, а 64-битное может приводить к увеличению времени выполнения арифметических операций. В этом случае, двумерная свертка работает с $48+48+\lceil \log_2 121 \rceil = 103$ битами представления [109]. После свертки выходной сигнал округляется до 48 бит. Так как для операции округления необходимы дополнительные блоки MAC (DSP48Es), то реализация для полностью параллельной схемы двумерной свертки является затруднительной. Уменьшить размер этого блока можно путем применения непозиционной системы счисления, а именно использовать вложенную систему остаточных классов.

Определим понятие вложенной системы остаточных классов [2]. Свойства попарно взаимно простых модулей СОК накладывает ограничения на их реализацию с помощью LUT-таблиц одинакового размера. В этом случае, предположим, что целое число Z представлено в СОК в виде $Z = (Z_2, Z_2, ..., Z_L)$, а

модуль Z_i из этого набора представлен рекурсивно. Такое представление называется Вложенной Системой Остаточных Классов, которая определяется набором модулей j вида $(m_{i1}, m_{i2}, ..., m_{ij})$, где нет пары модулей, которая имела бы общий множитель с любой другой. Во вложенной СОК Z_i может быть однозначно представлено в виде кортежа j целых чисел следующим образом:

$$Z = (Z_1, Z_2, ...(Z_{i1}, Z_{i2}, ..., Z_{ij})_i, ..., Z_I) \quad (3.2)$$

Далее вложенность будет применяться к крайнему правому модулю m_i. Поскольку вложенная СОК может применяться к каждому элементу в СОК рекурсивно, то поэтому модули могут после разложения иметь одинаковый размер. В этом случае особое внимание следует уделять размеру динамического диапазона.

Пусть A и B целые числа имеющие представление в СОК $A = (A_1, A_2, ..., A_L)$ и $B = (B_1, B_2, ..., B_L)$, соответственно. Применяя вложенную СОК к A_i и B_i получаем представление этих чисел в следующим виде $A = (A_1, A_2, ...(A_{i1}, A_{i2}, ...A_{ij})_{m_i}, ..., A_L)$ и $B = (B_1, B_2, ...(B_{i1}, B_{i2}, ..., B_{ij})_{m_i}, ...B_L)$. Поскольку динамический диапазон для вложенной СОК вычисляется по формуле $P_{p_i} = \prod_{l=1}^{j} p_{il}$ и при этом должно выполняться соотношение $P_{p_i} \geq A_i \circ B_i$, то в некоторых случаях применение вложенной СОК не целесообразно, так как ограничение, накладываемое на динамический диапазон, может привести к некорректной работе системы счисления.

Пример 3.2. Рассмотрим СОК $(3,4,5)$, представленную во вложенной СОК набором модулей вида $(3,4,(3,4)_5)$. Пусть $X = 17$ и $Y = 3$. Во вложенной СОК X и Y представлены как $X = (2,1,(2,2)_5)$ и $Y = (0,3,(0,3)_5)$. Таким образом сумма этих чисел вычисляется следующим образом:

$$X + Y = (2 \bmod 3, 4 \bmod 4, (2 \bmod 3, 5 \bmod 4)_5) = (2,0,(2,1)_5)$$

Преобразуя полученный результат вложенного модуля $(2,1)_5$ сначала обратно в модуль СОК $(3,4)_5$, а затем в двоичное число мы получаем результат равный 5. Отметим, что $5 \bmod 5 = 0$, тогда получим:

$$(2,0,(2,1)_5) = (2,0,5) = (2,0,0).$$

Преобразуя полученный результат $(2,0,0)$ в СОК $(3,4,5)$, а затем в двоичное число получаем 20, что равно результату суммы $17+3$.

Рассмотрим более подробно два варианта реализации преобразования из позиционной системы счисления в СОК, каждая из которых для преобразования данных будет использовать в своей архитектуре комбинации LUT-таблиц.

При реализации преобразователей кода LUT-таблицы содержат предварительно рассчитанные суммы произведений коэффициентов исходного числа с учетом их весов [59]. Такой подход помогает достичь высокой параллельности обработки данных, что в свою очередь экономит ресурсы вычислительной архитектуры.

Для реализации преобразований из ПСС в СОК требуется $2^n \sum_{i=1}^{L} \lceil \log_2 p_i \rceil$ бит памяти, что является достаточно большим объемом при большем n. В [109] авторы предлагают уменьшить общее количество памяти, применив LUT-таблицы, полученные по средством функционального разложение.

Рассмотрим более подробно метод построения LUT-таблиц на основе функциональной декомпозиции. Пусть задана функция $F(X)$: $B^n \to \{0,1,\ldots,m-1\}$, где $B = \{0,1\}$ и $X = (x_1, x_2, \ldots, x_n)$. (X_L, X_H) это разбиение X на две части. Тогда результат разложения F представляет собой двумерную матрицу, в которой каждый столбец является отдельным элементом из X_L, а каждая строка является отдельным элементом из X_H, что соответствует значению матрицы $F(X_L, X_H)$. При этом, X_L обозначает связанные переменные, а X_H – свободные переменные. Количество таких закономерностей для другого столбца в диаграмме разложения является столбцом кратности μ, $r = \lceil \log_2 \mu \rceil$ – это направляющие связи со

смежными LUT-таблицами [109]. Рисунок 3.8 иллюстрирует рассмотренную функциональную декомпозицию.

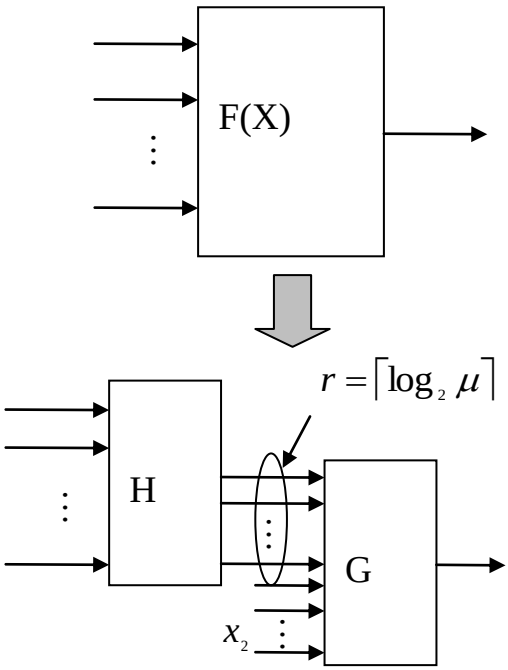


Рисунок 3.8. Функциональное разложение

Пример 3.3. Рассмотрим функциональное разложение числа, представленного во вложенной СОК (таблица 3.2), и соответствующий этому разложению, каскад из LUT-таблиц (рисунок 3.9). Предположим, что $X = (x_0, x_1, x_2, x_3, x_4)$, $X_L = (x_0, x_1, x_2)$ и $X_H = (x_3, x_4)$. Тогда из декомпозиции для $X \bmod 3$ видно, что кратность столбца μ равна 3, а количество отношений r это $\lceil \log_2 3 \rceil = 2$. Для реализации одного такого преобразования необходимо $2^5 \times 2 = 64$ бит памяти, при этом при преобразовании с использованием каскада LUT-таблиц требуется $2^3 \times 2 + 2^{2+2} \times 2 = 48$ бит памяти.

Архитектура преобразователя из ПСС в СОК с применением LUT-таблиц, полученных с помощью рассмотренного нами функционального разложения, представлена на рисунке 3.10 [109]. Из свойства модульных операций получаем, что для каждого модуля p_i столбец кратности не больше p_i. Тогда, если s это количество памяти (BRAM) для k входов, то для каскада состоящего из LUT-таблиц для модуля p_i мы получаем:

$$s = \left\lceil \frac{n-r}{k-r} \right\rceil = \left\lceil \frac{n-\lceil \log_2 p_i \rceil}{k-\lceil \log_2 p_i \rceil} \right\rceil \qquad (3.3)$$

Таблица 3.2. Пример разложения $X \bmod 3$.

		x_0	0	0	0	0	1	1	1	1
		x_1	0	0	1	1	0	0	1	1
x_3	x_4	x_2	0	1	0	1	0	1	0	1
0	0		0	1	2	0	1	2	0	1
0	1		1	2	0	1	2	0	1	2
1	0		2	0	1	2	0	1	2	0
1	1		0	1	2	0	1	2	0	1
	r		0	1	2	0	1	2	0	1

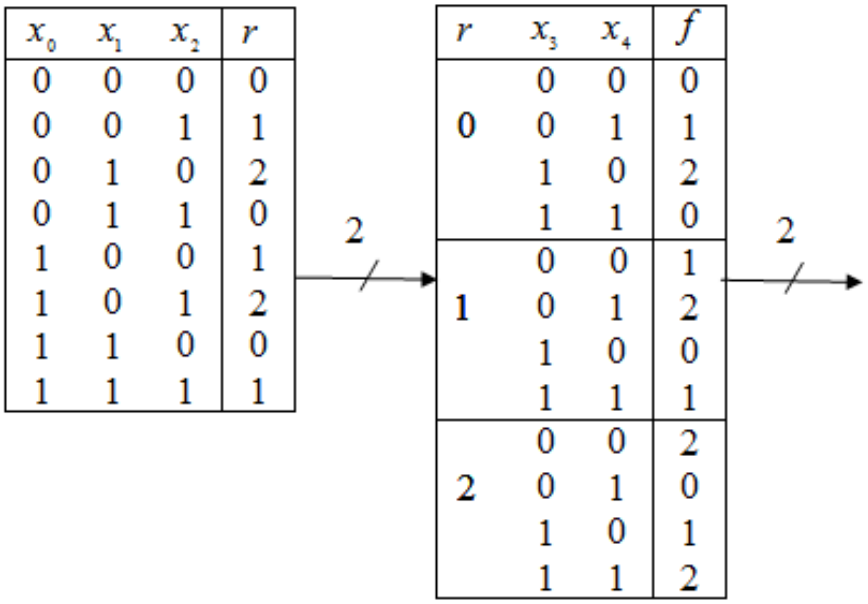


Рисунок 3.9. Каскад LUT-таблиц для разложения $X \bmod 3$.

Однако, предложенная архитектура преобразователя [109] обладает рядом недостатков:

1. Время необходимое для выполнения операций растет прямо пропорционально с увеличением количества модулей.

2. Архитектура зависима от разрядности данных, что также влияет на скорость обработки данных.

3. Общее количество памяти требуемой для вычислений увеличивается линейно с ростом диапазона системы.

Применение распределенной арифметики в такой архитектуре сможет не только увеличить скорость обработки данных, но и существенно сократить объем необходимой памяти.

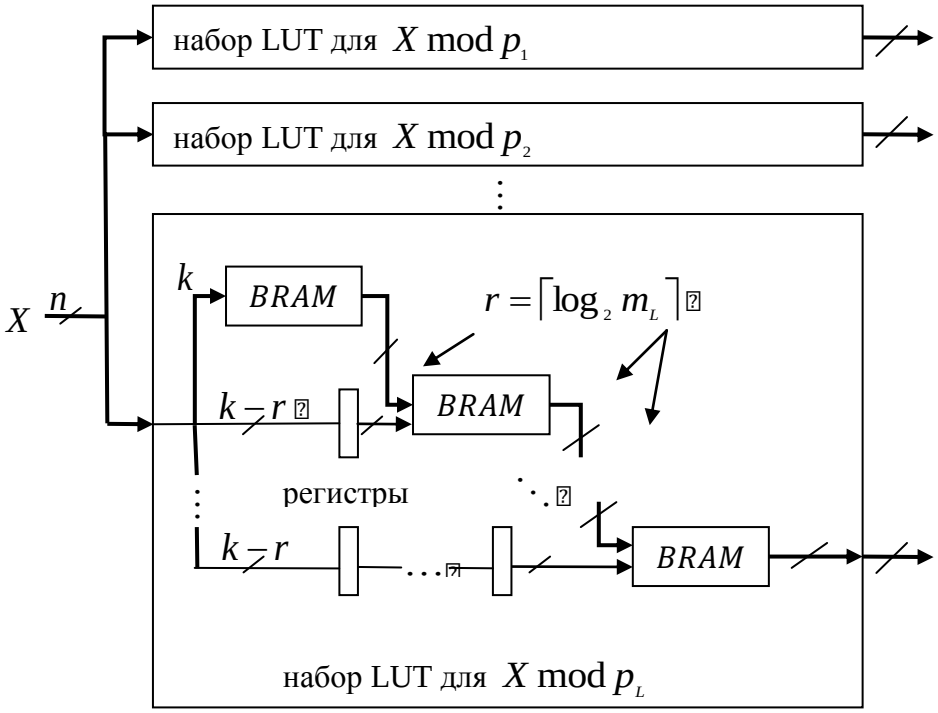


Рисунок 3.10. Преобразователь из ПСС в СОК с применением LUT-таблиц на основе функционального разложения.

Рассмотрим обратное преобразование из СОК в двоичную систему счисления. Для реализации такого преобразования будем использовать формулу вида (3.4):

$$X = x_i \left| P_i^{-1} \right|_{m_i} \cdot P_i, \tag{3.4}$$

где $P_i = P / p_i$, $P_i^{-1} \cdot P_i = 1$. Пример обратного преобразователя из СОК в ПСС, для СОК вида $(3,4,5)$ показан на рисунке 3.11.

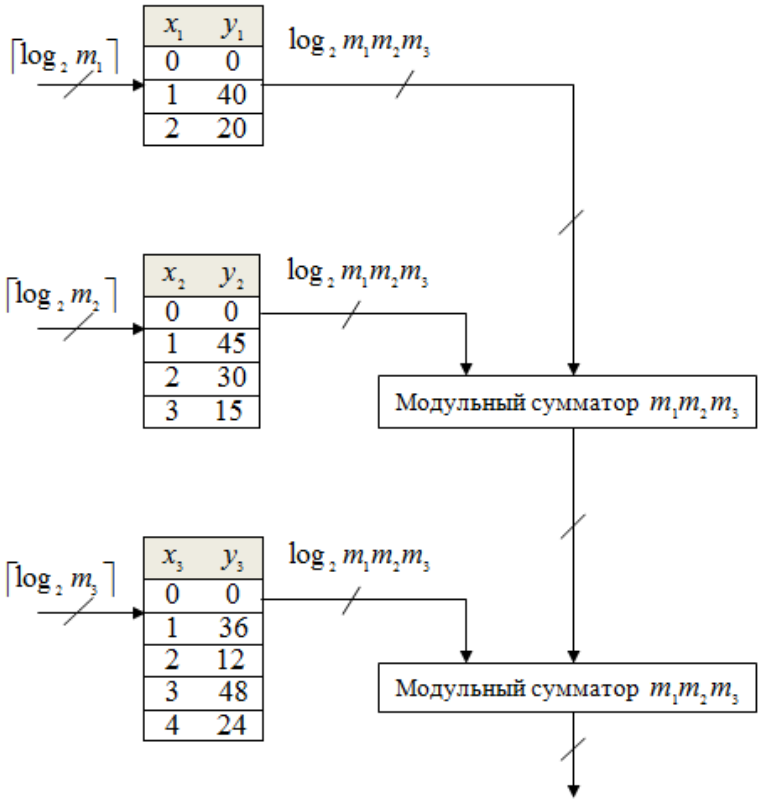


Рисунок 3.11. Пример обратного преобразователя из СОК в ПСС

Предположим, что целое число X представлено в СОК в виде $(X_1, X_2, ..., X_L)$. Для X_i память генерирует P_i ,что является произведением всех модулей, кроме p_i . Структура дерева сумматоров модулей X_i работает на высокой тактовой частоте. Как показано на рисунке 3.12, такой модульный сумматор может быть реализован с помощью двух двоичных сумматоров, логического элемента ИЛИ и мультиплексора [73], где j в этой схеме является целым числом, причем $2^{j-1} < p \le 2^j$. Следует отметить, что преобразование из СОК в ПСС может быть реализовано с помощью $O(L)$ блоков DSP48E и блока BRAM, поэтому, это не является существенной проблемой в дальнейших исследованиях.

Следствие 4.1. Пусть целое число Z представлено в СОК в виде $(Z_1, Z_2, ..., Z_L)$ и p_i – это i -модуль. Когда Z_i представлено во вложенной СОК $(Z_{i1}, Z_{i2}, ..., Z_{i,j})$, то каскад из LUT-таблиц при реализации преобразования из двоичной системы во вложенную СОК имеет более $\lceil \log_2 p_i \rceil$ отношений [109].

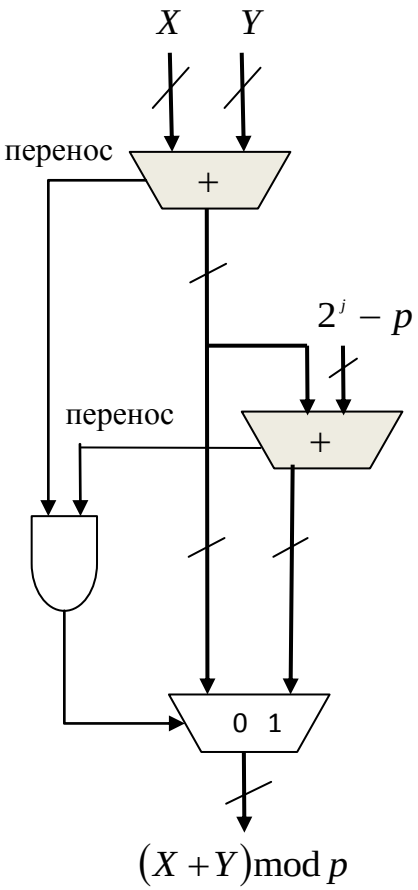


Рисунок 3.12. Модульный сумматор

Это следствие показывает, что объем памяти для блока BRAM увеличивается только для хранения целых чисел вложенной СОК, что в свою очередь пропорционально числу модулей L. С другой стороны, как показано на рисунке 3.10, преобразователь перевода из вложенной СОК в ПСС реализует дерево преобразователей $O(L)$ перевода из традиционной СОК в двоичную систему [73]. Реализуется это в блоке RAM и DSP48E.

Ранее было отмечено, что для работы двумерной свертки необходимо 103 бита. Выберем набор модулей СОК из небольших целых чисел, так чтобы они охватывали динамический диапазон 2^{103}. В этом случае мы имеем следующие модули СОК [109]:

$$(3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,59,61,67,71,73,79,83)$$

Чтобы уменьшить блоки MAC, применим вложенную СОК к 2 модулям, которые больше 15. Тогда мы имеем следующее множество модулей для вложенной СОК:

$$(3,5,7,11,13,(3,4,5,7,11,13)_{17},(3,4,5,7,11,13)_{19},(3,4,5,7,11,13,(3,4,5,7,11,13)_{17})_{23},$$
$$(3,4,5,7,11,13,(3,4,5,7,11,13)_{17})_{29},\ldots(3,4,5,7,11,13,(3,4,5,7,11,13)_{17})_{83}.$$

Вложенная СОК дает возможность разложить 48-битный блок MAC и получить в результате параллельные 4-битные блоки MAC. Таким образом, можно получить 8-входных и 4-выходных LUT-таблиц.

Двумерная перестраиваемая свертка размером 3×3 на основе алгебраических вычислений во вложенной СОК проиллюстрирована на рисунке 3.13. Предложенная схема раскладывает 48-битный MAC-блок в параллельные 4-битные, и реализует их с 8 входными и 4 выходными LUT-таблицами. Так как схема не генерирует передачу сигналов, то она работает на высокой тактовой частоте. Пусть t число нейронов в глубокой сверточной нейронной сети. Тогда, число весов w_{ij} увеличивается по $O(2^t)$. Таким образом, большой размер нейронной сети не может хранить веса в блоке памяти RAM [87], по этой причине их хранение происходит во внешнем блоке памяти DDR3SODIMM. Так как обновление весов происходит не так часто, то имеется достаточно времени, чтобы считать их из DDR3SODIMM.

Таким образом, результатом рассмотрения методов и способов проектирования сверточной сети [109] является разработанная архитектура глубокой нейронной сети на основе вложенной СОК, представленная на рисунке 3.14. Разработанная архитектура считывает входные изображения, веса и конфигурации бит из блока внешней памяти DDR3SODIMM. Также она преобразует двоичные сигналы во вложенную СОК; выполняет двумерную свертку, которая имеет соответствующий размер ядра; преобразует сигналы вложенной СОК в диапазоне 103 бит с использованием бинарного преобразователя и, наконец, округляет вывод сигнала до 48-бит. Внешняя память

блока DDR3SODIMM используется для хранения временных данных для разных слоев.

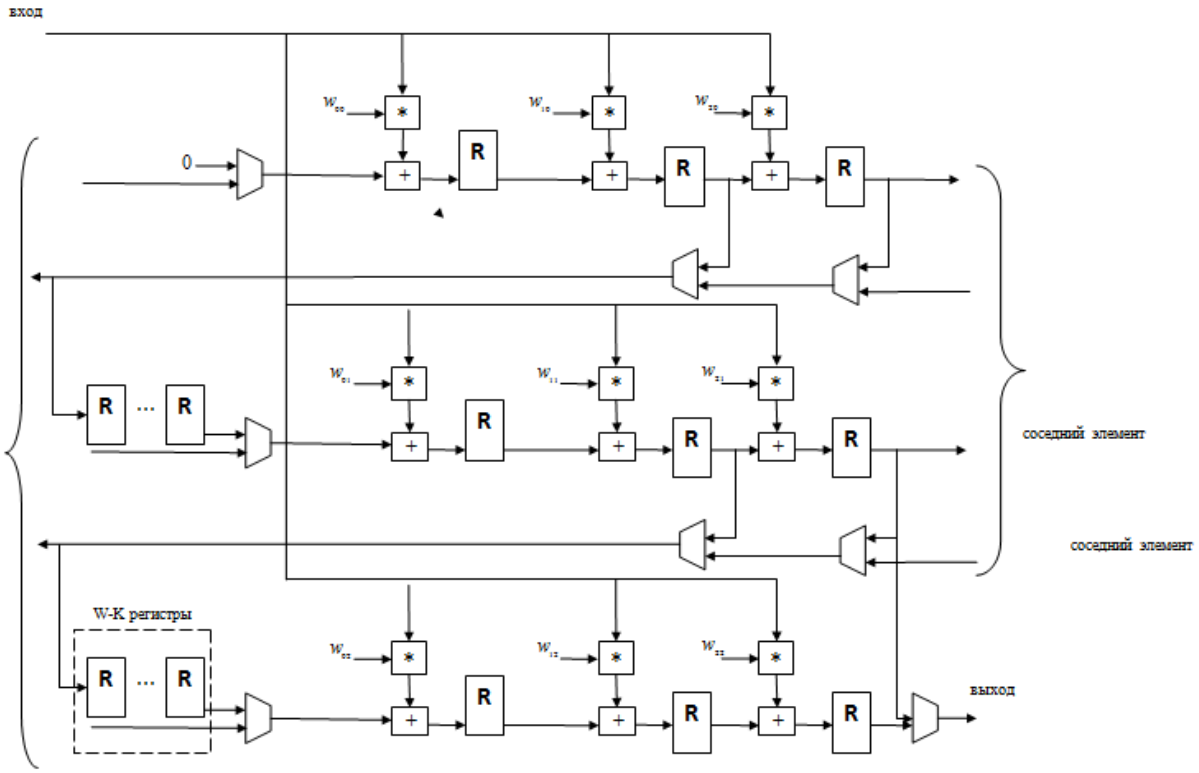


Рисунок 3.13. Двумерная свертка с ядром размера 3×3.

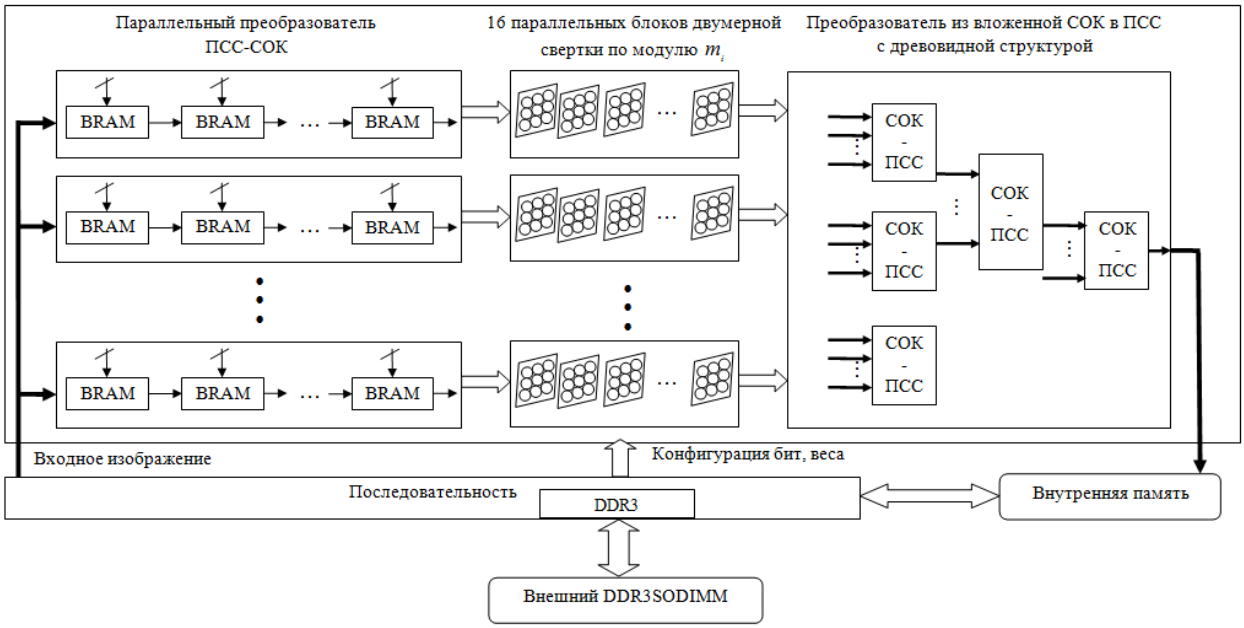


Рисунок 3.14. Глубокая сверточная нейронная сеть с вычислениями во вложенной СОК

Однако, предлагаемая в [109] архитектура сверточной нейронной сети обладает рядом недостатков, которые влияют на скорость и эффективность ее работы.

1. Сеть имеет большую логическую глубину, обусловленную использованием последовательности вложенных СОК, что приводит к существенному усложнению операции прямого и обратного преобразования.

2. Для вычислений в рассмотренной архитектуре СНС используется большой динамический диапазон, возникающий по причине использования 48-битных весовых коэффициентов.

Устранить эти недостатки возможно с применением в архитектуре СНС преобразователя на основе распределенной арифметики, а также модулей специального вида СОК.

3.2 Разработка архитектуры глубокой сверточной нейронной сети на основе использования распределенной арифметики в преобразователе ПСС-СОК

Рассмотрим более эффективный метод преобразования двоичного числа в систему остаточных классов на основе распределенной арифметики [59,67]. Заключается предлагаемый метод в том, что вначале исходное число разбивается на отдельные части с заранее известным количеством двоичных разрядов B. Тогда n-битное двоичное число можно представить в виде форматов, каждый из которых является позиционной комбинацией $\frac{n}{B}$. При этом каждому такому формату присваивается свой вес 2^j, где $j = 0, B, 2B, \ldots, MB$. На следующем этапе преобразования из двоичной записи числа в СОК происходит модульное суммирование остатков по модулю $p_i (i = 1,2,\ldots,n)$ каждого разряда $\frac{n}{B}$ формата с учетом их весов.

Тогда любое двоичное число можно представить в виде [59,106]:

$$X = \sum_{j=0}^{M} \left(\sum_{i=0}^{B-1} x_i 2^i \right) 2^j \qquad (3.5)$$

где B – количество разрядов выбранного формата, M – степень формата, x_i – коэффициент 0 или 1, $j = 0, B, 2B, \ldots, MB$ – позиция формата, j – позиция разряда в формате. С учетом этого, формула (3.5) можно переписать в виде:

$$\begin{aligned} X = &\left(x_0 2^0 + x_1 2^1 + \ldots + x_{B-1} 2^{b-1}\right) 2^0 + \left(x_0 2^0 + x_1 2^1 + \ldots + x_{B-1} 2^{B-1}\right) 2^B + \cdots \\ &+ \left(x_0 2^0 + x_1 2^1 + \ldots + x_{B-1} 2^{B-1}\right) 2^{MB} \end{aligned} \qquad (3.6)$$

Тогда 32-разрядное входное число с форматом равным 8 бит на основании формулы перепишется в следующем виде:

$$x[n] = x[n;31\ldots24]2^{24} + x[n;23\ldots16]2^{16} + x[n;15\ldots8]2^{8} + x[n;7\ldots0]2^{0} \qquad (3.7)$$

Остаток по модулю будет записан как:

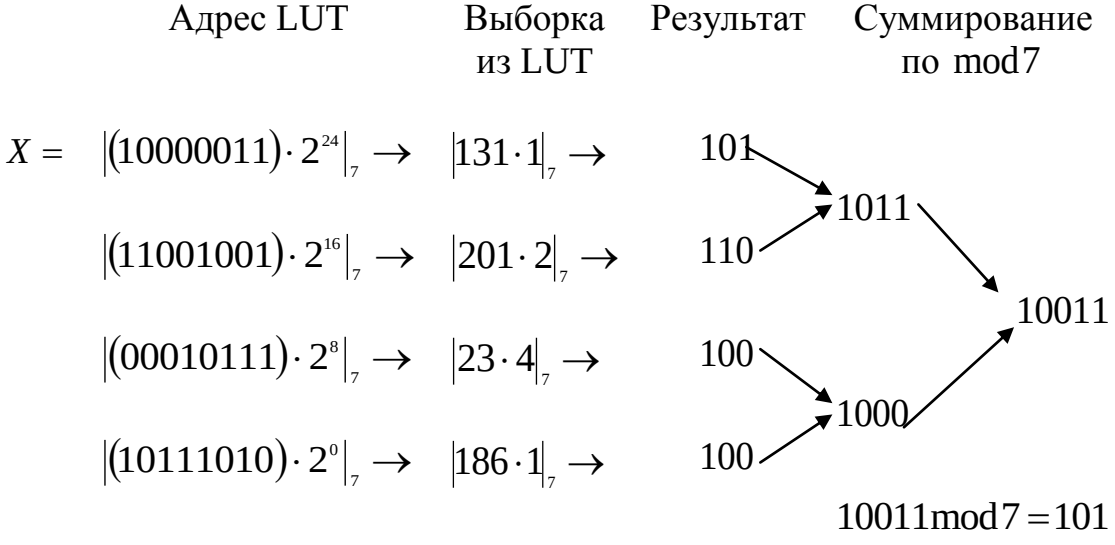
$$X_i[n] = \left| x[n;31\ldots24]2^{24} \right|_{p_i} + \left| x[n;23\ldots16]2^{16} \right|_{p_i} + \left| x[n;15\ldots8]2^{8} \right|_{p_i} + \left| x[n;7\ldots0]2^{0} \right|_{p_i} \Big|_{p_i} \qquad (3.8)$$

Пример 3.4. Пусть дано число X =10000011 11001001 00010111 10111010. Необходимо найти остаток числа X по модулю 7.

Разобьем двоичное представление числа X на 4 формата по 8 бит в каждом. Для дальнейших вычислений, найдем константы $\omega_k = N^i \bmod 7$, где $k = 0..3$, $i = 0, 8, 16, 24$:

$$\omega_0 = \left|2^0\right|_7 = 1;\ \omega_1 = \left|2^8\right|_7 = 4;\ \omega_2 = \left|2^{16}\right|_7 = 2;\ \omega_3 = \left|2^{24}\right|_7 = 1.$$

Найденные константы запишем в соответствующую LUT-таблицу. Тогда, дальнейшие вычисления будут представлены в виде следующей схемы [59]:



Согласно данным вычислениям, $\left|X\right|_7 = 101$, что соответствует действительности [2].

Из рассмотренного примера видно, что данный способ вычисления остатка от деления хорошо подходит для работы с числами большой разрядности. В данном случае за счет параллельной обработки данных в несколько раз сокращается разрядность операндов математических преобразований. Такой подход позволяет ускорить вычислительные операции и, кроме того, упрощает представление чисел большой разрядности за счет использования более короткой разрядной сетки.

Структурная схема преобразователя чисел из ПСС в СОК, на основе выражения (3.8), приведена на рисунке 3.15. Она состоит из просмотровых таблиц LUT 1 – LUT 4, двух блоков сумматоров EAC-CSA (End Around Carry - Carry Save Adder) и одного блока EAC-KSA (End Around Carry - Kogge-Stone Adder) [59]. Использование блоков EAC-CSA и EAC-KSA позволяет существенно сократить временные затраты в ходе арифметических операций, за счет параллельной архитектуры [124].

Таким образом, рассмотренную схему преобразователя нейронной сети (рисунок 3.15) можно характеризовать как схему с повышенной скоростью прямого преобразования двоичных чисел в модулярную форму [59,67].

Развитием параллельного преобразователя из ПСС в СОК является схема параллельно-последовательного конвейерного преобразователя (рисунок 3.16) [67]. Отличительная особенность преобразователя, который включает в себя прохождения четырех стадий обработки, заключается в том, что все они могут работать параллельно. Преобразователь состоит из блока мультиплексор на 4 группы входа размером 1 бит каждый и одним адресным входом AB1, $\lceil \log p_i \rceil$ – LUT-таблиц, трех умножителей по модулю p_i , демультиплексора DMX с адресным входом AB2, осуществляющего подключение выхода LUT-таблицы на один из четырех выходов и сумматора по модулю p_i .

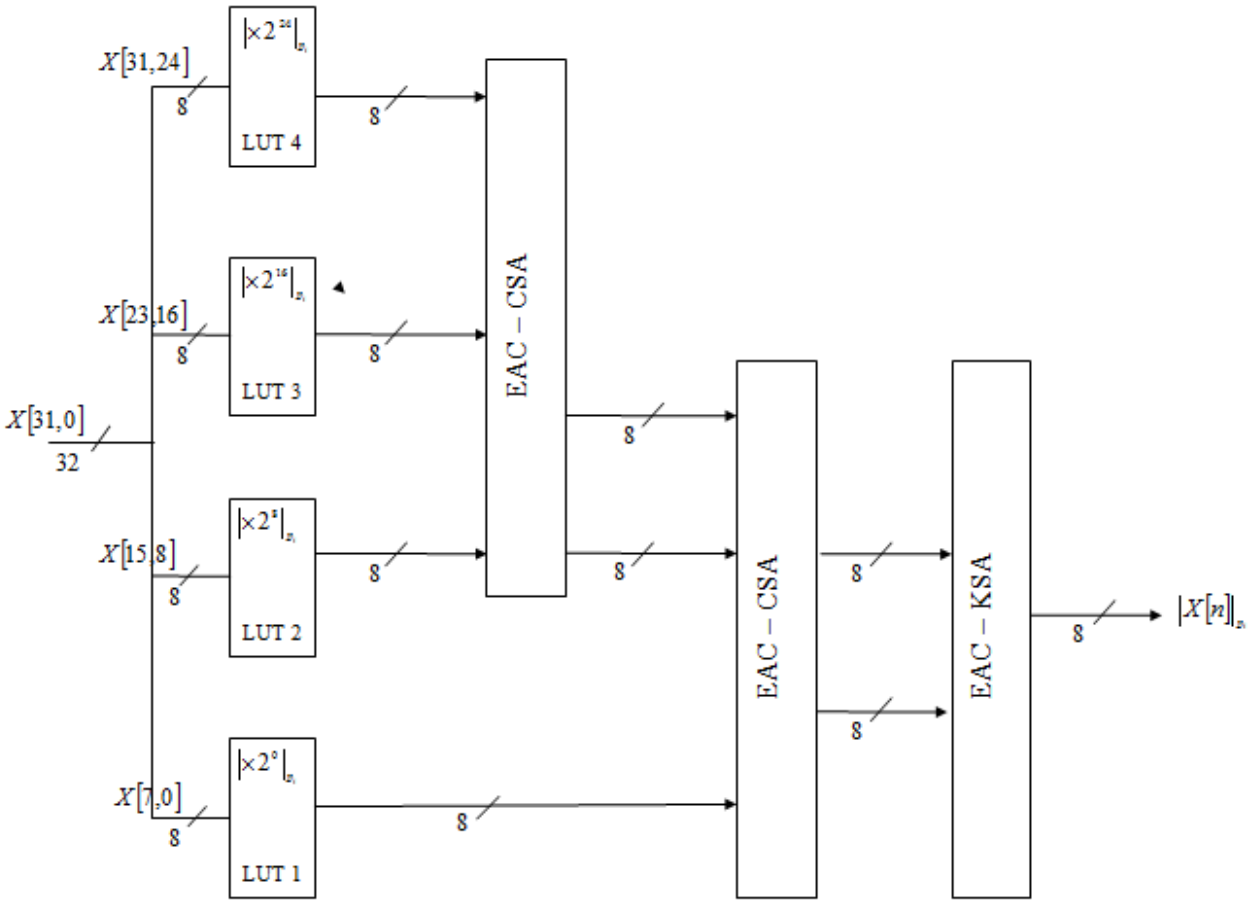


Рисунок 3.15. Преобразователь из ПСС в СОК с применением LUT-таблиц на основе алгоритма распределенной арифметики.

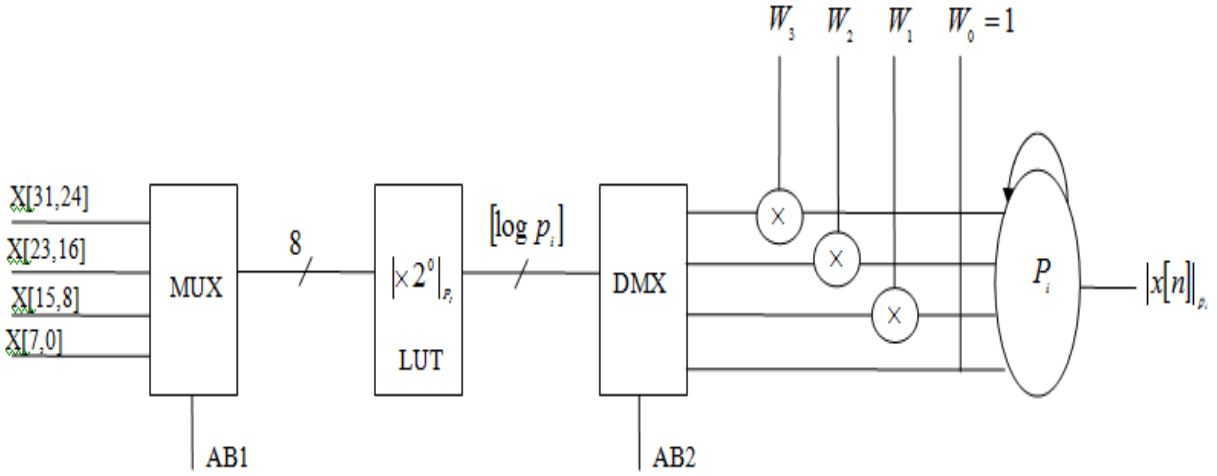


Рисунок 3.16. Структурная схема последовательно-параллельной конвейерной нейронной сети конечного кольца [59]

Первая стадия C_1 преобразователя состоит в том, что вначале обработки на адресные входы AB1 поступает код числа, который вызывает старший байт числа $x[31,24]$ и помещает его на выход мультиплексора. Выход мультиплексора является входом LUT-таблицы. После чего на стадии C_2 происходит считывание этого байта, но уже в виде вычета (остатка) по модулю p_i. Далее на стадии C_3 происходит умножение полученного остатка $\left|\left(\sum_{c=0}^{B-1} x_i 2^i\right)2^0\right|_{p_i}$ на значение соответствующего синаптического веса $w_j = \left|2^{ji}\right|_{p_i}$, где j – степень байта. Коммутация байтов осуществляется DMX под действием адресного входа AB2. На заключительной стадии C_4 выполняется суммирование остатков всех байтов по $\bmod p_i$ [67]. Рассмотрим пример иллюстрирующий работу данной схемы.

Пример 3.5. Пусть дано число X =10000011 11001001 00010111 10111010. Необходимо найти остаток числа X по модулю 7.

Для дальнейших вычислений воспользуемся найденными константами ω_i из примере 3.4:

$$\omega_0 = \left|2^0\right|_7 = 1;\ \omega_1 = \left|2^8\right|_7 = 4;\ \omega_2 = \left|2^{16}\right|_7 = 2;\ \omega_3 = \left|2^{24}\right|_7 = 1.$$

Тогда, дальнейшие вычисления будут представлены в виде следующей схемы:

	Адрес LUT	Выборка из LUT	Умножение на ω_i	Суммирование по $\bmod 7$
$X =$	$\left\|(10000011)\right\|_7 \rightarrow$	$101\rightarrow$	001	101
	$\left\|(11001001)\right\|_7 \rightarrow$	$101\rightarrow$	100	10100
	$\left\|(01000101)\right\|_7 \rightarrow$	$010\rightarrow$	010	100
	$\left\|(10111111)\right\|_7 \rightarrow$	$100\rightarrow$	001	100
				$100001 \bmod 7 = 101$

Полученный результат равен результату из примера 4.2. и соответствует действительности.

Таким образом, работа рассмотренной схемы характеризуется повышенной скоростью работы конвейерного преобразователя. Если предположить, что один цикл в этом преобразователе работает 5 нс, то общее время работы конвейера, необходимое для прохода через него одного байта будет равно 20 нс. Из чего также следует, что за время работы такой конвейер может завершить преобразование 200 млн. байт в секунду. При такой скорости работы, выигрыш по сравнению с последовательно-параллельной конвейерной нейронной сетью равен 4 [67].

Разработанная архитектура глубокой СНС с применением полученного преобразователя ПСС-СОК из рисунка 3.15 представлена на рисунке 3.17. Предлагаемая архитектура считывает входные изображения, веса и конфигурации бит из блока внешней памяти. Преобразует двоичные сигналы во вложенную СОК; выполняет двумерную свертку, которая имеет соответствующий размер ядра; преобразует сигналы вложенной СОК в диапазоне 63 бит с использованием бинарного преобразователя и, наконец, округляет вывод также до 63-бит. Внешняя память используется для хранения временных данных для разных слоев.

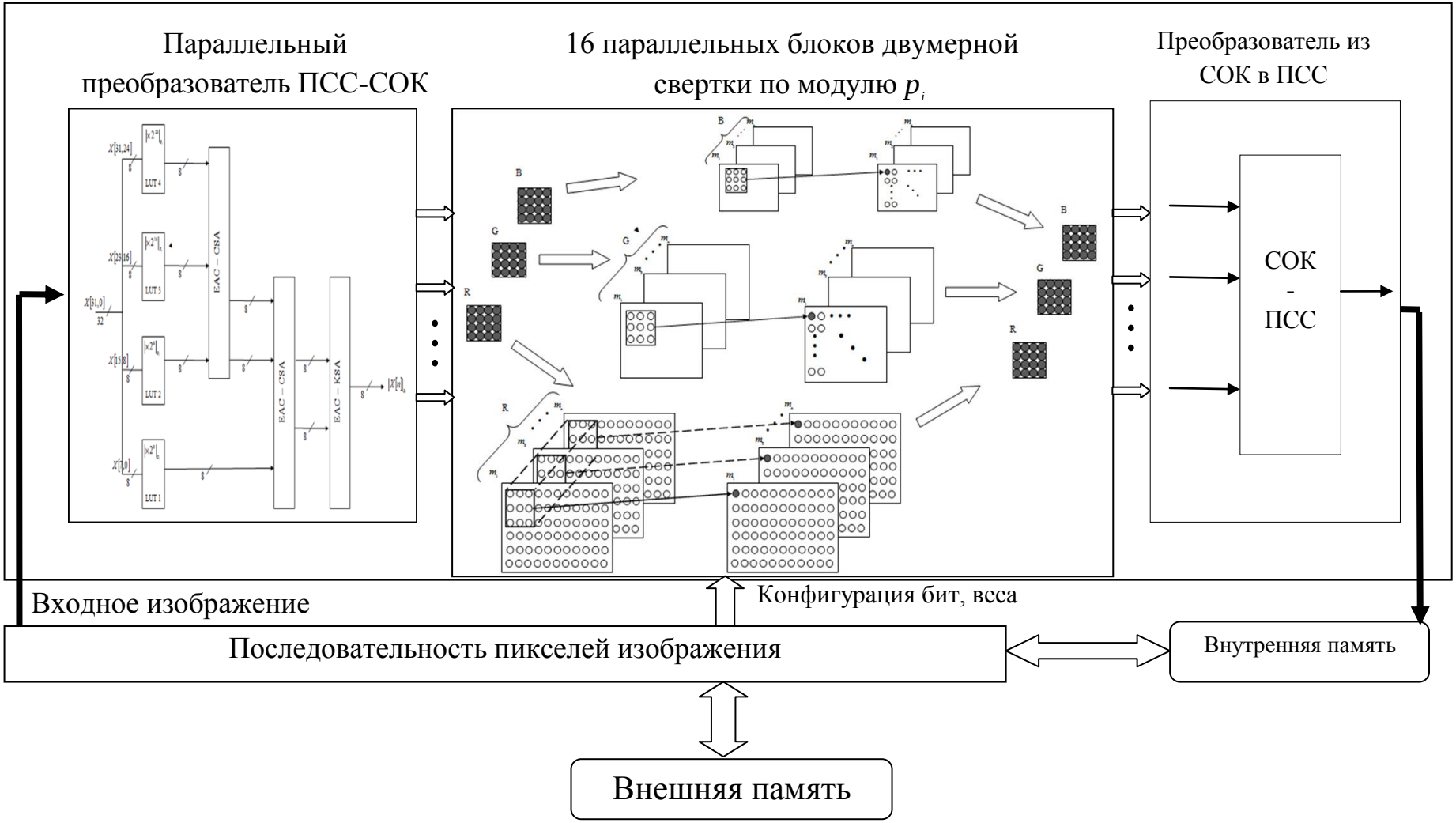


Рисунок 3.17. Глубокая сверточная нейронная сеть с вычислениями в СОК

3.3 Разработка архитектуры глубокой сверточной нейронной сети с вычислениями в системе остаточных классов с модулями специального вида

Рассмотрим с учетом определений системы остаточных классов основные арифметические устройства для модулей специального вида.

Сумматоры с сохранением переноса преобразуют операцию сложения трех чисел $A=\overline{A_{n-1}A_{n-2}\dots A_0}$, $B=\overline{B_{n-1}B_{n-2}\dots B_0}$ и $D=\overline{D_{n-1}D_{n-2}\dots D_0}$ к сложению чисел $S=\overline{S_nS_{n-1}\dots S_0}$ и $C=\overline{C_nC_{n-1}\dots C_0}$, где S_i - сумма трех слагаемых, а C представляет собой совокупность битов переноса. На рисунке 3.17а представлена схема работы данного сумматора, реализуемая алгоритмом 1. В полном сумматоре на каждом шаге цикла происходит вычисление бит суммы S_i и бит переноса C_{out} (рисунок 3.17б).

Алгоритм 1. Сумматор с сохранением переноса

Входные данные:	$A=\overline{A_{n-1}A_{n-2}\dots A_0}$, $B=\overline{B_{n-1}B_{n-2}\dots B_0}$, $D=\overline{D_{n-1}D_{n-2}\dots D_0}$

Вычисления:	for $i=\overline{0,n-1}$ do
	$S_i=A_i\oplus B_i\oplus D_i$;
	$C_i=(A_i\wedge B_i)\vee(A_i\wedge D_i)\vee(B_i\wedge D_i)$;
	end for;

Выходные данные:	$S=\overline{S_nS_{n-1}\dots S_0}$, $C=\overline{C_nC_{n-1}\dots C_0}$

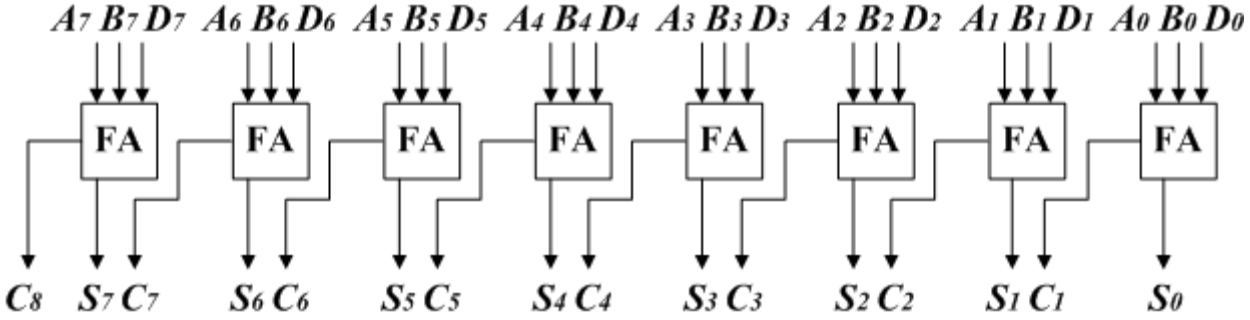
В сумматоре с сохранением переноса вычисления по каждому биту происходят параллельно, поэтому он обладает высокой скоростью.

Рисунок 3.18 показывает схему параллельного сложения, состоящую из трех стадий. На первой стадии осуществляется предварительное вычисление битов G_i, генерирующих перенос, битов P_i, передающих перенос, и полусумм H_i, для любого i, $0\le i\le n-1$:

$$G_i=A_i\wedge B_i,\ P_i=A_i\vee B_i,\ H_i=A_i\oplus B_i \qquad (3.9)$$

Вторая стадия сложения, называемая далее блоком вычисления переноса, вычисляет сигналы переноса C_i, для $0 \le i \le n-1$, используя G_i и P_i. На третьей стадии вычисляется сумма

$$S_i = H_i \oplus C_{i-1}. \tag{3.10}$$



а)

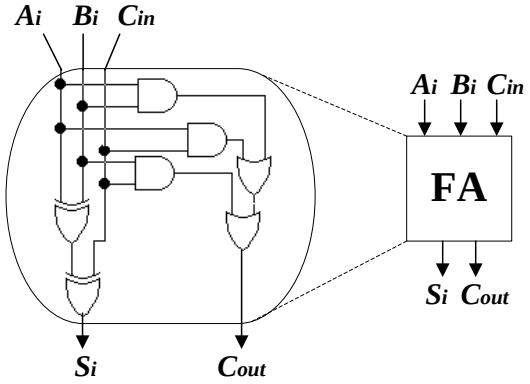


Рисунок 3.17. а) структура 8-битного сумматора с сохранением переноса [11]; б) устройство полного сумматора.

Блок вычисления переноса преобразуется в параллельно-префиксную форму с помощью оператора $\circ$, который связывает пары генерирующих и передающих бит и определен как

$$(G,P)\circ(G',P') = (G \vee (P \wedge G'), P \wedge P'). \tag{3.11}$$

Для последовательного вычисления пар генерирующих и передающих бит (G,P) введем обозначение $(G_{k:j}, P_{k:j})$, $k > j$, где соответствующая пара вычислена на основе бит $k, k-1, \ldots, j$ следующим образом:

$$(G_{k:j}, P_{k:j}) = (G_k, P_k)\circ(G_{k-1}, P_{k-1})\circ\ldots\circ(G_j, P_j). \tag{3.12}$$

Так как перенос $C_i = G_{i:0}$ для всех $i > 0$, то все переносы могут быть вычислены с использованием только оператора $\circ$ [124].

На рисунке 3.18 представлена схема 8-битного параллельно-префиксного сумматора, а на рисунке 3.19 - реализация его логических уровней.

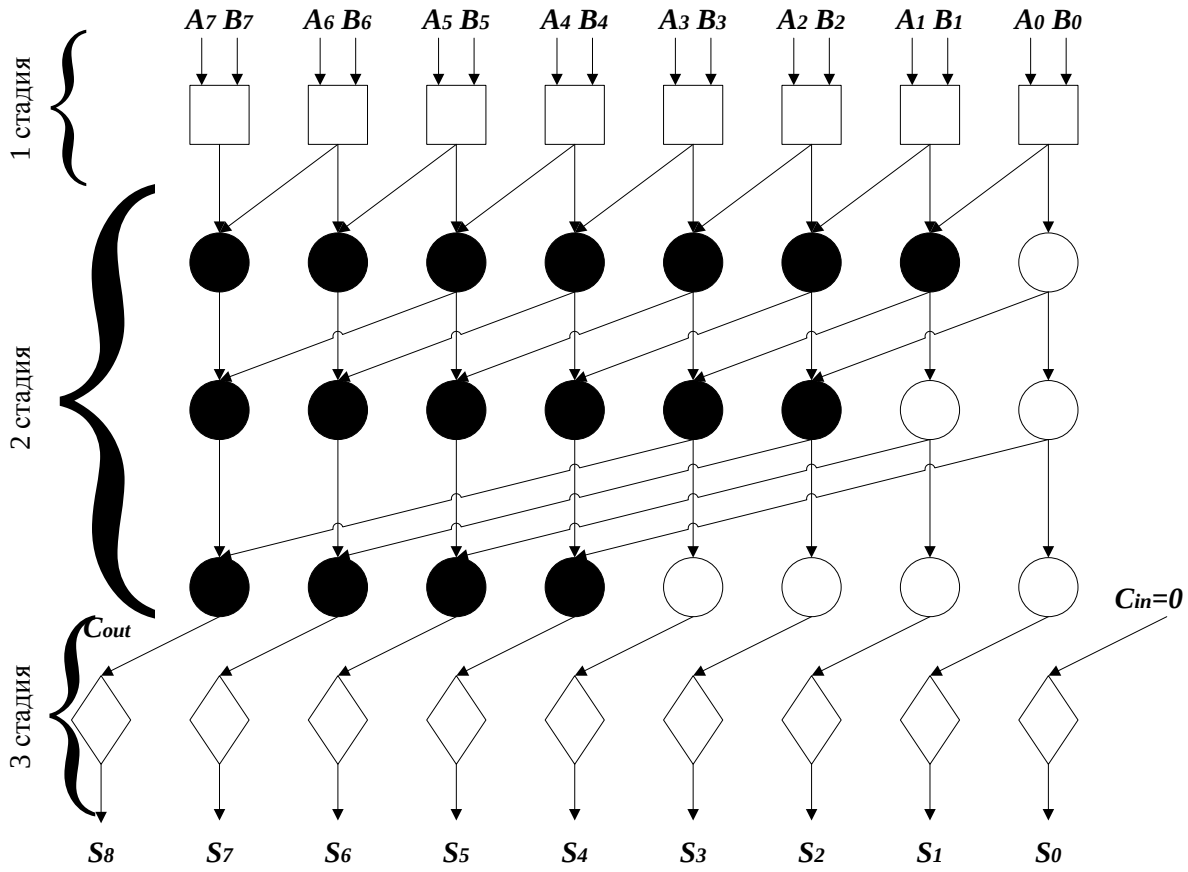


Рисунок 3.18. Структура 8-битного параллельно-префиксного сумматора Когге-Стоуна для целочисленного сложения [124].

Достоинством параллельно-префиксного сумматора является скорость его работы, так как вычисление всех разрядов происходит одновременно. К недостаткам такого сумматора относится использование большого числа логических элементов, что увеличивает площадь устройства.

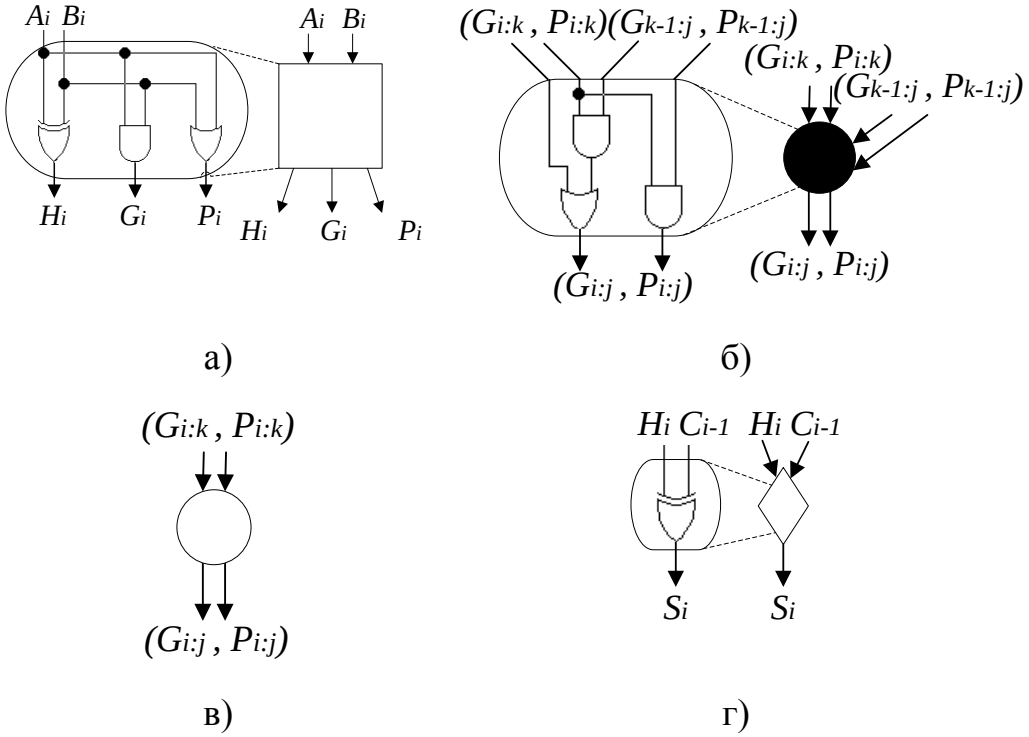


Рисунок 3.19. Устройство блоков сложения параллельно-префиксного сумматора: а) блоки первой стадии; б), в) блоки второй стадии; г) блоки третьей стадии [124].

Рассмотрим специальный набор модулей вида $\{2^n, 2^n-1, 2^{n+k}-1\}$. Для перевода числа в СОК необходимо вычислить остатки от деления по каждому из модулей.

Операция вычисления остатка от деления по модулю 2^n осуществляется простой обрезкой n младших бит исходного числа.

Для модуля 2^n-1 вычисление остатка от деления происходит сложнее. Пусть $X=\overline{X_{k-1}X_{k-2}\ldots X_0}$ – исходное число, разобьем его на $m=\lceil k/n \rceil$ чисел размерности n бит. Для этого дополним X справа 0 до размерности $k'=m\cdot n$, теперь $X'=\overline{X_{k'-1}X_{k'-2}\ldots X_0}$. Тогда $Y_0=\overline{X_{n-1},..,X_1,X_0}$, $Y_1=\overline{X_{2n-1},..,X_{n+1},X_n}$, ..., $Y_m=\overline{X_{k'-1},..,X_{(m-1)n+1},X_{(m-1)n}}$. Представим число X' как $X'=Y_0+Y_1\cdot 2^n+Y_2\cdot 2^{2n}+\ldots+Y_m\cdot 2^{mn}$, тогда

$$|X'|_{2^n-1} = |Y_0 + Y_1 \cdot 2^n + Y_2 \cdot 2^{2n} + \ldots + Y_m \cdot 2^{mn}|_{2^n-1} =$$
$$= \left\| Y_0 \right|_{2^n-1} + \left| Y_1 \cdot 2^n \right|_{2^n-1} + \left| Y_2 \cdot 2^{2n} \right|_{2^n-1} + \ldots + \left| Y_m \cdot 2^{mn} \right|_{2^n-1} \Big|_{2^n-1} = \left\| Y_0 \right|_{2^n-1} +$$
$$+ \left| Y_1 \cdot 2^n + Y_1 - Y_1 \right|_{2^n-1} + \left| Y_2 \cdot 2^{2n} + Y_2 - Y_2 \right|_{2^n-1} + \ldots + \left| Y_m \cdot 2^{mn} + Y_m - Y_m \right|_{2^n-1} \Big|_{2^n-1} =$$
$$= \left\| Y_0 \right|_{2^n-1} + \left| Y_1 \cdot \left(2^n - 1\right) + Y_1 \right|_{2^n-1} + \left| Y_2 \cdot \left(2^{2n} - 1\right) + Y_2 \right|_{2^n-1} + \ldots +$$
$$+ \left| Y_m \cdot \left(2^{mn} - 1\right) + Y_m \right|_{2^n-1} \Big|_{2^n-1} = \left\| Y_0 \right|_{2^n-1} + \left| Y_1 \right|_{2^n-1} + \left| Y_2 \right|_{2^n-1} + \ldots + \left| Y_m \right|_{2^n-1} \Big|_{2^n-1} =$$
$$= |Y_0 + Y_1 + Y_2 + \ldots + Y_m|_{2^n-1}$$

Таким образом

$$|X'|_{2^n-1} = |Y_0 + Y_1 + Y_2 + \ldots + Y_m|_{2^n-1} \qquad (3.13)$$

То есть вычисление остатка от деления по модулю $2^n - 1$ сводится к сложению n-битных чисел по модулю $2^n - 1$. Схема вычисления остатка от деления по модулю $2^n - 1$ для 16-битного числа и $n = 4$ представлена на рисунке 3.20. Для сложения по модулю $2^n - 1$ используются сумматоры с циклическим переносом (рисунки 3.21, 3.22).

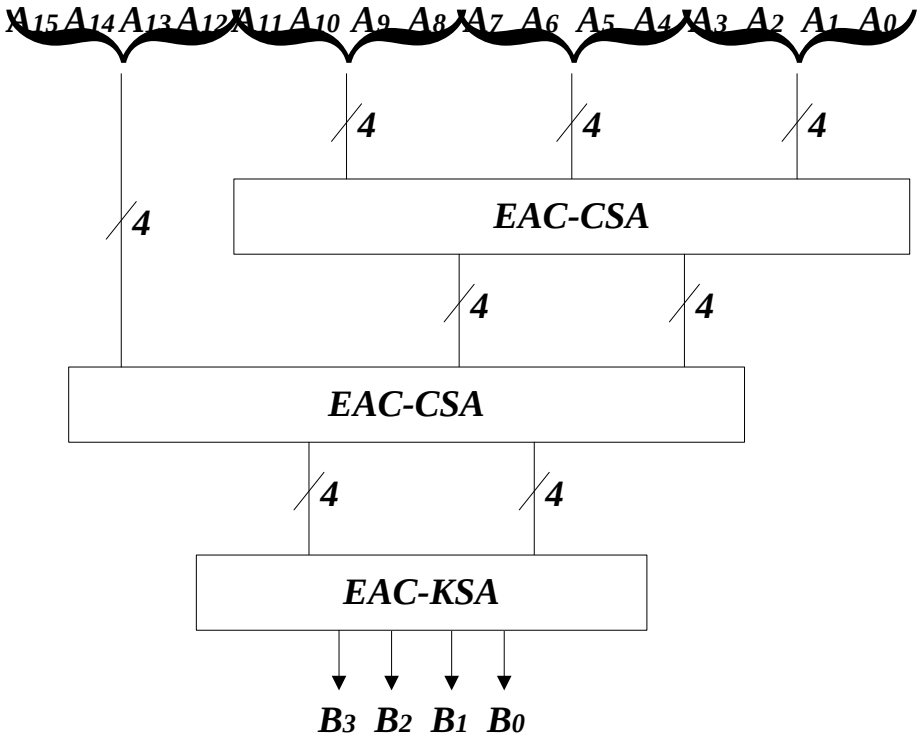


Рисунок 3.20. Схема вычисления остатка от деления по модулю $2^n - 1$ для 16-битного числа и $n = 4$.

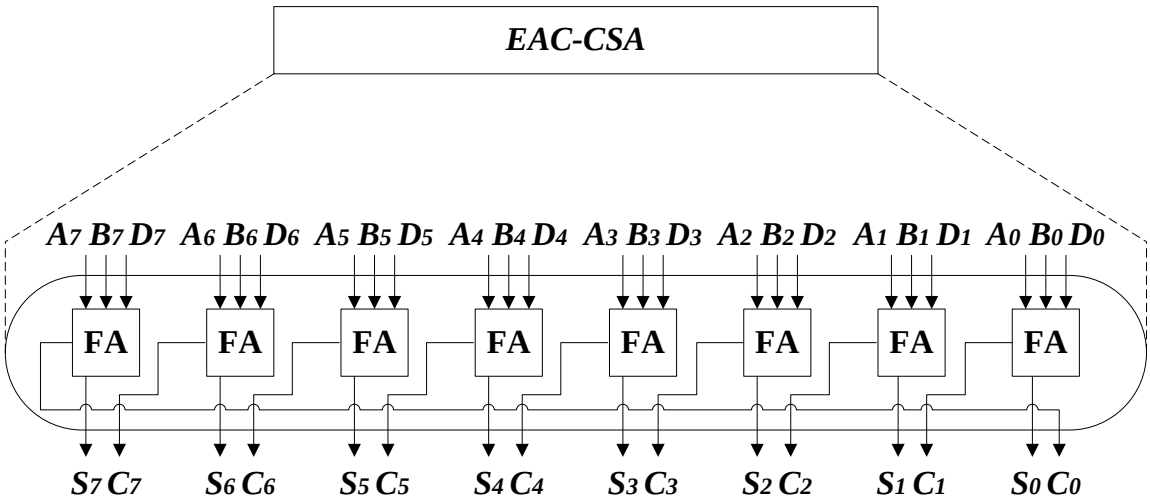


Рисунок 3.21. Структура 8-битного сумматора с сохранением переноса и циклическим переносом.

Наиболее проблемными операциями в СОК для СНС являются прямое и обратное преобразование. Для повышения эффективности этих операций мы предлагаем использовать модули вида 2^n и 2^{n-1}.

В пункте 3.1 была рассмотрено 48-битное представление данных, обеспечивающее высокую точность обработки изображений в практических приложениях. В этом случае, двумерная свертка СНС оперирует с представлением в 103 бит.

Для аппаратной реализации 103-битного диапазона в СОК в этом пункте будет рассмотрен набор модулей $\{2^7-1,2^{11}-1,2^{13}-1,2^{15}-1,2^{16}-1,2^{17}-1,2^{25}\}$ вместо набора модулей $\{3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,59,61,67,71,73\}$. Предложенный набор модулей СОК позволяет использовать параллельные арифметические устройства, изображенные на рисунках 5-7 для достижения высокой скорости обработки данных, представленных в модулярной форме.

Для оценки технических характеристик предложены архитектуры СНС, использующей вычисления в СОК было проведено моделирование на FPGA Artix7 XC7A200T в САПР Xilinx ISE Design Suite 14.7. Для моделирования использовались параметры синтеза, представленные в таблице 3.3. Целью моделирования было сравнение модулярных сумматоров и устройств прямого и

обратного преобразования в предложенной архитектуре и в известной СНС из [109].

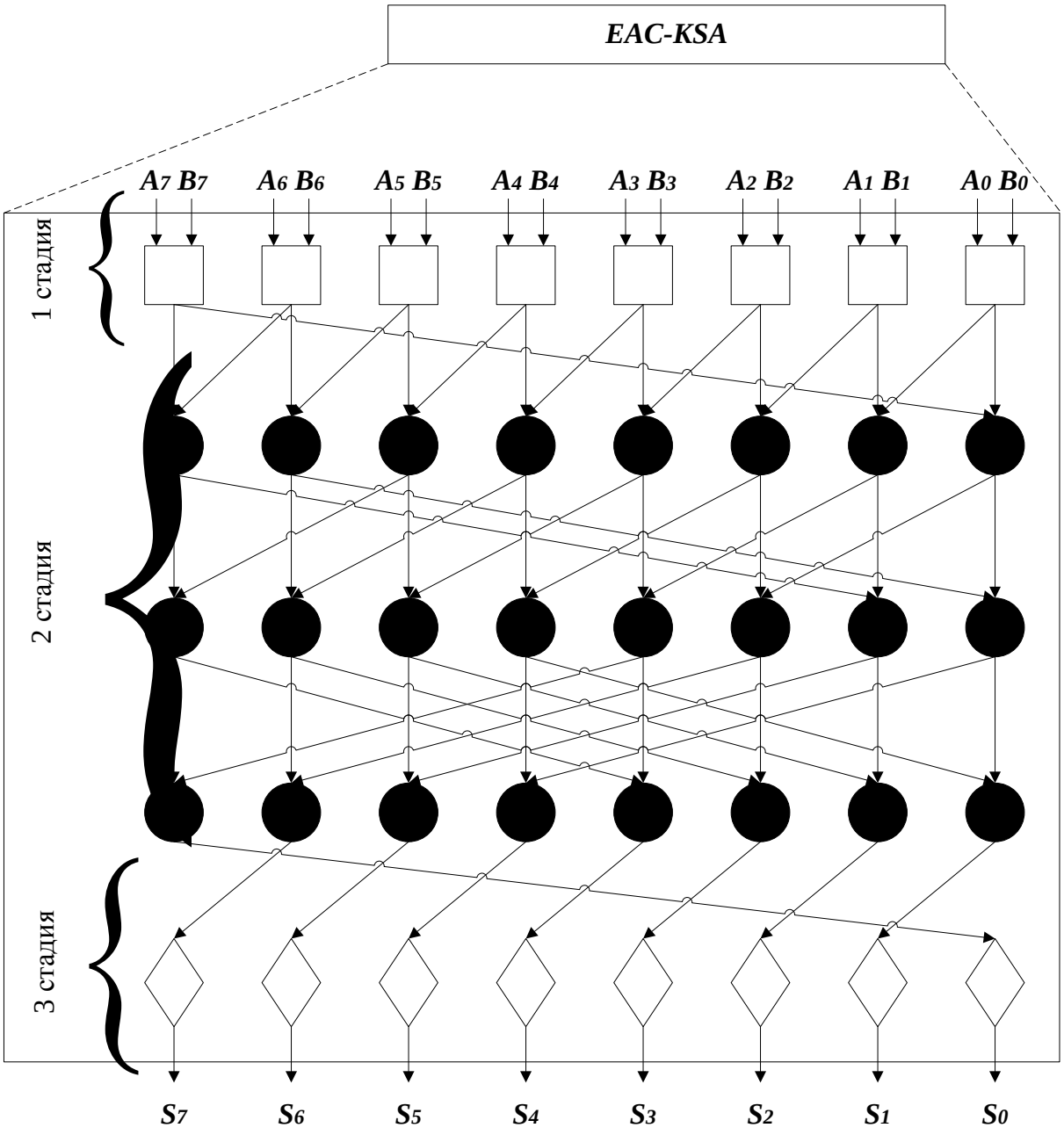


Рисунок 3.22. Структура 8-битного параллельно-префиксного сумматора Когге-Стоуна с циклическим переносом [124].

В таблицах 3.4 и 3.5 представлены результаты моделирования модулярных сумматоров СНС. Из таблицы 3.4 видно, что в предложенной архитектуре наибольшая задержка возникает в вычислительном канале по модулю $2^{16}-1$ и

составляет 9,127 нс. Из таблицы 3.5 видно, что в архитектуре СНС из [109] наибольшая задержка возникает в вычислительном канале по модулю 67 и составляет 7,453 нс. Таким образом, сумматоры в предложенной архитектуре работают медленнее на 22%, чем в известной архитектуре.

Таблица 3.3. Параметры синтеза

Параметр	**Значение**
Цель оптимизации	Скорость
Качество оптимизации	Высокое
Глобальная цель оптимизации	Максимальная задержка
Стиль реализации	Полностью параллельная
Извлечение ОП	Не выбрано
Извлечение ПЗУ	Не выбрано
Извлечение регистра сдвига	Не выбрано
Использование блоков цифровой обработки сигнала	Нет
Перемещение первой стадии триггера	Не выбрано
Перемещение последней стадии триггера	Не выбрано
Пакет регистров ввода-вывода	Нет
Объединение таблиц соответствия	Нет
Уменьшение набора элементов управления	Нет
Оптимизация конкретных примитивов	Не выбрано

Таблица 3.4. Результаты аппаратного моделирования модулярных сумматоров предложенной архитектуры

m	Задержка, нс	Использованные просмотровые таблицы FPGA	Использованные слайсы FPGA
2^7-1	7,500	40	22
$2^{11}-1$	7,692	81	32
$2^{13}-1$	7,618	90	36
$2^{16}-1$	**9,127**	108	51
$2^{17}-1$	8,160	121	49
$2^{15}-1$	9,097	105	41
2^{25}	8,646	142	74

Результаты моделирования блока прямого преобразования из позиционной системы счисления в СОК для предложенной архитектуры СНС и архитектуры из пункта 3.1 представлены в таблице 3.6. Из таблицы 3.6 видно, что операция

прямого преобразования в предложенной архитектуре работает в 21 раз быстрее и требует в 56 раз меньше аппаратных затрат.

Таблица 3.5. Результаты аппаратного моделирования модулярных сумматоров архитектуры [124]

m	Задержка, нс	Использованные просмотровые таблицы FPGA	Использованные слайсы FPGA
3	5,068	2	1
5	5,076	3	2
7	5,076	3	2
11	5,811	13	5
13	5,550	12	5
17	5,998	13	5
19	6,735	33	14
23	6,201	19	8
29	5,998	13	5
31	5,876	18	6
37	6,341	26	10
41	6,045	20	6
43	6,542	26	9
47	6,304	28	10
53	6,823	24	13
59	6,230	25	9
61	6,688	23	11
67	**7,453**	54	28
71	6,596	33	12
73	6,607	32	9
79	6,612	31	9
83	7,198	28	18

Результаты моделирования блока обратного преобразования из позиционной системы счисления в СОК для предложенной архитектуры СНС и архитектуры из пункта 3.1 представлены в таблице 3.7. Из таблицы 3.7 видно, что операция обратного преобразования в предложенной архитектуре работает на 25% быстрее и требует на 32,5% меньше аппаратных затрат.

Таким образом, предложенная архитектура СНС позволяет существенно сократить аппаратные и временные затраты на наиболее проблемные операции

прямого и обратного преобразования СОК. Достигнутое преимущество получено за счет небольшого снижения эффективности модулярных сумматоров СНС.

Таблица 3.6. Результаты аппаратного моделирования (прямое преобразование из ПСС в СОК)

	Предложенная архитектура	Архитектура из пункта 4.1
Задержка, нс	**21,434**	450,238
Использованные слайсы FPGA	**716**	40042

Таблица 3.7. Результаты аппаратного моделирования (обратное преобразование из СОК в ПСС)

	Предложенная архитектура	Архитектура из пункта 4.1
Задержка, нс	**60,158**	80,546
Использованные просмотровые таблицы FPGA	**16204**	22120
Использованные слайсы FPGA	**4182**	6773

Однако, на практике чаще всего для обработки изображений используются 8-битные изображения, поэтому проведем аналогичное моделирование работы СНС с вычислениями в СОК с модулями специального вида с представлением в 63 бита.

Моделирование производилось в среде ISE Design Suite 14.7. Целевая плата – Kintex 7 XC7K70T. Параметры синтеза представлены в таблице 3.8.

Результаты моделирования представлены в таблицах 3.9 и 3.10.

Таблица 3.8. Параметры синтеза

Свойство	Значение
Цель оптимизации	Скорость
Качество оптимизации	Высокое
Глобальная цель оптимизации	Максимальная задержка
Стиль реализации	Полностью параллельная
Извлечение ОП	Не выбрано
Извлечение ПЗУ	Не выбрано
Извлечение регистра сдвига	Не выбрано
Использование блоков цифровой обработки сигнала	Нет
Перемещение первой стадии триггера	Не выбрано
Перемещение последней стадии триггера	Не выбрано
Пакет регистров ввода-вывода	Нет
Объединение таблиц соответствия	Нет
Уменьшение набора элементов управления	Нет
Оптимизация конкретных примитивов	Не выбрано

Таблица 3.9. Результаты аппаратного моделирования (суммирование по модулям специального вида)

m	Задержка, нс	Таблицы соответствия слайсов	Занятые слайсы
2^7-1	7,500	40	22
$2^{11}-1$	7,692	81	32
$2^{13}-1$	7,618	90	36
$2^{17}-1$	8,160	121	49
2^{15}	**8,561**	44	21

Таблица 3.10. Результаты аппаратного моделирования прямого и обратного преобразования

	Прямое преобразование	Обратное преобразование
Задержка, нс	13,510	28,147
Таблицы соответствия слайсов	583	5474
Занятые слайсы	219	1591

3.4 Выводы по третьей главе

1. Разработана архитектура СНС с вычислениями в СОК с использованием модулей специального вида. Проведен сравнительный анализ работы модулярных сумматоров и устройств прямого и обратного преобразования в предложенной архитектуре и в известной сверточной нейронной сети. Показано, что наилучшая скорость выполнения операций прямого и обратного преобразования получена в предложенной архитектуре за счет оптимизации вычислений по модулям специального вида.

2. Проведено моделирование, которое показало, что прямое преобразование данных из ПСС в СОК для разработанной архитектуры СНС работает в 21 раз быстрее и требует в 56 раз меньше аппаратных затрат, чем известная архитектура СНС. Обратное преобразование в предложенной архитектуре СНС работает на 25% быстрее, чем в известной архитектуре СНС и требует на 32,5% меньше аппаратных затрат.

ГЛАВА 4. МОДЕЛИРОВАНИЕ АРХИТЕКТУРЫ СВЕРТОЧНОЙ НЕЙРОННОЙ СЕТИ С ВЫЧИСЛЕНИЯМИ В СИСТЕМЕ ОСТАТОЧНЫХ КЛАССОВ

В пункте 1.2 было отмечено, что основной задачей, которая возникает в системах цифрового видеонаблюдения, в контексте цифровой обработки изображений является задача распознавания образов. Ранее в качестве основного инструмента поставленной задачи была выбрана СНС. Выбор именно СНС обоснован тем, что она имитируют работу человеческого зрения. В функции зрительной системы человека входит создание представления окружающего мира в таком виде, который обеспечивает возможность взаимодействия с этим миром. Мозг последовательно выполняет ряд задач распознавания, например, распознавание знакомого лица в незнакомом окружении. В связи с этим алгоритмы на основе работы СНС нашли широкое применение во встраиваемых системах технического зрения, включающих в себя решение задач распознавания рукописного текста [99], распознавание лиц [116], распознавание места [113] и распознавание объектов [87].

К преимуществам нейронных сетей можно отнести распараллеливание обработки информации и способность самообучаться, т.е. создавать обобщения [70]. Большинство известных алгоритмов СНС требуют значительных затрат памяти, для хранения весовых коэффициентов при обучении, и времени работы [86,99,109]. В [109] приводится архитектура СНС, однако предложенная авторами идея об использовании многоступенчатой вложенной СОК для ее реализации требует неоправданно больших временных и аппаратных затрат на прямое и обратное преобразование из ПСС в СОК. Кроме того, в работе [109] использованы только модули общего вида, что не позволяет использовать математические свойства модулей специального вида, приводящие к оптимизации аппаратной архитектуры устройств модулярных операций. В данной главе будет представлена архитектура СНС, построенная с использованием модулей СОК

специального вида и без использования вложенных преобразований, что позволит сократить аппаратные и временные затраты на реализацию нейронной сети.

4.1 Программное моделирование сверточной нейронной сети для распознавания изображений

В главе 1 пункте 1.6.2 приводятся основные определения, касающиеся понятия СНС, а также пример ее архитектуры, состоящий из 8 слоев. Рассмотрим более подробно процесс получения карт признаков в СНС.

Предположим, что изображение I, состоящее из R строк и C столбцов и D слоев представляет собой трехмерную функцию $I(x, y, z)$, где $0 \le x < R$ и $0 \le y < C$, $0 \le z < D$ – это пространственные координаты, а амплитуда I в любой точке с координатами (x, y, z) называется интенсивностью в этой точке.

Процедура получения карт признаков может быть представлена в виде свертки:

$$I_f(x, y, z) = \sum_{k=0}^{D-1} \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} W_{i,j,k} I(x+i, y+j, z+k), \qquad (4.1)$$

где I_f - обработанное изображение, а $W_{i,j,k}$ – коэффициенты маски фильтра для обработки D двумерных массивов [123]. Наглядно процедура получения карт признаков показана на рисунке 4.1.

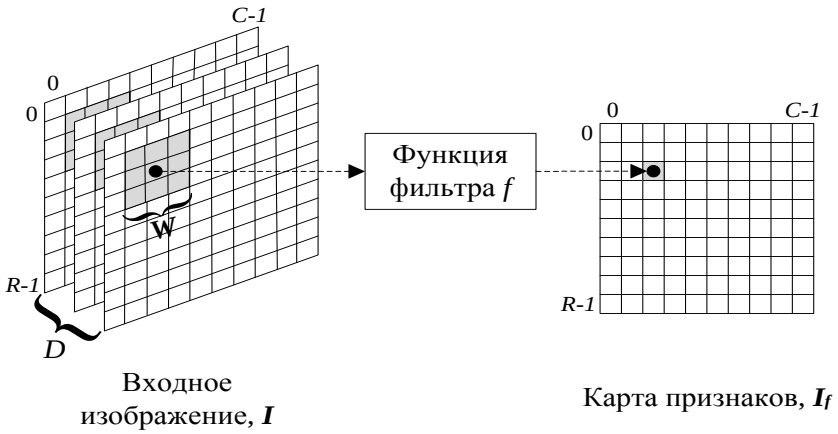


Рисунок 4.1. Процедура получения карт признаков.

На рисунке 4.2 представлена процедура выбора максимальных элементов изображения с помощью маски фильтра размером $m \times m$ и шагом вычислений m. Для части изображения, состоящего из четырех пикселей величиной 1, 5, 8 и 2 максимальным элемент будет 8.

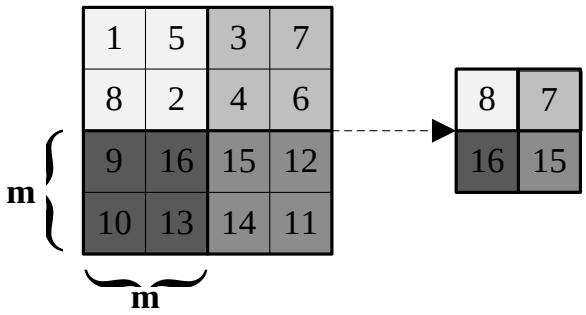


Рисунок 4.2. Процедура выбора максимальных элементов

Поставим задачу, в ходе решения которой создадим СНС, распознающую 8 образов на примере базы изображений из университета Иллинойса (Image database of the University of Illinois at Urbana-Champaign) НЕ ВСТАВИЛА. Размер изображений исходной базы был модифицирован с помощью программы Photoshop CS6 с использованием алгоритма бикубической интерполяции. На рисунке 4.3 представлены классы изображений, всего для обучения используются 161 изображения, принадлежащие разным классам. Пример набора изображений одного класса изображен на рисунке 4.4.

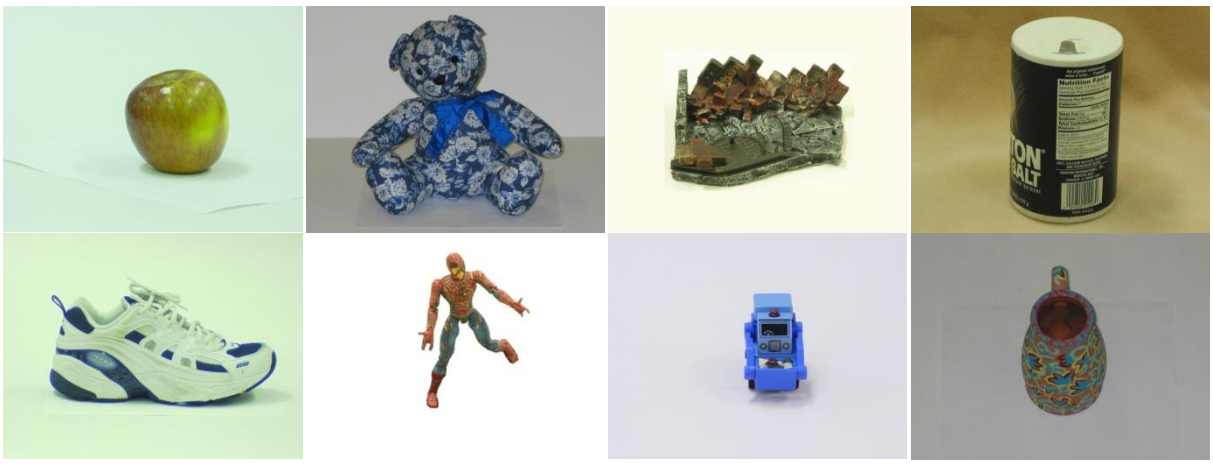


Рисунок 4.3. Классы изображений

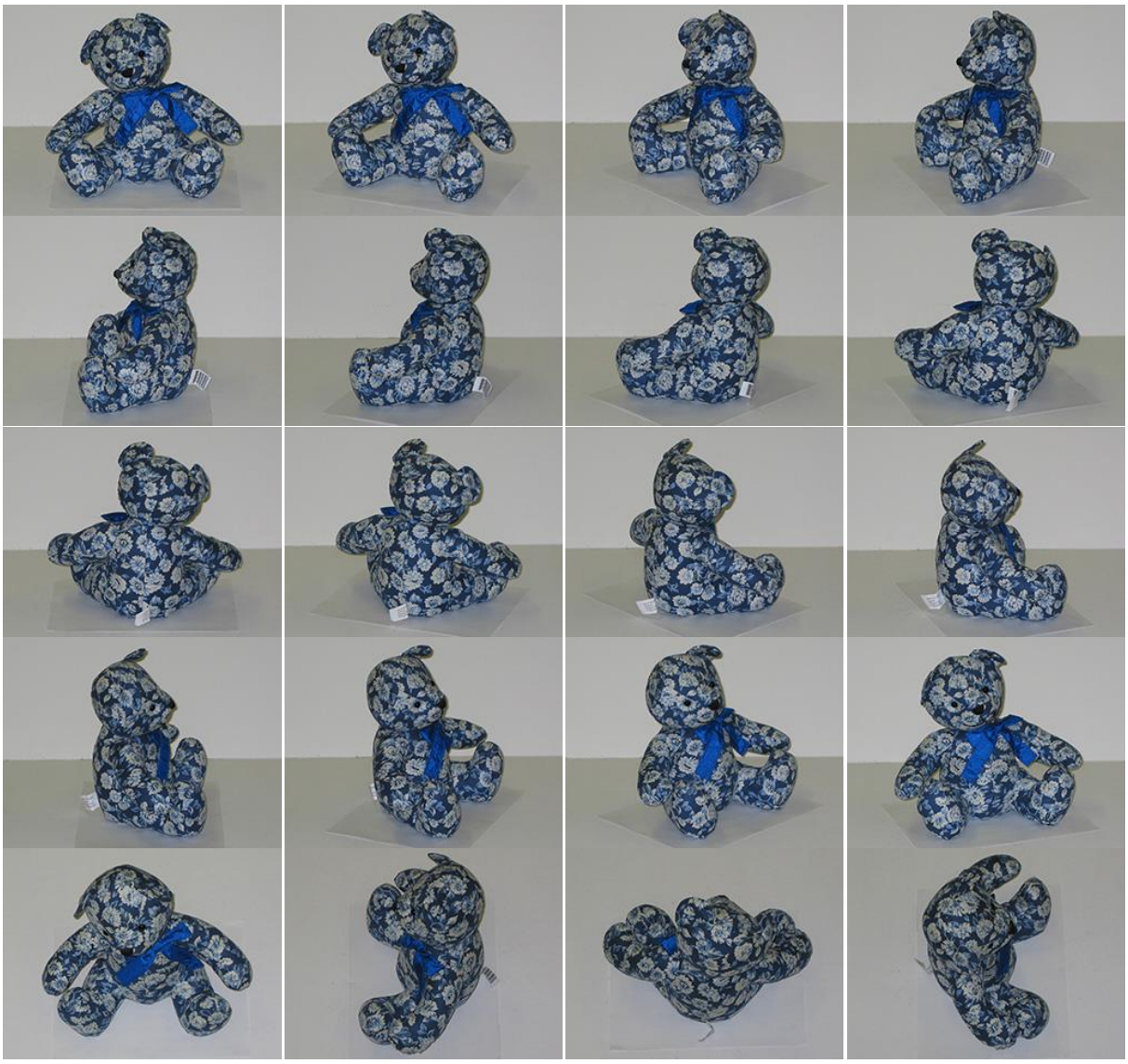


Рисунок 4.4. Пример изображений одного класса

В качестве примера рассмотрим архитектуру СНС, в которой на вход подается RGB изображение размера 256×192, первые четыре слоя отвечают за выделение признаков изображения, чередуя слои двумерной свертки и выбора максимальных элементов (рисунок 4.5). Первый слой производит операцию свертки с помощью 12-ти масок, размер маски фильтра $[5\times5]\times3$, а шаг вычислений 5. Результатом вычислений первого слоя являются 12 карт признаков размером 51×38. Второй слой осуществляет выбор максимальных элементов, размер маски 2×2, шаг вычислений 2. На выходе второго слоя формируются 12 карт признаков размером 25×19. Третий слой снова реализует операцию свертки с помощью 18-ти масок, размер маски фильтра $[3\times3]\times12$, шаг

вычислений 3. Результатом работы третьего слоя является 18 карт признаков размером 8×6. Четвертый слой работает так же как и второй, на выходе формируется карта признаков размером 4×3. Последние три слоя отвечают за классификацию изображения и представляют из себя полносвязную нейронную сеть. Пятый слой состоит из 18 нейронов, шестой из 12, а седьмой из 8 нейронов каждый из которых соответствует определенному классу.

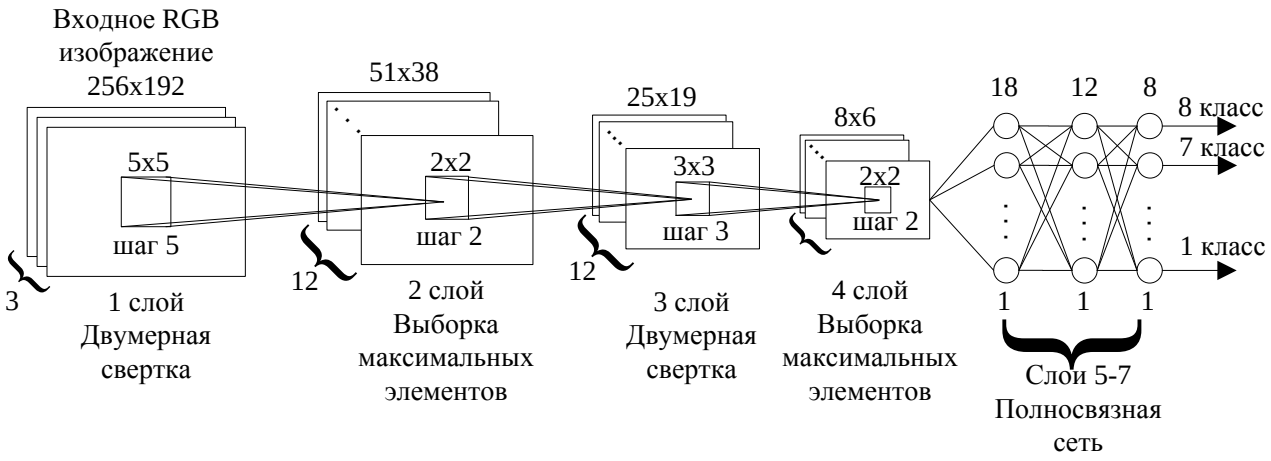


Рисунок 4.5. Предложенная архитектура сверточной нейронной сети

Для решения поставленной задачи предлагается использовать архитектуру СНС, представленную на рисунке 4.6. Сеть состоит из четырех слоев. На вход подается RGB изображение размера 256×192, первые два слоя отвечают за выделение признаков изображения. Первый слой производит операцию свертки с помощью 8-ми фильтров, размер маски фильтра $[3\times3]\times3$, а шаг вычислений 3. Результатом вычислений первого слоя являются 8 карт признаков размером 85×64. Для операции свертки использовались фильтры Собеля [118], которые осуществляют выделение контуров по направлению (таблица 4.1). Второй слой осуществляет выбор максимальных элементов, размер маски 2×2, шаг вычислений 2. На выходе второго слоя формируются 8 карт признаков размером 42×32. Последние два слоя отвечают за классификацию изображения и представляют из себя полносвязную нейронную сеть. Третий слой состоит из 10

нейронов, а четвертый из 8 нейронов каждый из которых соответствует определенному классу.

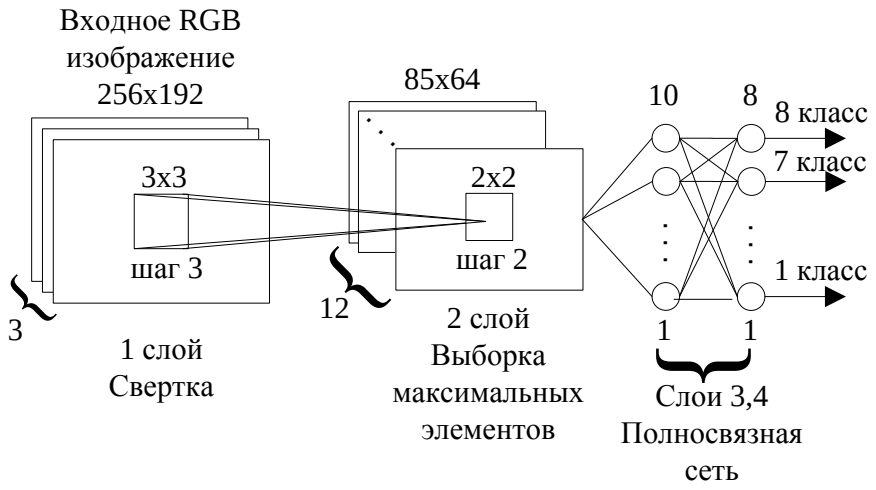


Рисунок 4.6. Предложенная архитектура сверточной нейронной сети

Таблица 4.1. Маски фильтров для первого слоя СНС

№ п/п	Маска фильтра	№ п/п	Маска фильтра
1	$\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$	5	$\begin{bmatrix} 0 & -1 & -2 \\ 1 & 0 & -1 \\ 2 & 1 & 0 \end{bmatrix}$
2	$\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$	6	$\begin{bmatrix} -2 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 2 \end{bmatrix}$
3	$\begin{bmatrix} 2 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -2 \end{bmatrix}$	7	$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$
4	$\begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{bmatrix}$	8	$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$

Следует отметить, что выбор фильтра Собеля для осуществления операции выделения признаков обоснован основной функцией, которую выполняет данный класс фильтров. Фильтры Собеля выделяют контуры, а в свою очередь, этой

информации достаточно, чтобы сделать вывод о том, к какому классу принадлежит распознаваемое изображение.

Для оценки временных затрат работы СНС вначале проведем моделирование процесса с различными конфигурациями сети и различным разрешением входного изображения. В качестве варьируемых параметров будем использовать разные размеры масок фильтров и шаг вычислений.

Моделирование производилось в математическом пакете Matlab R2014a [95]. Характеристики компьютера представлены в таблице 4.2.

Таблица 4.2. Характеристики компьютера

Наименование	Значение
Процессор	Intel(R) Core(TM) i7-4790K CPU @ 4.00GHz, 4.00GHz
Установленная память (ОЗУ)	16,0 Гб
Операционная система	Windows 7 Домашняя расширенная
Тип системы	64-разрядная

В таблице 4.3 приведены параметры первых четырех слоев с различными ее конфигурациями. Результаты моделирования приведены в таблице 4.4. Диаграммы результатов моделирования представлены на рисунке 4.7.

Из рисунка 4.7 а-г видно, что самыми вычислительно сложными для всех четырех моделируемых конфигураций являются 1 и 3 слои СНС, в то время как, меньше всего временных затрат требует 4 слой. Из таблицы 4.4 видно, что в среднем временные затраты для первого слоя составляют 18%, для третьего слоя 80%. Причем временные затраты прямо пропорционально зависят от размера изображения.

Таким образом, результаты моделирования показали, что при уменьшении разрешения исходного изображения и размера масок фильтров временные затраты сокращаются. Оптимальный результат достигается при использовании третьей конфигурации сети и разрешении изображения 256×192.

Таблица 4.3. Конфигурации сети

Слой / №п/п	1	2	3	4
1	$[11\times11]\times3$ количество 24 шаг 11	4×4 шаг 4	$[5\times5]\times24$ количество 24 шаг 5	2×2 шаг 2
2	$[11\times11]\times3$ количество 24 шаг 1	4×4 шаг 1	$[5\times5]\times24$ количество 24 шаг 1	2×2 шаг 1
3	$[5\times5]\times3$ количество 12 шаг 5	2×2 шаг 2	$[3\times3]\times12$ количество 18 шаг 3	2×2 шаг 2
4	$[5\times5]\times12$ количество 12 шаг 1	2×2 шаг 1	$[3\times3]\times12$ количество 18 шаг 1	2×2 шаг 1

Таблица 4.4. Временные затраты

Размер изображения	Конфигурация / Слой	1	2	3	4
1280×960	1	4,9452	478,4239	6,0528	132,5852
	2	0,0156	19,1413	0,0780	4,8984
	3	0,0312	2059,1000	0,7020	647,0921
	4	0,0000	9,6721	0,0000	7,7064
1024×768	1	3,2604	304,2644	3,8220	83,9597
	2	0,0156	12,5737	0,0468	3,0420
	3	0,0624	1310,5000	0,4368	411,4682
	4	0,0000	6,2712	0,0000	4,9608
768×576	1	1,7940	170,2127	2,1840	47,0655
	2	0,0156	7,1292	0,0156	1,6536
	3	0,0156	739,5695	0,2496	229,5555
	4	0,0000	3,5880	0,0000	2,6520
512×384	1	0,7488	75,3173	0,9204	20,8729
	2	0,0000	3,1356	0,0156	0,7488
	3	0,0156	324,6693	0,0936	101,1198
	4	0,0000	1,5600	0,0000	1,1076
256×192	1	1,1716	18,8605	**0,2184**	5,2104
	2	0,0000	0,7800	**0,0000**	0,1716
	3	0,0000	80,1221	**0,0312**	24,9602
	4	0,0000	0,3588	**0,0000**	0,2652

Обучение СНС производилось в математическом пакете Matlab R2014a с помощью инструмента Neural Pattern Recognition. В процессе обучения участвовало 161 изображение, принадлежащие 8-ми классам [12]. Из них 70%

используются непосредственно для обучения сети, 15% – для измерения обобщения сети и прекращения обучения, когда обобщение перестает улучшаться, и еще 15% – обеспечивают независимую оценку производительности сети вовремя и после обучения, так как сами не влияют на процесс обучения. Обучались только последние два слоя СНС, третий слой (скрытый слой) содержит 10 нейронов, а четвертый (выходной) содержит 8.

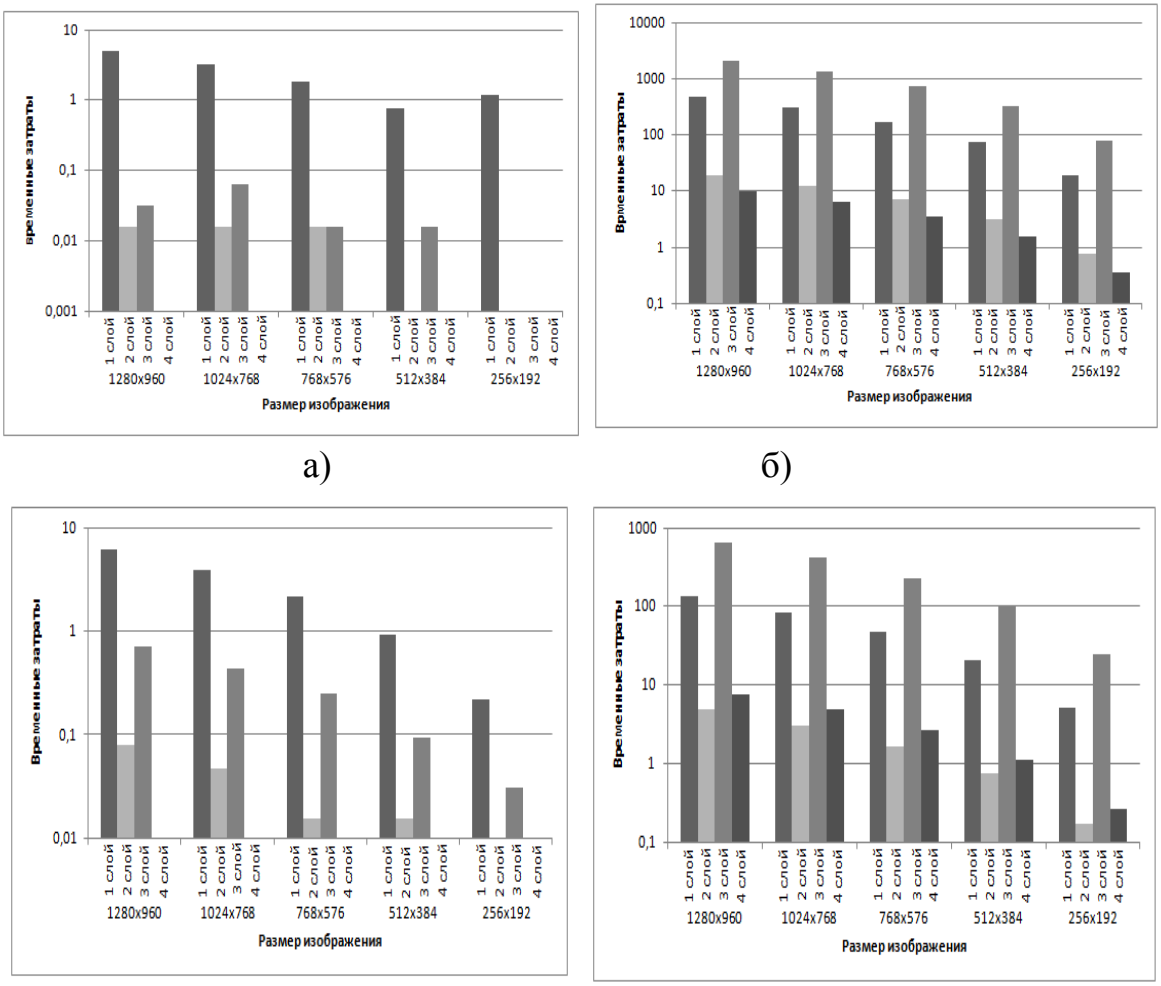


Рисунок 4.7 Диаграммы временных затрат первых четырех слоев СНС в зависимости от конфигурации сети и размерности изображений: а) конфигурация 1; б) конфигурация 2; в) конфигурация 3; г) конфигурация 4.

Использовался алгоритм обучения масштабированное сопряженное градиентное обратное распространение [70,99].

Пусть $\{x_i\}, i=\overline{1,m}$ – вектор подаваемый на вход третьего слоя, m – общее число входов. Каждый элемент вектора умножается на соответствующий весовой коэффициент w_{ji}, $i=\overline{1,m}$, $j=\overline{1,n}$, n – количество нейронов третьего слоя и результат суммируется:

$$y_j = \sum_{i=1}^{m} w_{ji} x_i \tag{4.2}$$

Результатом третьего слоя является вектор $\{h_j\}$, $j=\overline{1,n}$. Функция активации для скрытого слоя имеет вид:

$$\varphi(t) = \frac{2}{1+e^{-2t}} - 1 \tag{4.3}$$

таким образом, значение нейронов третьего слоя вычисляется как:

$$h_j = \frac{2}{1+e^{-2y_j}} - 1 \tag{4.4}$$

Результат вычислений третьего слоя подается на вход четвертого слоя, производим процедуру умножения на соответствующие весовые коэффициенты, результатом вычислений является вектор $\{z_k\}, k=\overline{1,l}$, l – количество нейронов третьего слоя. Результатом четвертого слоя является вектор $\{g_k\}$, $k=\overline{1,l}$. Функция активации выходного слоя имеет вид:

$$\psi(t) = e^{-2t} \tag{4.5}$$

таким образом значение нейронов последнего слоя вычисляется как:

$$g_k = \frac{e^{\left[z_k - \max\limits_{k=\overline{1,l}} z_k\right]}}{\sum\limits_{k=1}^{l} e^{\left[z_k - \max\limits_{k=\overline{1,l}} z_k\right]}} \tag{4.6}$$

Пересчет весовых коэффициентов последнего слоя производится следующим образом. Вычисляется сигнал ошибки:

$$e_k = d_k - g_k, \tag{4.7}$$

где $\{d_k\}, k=\overline{1,l}$ – целевой вектор. Далее вычисляется локальный градиент δ_k, $k=\overline{1,l}$ равный:

$$\delta_k = e_k \psi'(z_k) \tag{4.8}$$

Коррекция Δw_{kj}, применяемая к весовым коэффициентам w_{kj} четвертого слоя, определяется согласно правилу:

$$\Delta w_{kj} = \eta \delta_k g_k, \tag{4.9}$$

где η – параметр скорости обучения.

Для третьего слоя локальный градиент вычисляется следующим образом:

$$\delta_j = \varphi'(y_j) \sum_{k=1}^{l} \delta_k w_{kj} \tag{4.10}$$

Коррекция Δw_{ji}, применяемая к весовым коэффициентам w_{ji} четвертого слоя, определяется согласно правилу:

$$\Delta w_{ji} = \eta \delta_j y_j, \tag{4.11}$$

Нейронная сеть была обучена за 59 итераций, на рисунке 4.8 представлен график процесса обучения. Нейронная сеть обучалась до момента пока значения функции ошибки сети не стало меньше, чем 0,05. Это значение было достигнуто на 59 итерации обучения. Так же из графика видно, что наилучший показатель качества распознавания достигался на 53 итерации обучения. На рисунке 4.9 изображен график изменения градиента в процессе обучения. На 59 итерации обучения величина градиента составляла 0,029.

Обучение нейронной сети заканчивается после наступления шести подряд идущих ухудшений проверочной функции (рисунок 4.8.). В качестве обученной нейронной сети принимается последнее, предшествующее этим шести ухудшениям, состояние. На рисунке 4.9. видно, что это состояние наступило на 53 итерации обучения, и в этом случае достигалось минимальное значение ошибки проверочной функции. На рисунке 4.10. показан график изменения градиента нейронной сети, который имеет тенденцию к снижению со временем, что свидетельствует о приближении к требуемому экстремальному значению функции обучения СНС.

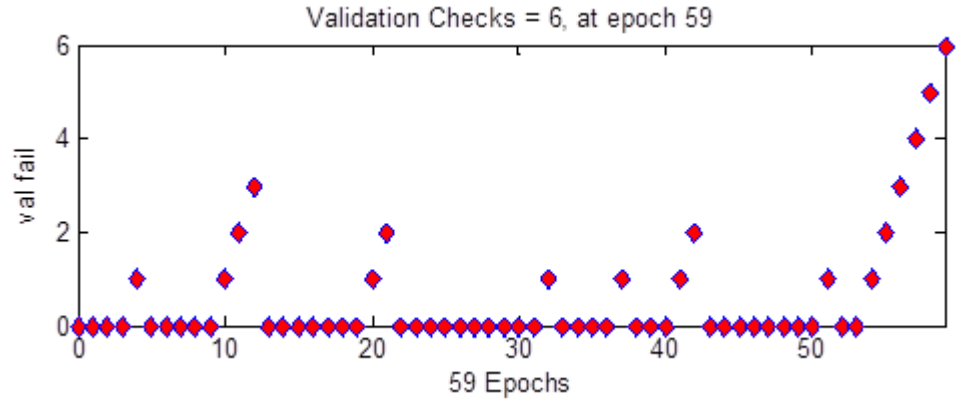


Рисунок 4.8. График ошибки проверочной функции CHC

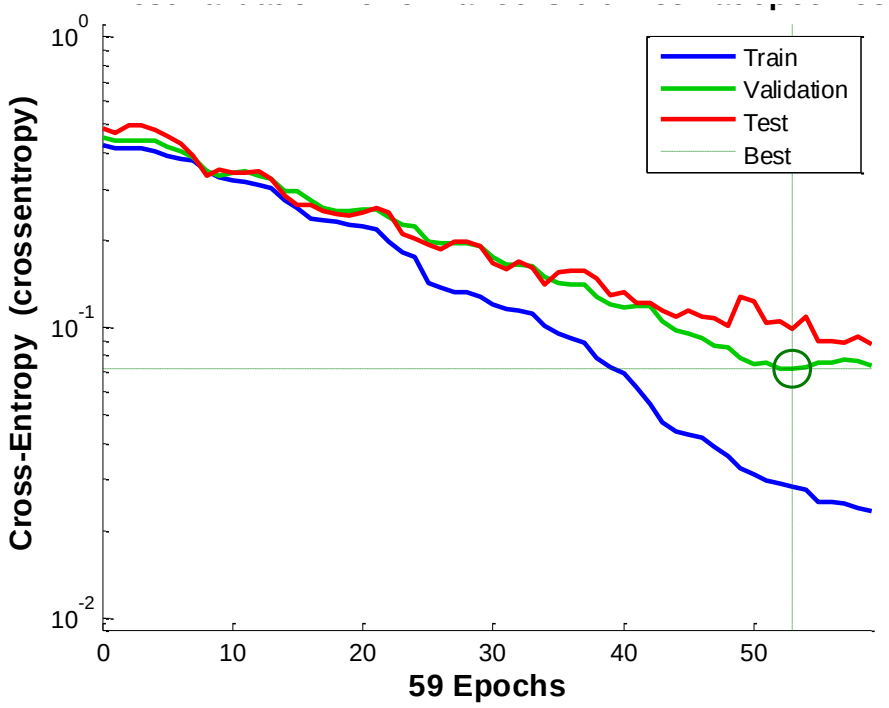


Рисунок 4.9. График результатов обучения нейронной сети

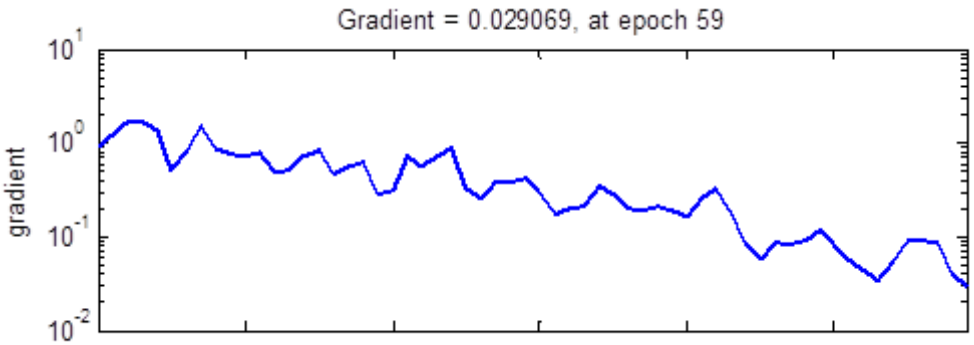


Рисунок 4.10. График изменения градиента в процессе обучения

Проведем настройку параметров СНС с целью выявления наилучших из них. В процессе настройки параметров будем варьировать такие параметры как количество фильтров и количество нейронов для последнего скрытого слоя. Для определения оптимальной аппаратной конфигурации будем варьировать разрядность коэффициентов нейронной сети.

Для тестирования способности СНС к отнесению произвольных изображений к указанным классам были отобраны 80 изображений из различных электронных ресурсов.

Для оценки качества работы СНС использовались следующие критерии:

1. Изображение считается распознанным, если отклик СНС по нужному классу превышает 50%. Данный критерий обозначен как k_1;

2. Изображение считается распознанным, если отклик СНС по нужному классу больше отклика по любому другому классу. Данный критерий обозначен как k_2

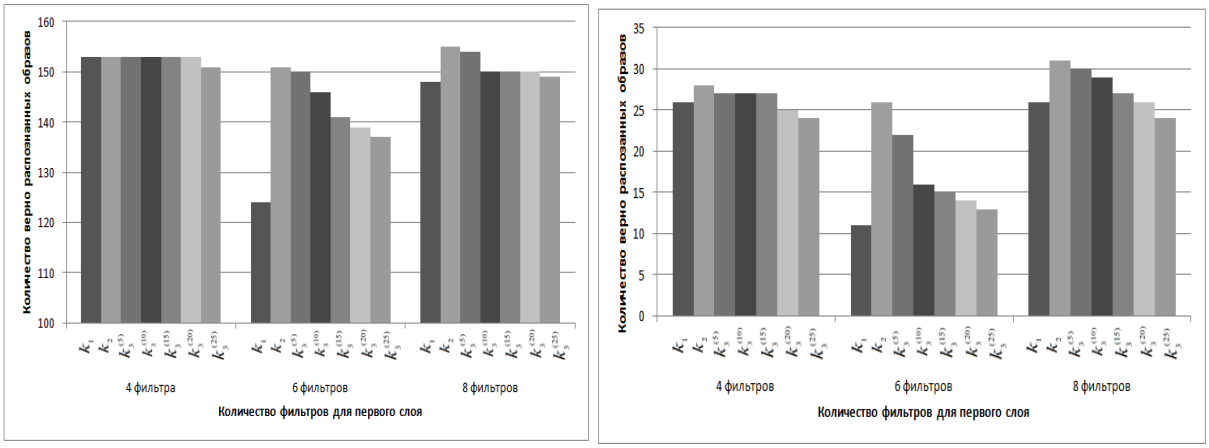
3. Изображение считается распознанным, если отклик СНС по нужному классу больше отклика по любому другому классу и разница между двумя классами с наибольшим откликом больше чем 5% (обозначается $k_3^{(5)}$), 10% (обозначается $k_3^{(10)}$), 15% (обозначается $k_3^{(15)}$), 20% (обозначается $k_3^{(20)}$), 25% (обозначается $k_3^{(25)}$).

Результаты тестирования различных конфигураций СНС представлены в таблицах 4.5-4.7. В таблицах под исходной базой понимается база из 161 изображения, использованная при обучении, а тестовая база - база 80 изображений использованная при тестировании.

Из рисунка 4.11 а-г видно, что наибольшее число распознанных изображений СНС получает при использовании 8 фильтров для первого слоя СНС. Причем количество верно распознанных изображений прямо пропорционально зависят от количества цифровых фильтров, используемых в слое СНС.

Таблица 4.5. Количество верно распознанных образов в зависимости от количества фильтров

Количество фильтров первого слоя		k_1	k_2	$k_3^{(5)}$	$k_3^{(10)}$	$k_3^{(15)}$	$k_3^{(20)}$	$k_3^{(25)}$
Исходная база	4	153	153	153	153	153	153	151
	6	124	151	150	146	141	139	137
	8	**148**	**155**	**154**	**150**	**150**	**150**	**149**
Тестовая база	4	26	28	27	27	27	25	24
	6	11	26	22	16	15	14	13
	8	**26**	**31**	**30**	**29**	**27**	**26**	**24**



а) б)

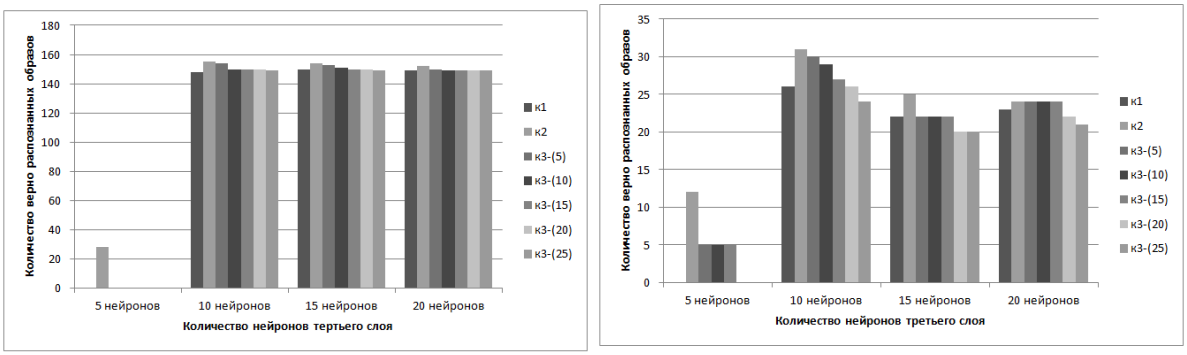
Рисунок 4.11 Диаграмма количества верно распознанных изображений для первого слоя СНС: а) исходная база; б) тестовая база

Таким образом, результаты моделирования показали, что при увеличении количества масок фильтров, количество верно распознанных изображений увеличивается.

Из рисунка 4.12 а-г видно, что наибольшее число распознанных изображений СНС получает при использовании 10 нейронов в третьем слое СНС. Наименьшее количество распознанных изображений достигается при использовании 5 нейронов в третьем СНС.

Таблица 4.6. Количество верно распознанных образов в зависимости от количества нейронов третьего слоя

Количество нейронов третьего слоя		k_1	k_2	$k_3^{(5)}$	$k_3^{(10)}$	$k_3^{(15)}$	$k_3^{(20)}$	$k_3^{(25)}$
Исходная база	5	0	28	0	0	0	0	0
	10	**148**	**155**	**154**	**150**	**150**	**150**	**149**
	15	150	154	153	151	150	150	149
	20	149	152	150	149	149	149	149
Тестовая база	5	0	12	5	5	5	0	0
	10	**26**	**31**	**30**	**29**	**27**	**26**	**24**
	15	22	25	22	22	22	20	20
	20	23	24	24	24	24	22	21



а) б)

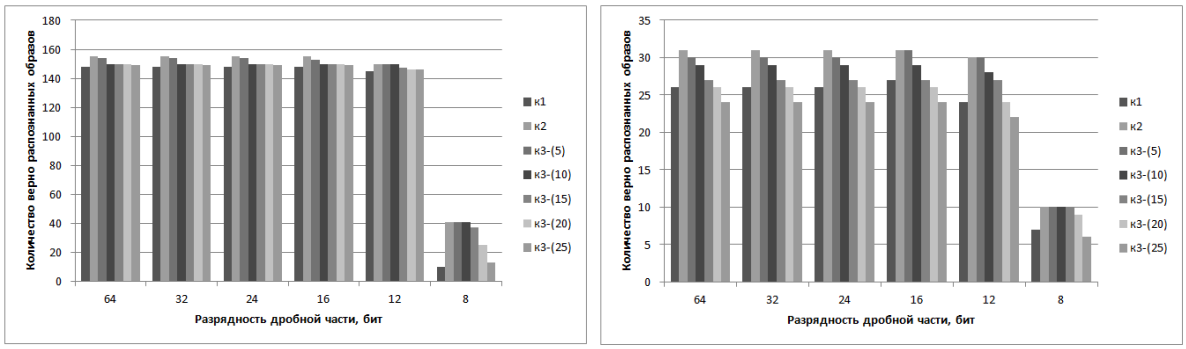
Рисунок 4.12 Диаграмма количества верно распознанных изображений для третьего слоя СНС: а) исходная база; б) тестовая база

Из рисунка 4.13 а-г видно, что качество распознавания СНС остается практически неизменным при снижении разрядной части до 16 бит, при уменьшении разрядности качество распознавания заметно снижается.

Таким образом, изучение зависимости количества верно распознанных образов от разрядности дробной части весовых коэффициентов третьего и четвертого слоев показало, что для разрядности дробной части можно использовать 16 бит без потери качества распознавания.

Таблица 4.7. Количество верно распознанных образов в зависимости от разрядности дробной части весовых коэффициентов третьего и четвертого слоев

Разрядность дробной части, бит		k_1	k_2	$k_3^{(5)}$	$k_3^{(10)}$	$k_3^{(15)}$	$k_3^{(20)}$	$k_3^{(25)}$
Исходная база	64	148	155	154	150	150	150	149
	32	148	155	154	150	150	150	149
	24	148	155	154	150	150	150	149
	16	**148**	**155**	**153**	**150**	**150**	**150**	**149**
	12	145	150	150	150	147	146	146
	8	10	41	41	41	37	25	13
Тестовая база	64	26	31	30	29	27	26	24
	32	26	31	30	29	27	26	24
	24	26	31	30	29	27	26	24
	16	**27**	**31**	**31**	**29**	**27**	**26**	**24**
	12	24	30	30	28	27	24	22
	8	7	10	10	10	10	9	6



а) б)

Рисунок 4.13 Диаграмма количества верно распознанных изображений в зависимости от разрядности дробной части: а) исходная база; б) тестовая база

Таким образом, выбрана конфигурация содержащая 8 фильтров для операции свертки, 10 нейронов третьего слоя и 16 разрядов дробной части весовых коэффициентов третьего и четвертого слоев. На рисунке 4.14 представлен пример результатов работы предлагаемой СНС.

Тестирование показало, что СНС распознает 149 изображений из исходной базы и 24 изображения из тестовой базы.

Рисунок 4.14. Пример результатов работы сверточной нейронной сети

Далее будет показана аппаратная реализация СНС, с использованием оптимальных параметров, полученных в данном пункте.

4.2 Аппаратная реализация сверточной нейронной сети для распознавания изображений

В пункте 1.7. главы 1 были приведены основные понятия системы остаточных классов, также было обозначено, что одним из важных вопросов при использовании такой системы счисления является правильный выбор набора модулей. Наборы модулей специального вида $\{2^n, 2^n-1, 2^{n+k}-1\}$ позволяют использовать быстродействующие алгоритмы вычислений таких операций как сложение, умножение, преобразование числа из позиционной системы счисления в СОК и обратно [67,68,98,124].

Рассмотрим процедуру получения карт признаков изображений с помощью фильтров Собеля. Маски данных фильтров имеют размерность 3×3 и одинаковые наборы коэффициентов, отличающиеся лишь их расположением. Коэффициенты масок фильтров равные 0 могут быть исключены из вычислений. Таким образом, операция пространственной свертки сводится к умножению на числа 1, -1, 2 и -2 соответствующих пикселей изображения и сложению полученных результатов.

Для аппаратной реализации операции пространственной свертки на FPGA необходимо использовать кэш-память. Одной из распространенных форм кэширования является буферизация строк [123, 74]. На рисунке 4.15 изображено кэширование пикселей с буферизацией строк для синхронизации. Здесь на вход подается RGB изображение I, состоящее из трех слоев, результатом является изображение I_f, обработанное с помощью функции фильтра f.

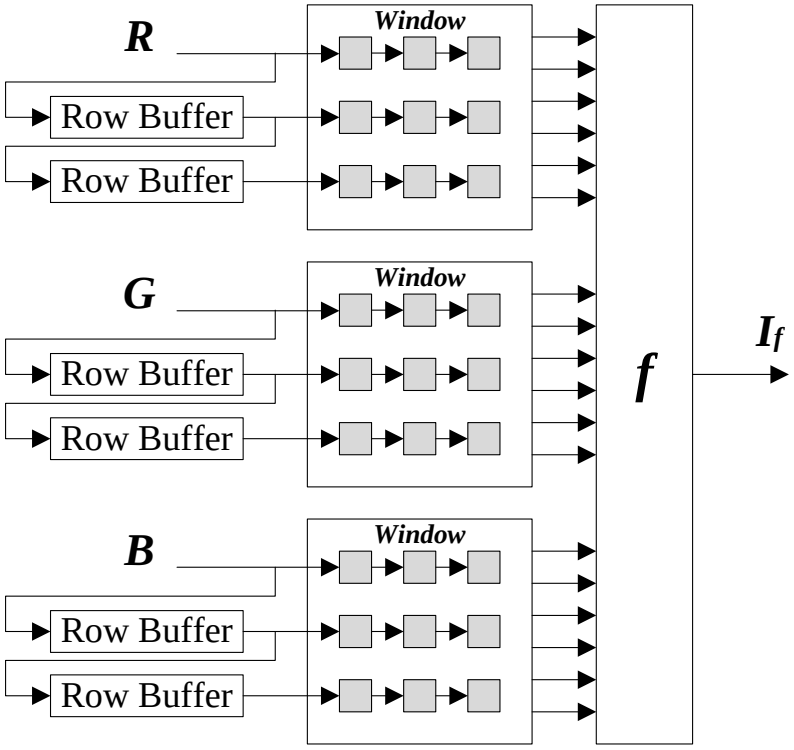


Рисунок 4.15. Кэширование пикселей с буферизацией строк для синхронизации

На рисунке 4.16 представлена архитектура устройства, выполняющего операцию свертки по модулю 2^n. Каналы F_1, F_3, F_7, F_9, F_{13} и F_{15} соответствуют умножению на коэффициент 1, каналы F_4, F_6, F_{10}, F_{12}, F_{16} и F_{18} – на коэффициент -1, каналы F_2, F_8 и F_{14} – на коэффициент 2 и каналы F_5, F_{11} и F_{17} – на коэффициент -2. Умножение на -1 по модулю 2^n осуществляется как инверсия числа с добавлением 1, поэтому вводится корректирующий коэффициент равный количеству умножений на -1. Умножение на 2 по модулю 2^n производится как сдвиг числа на один бит влево (Shift left). Сложение представляет собой дерево сумматоров с сохранением переноса (CSA-tree), результат которого суммируется с помощью параллельно-префиксного сумматора (KSA) [67,68,98,124].

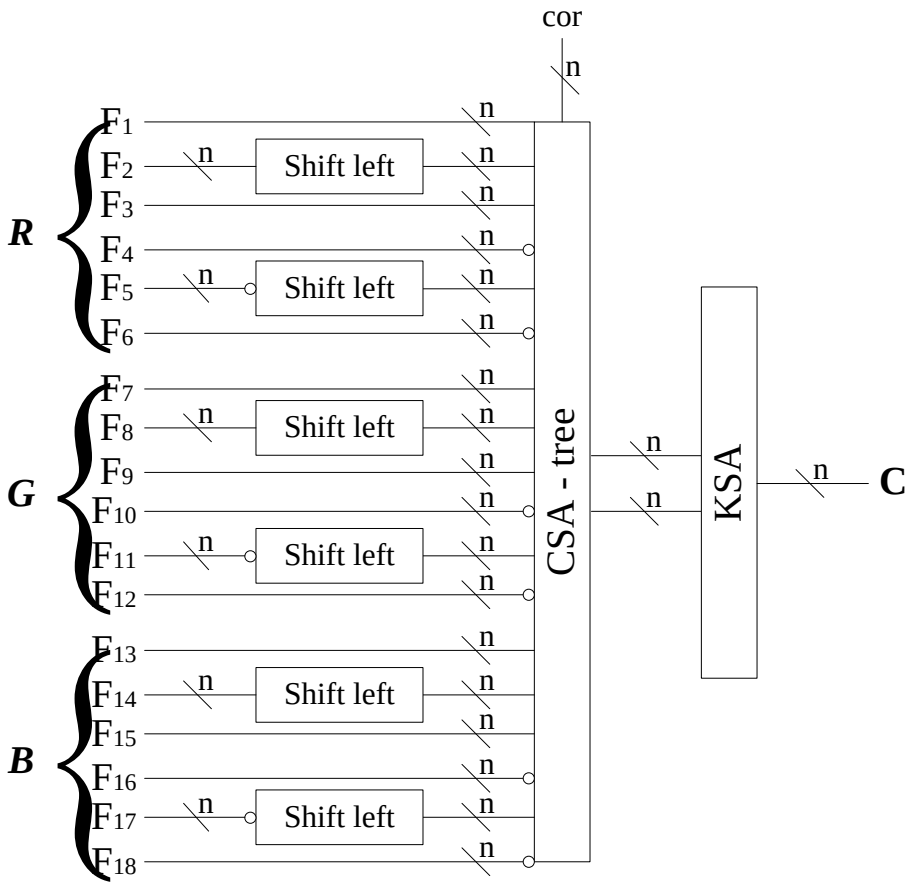


Рисунок 4.16. Архитектура устройства, реализующего пространственную свертку по модулю 2^n

На рисунке 4.17 изображена архитектура устройства, выполняющего операцию свертки по модулю $2^n - 1$. Здесь умножение на -1 осуществляется как инверсия числа, умножение на 2 производится как циклический сдвиг числа на один бит влево (Shift left around). Сложение представляет собой дерево сумматоров с сохранением переноса и циклическим переносом (EAC-CSA-tree), результат которого суммируется с помощью параллельно-префиксного сумматора с циклическим переносом (EAC-KSA) [67,68,98,124].

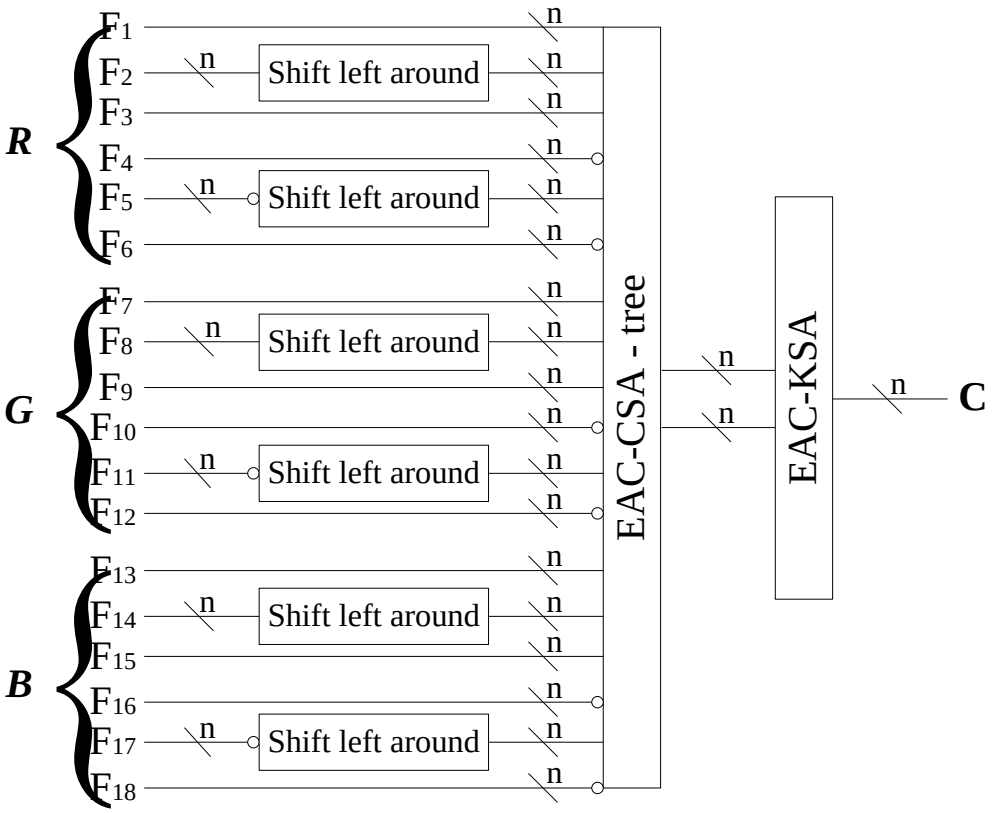


Рисунок 4.17. Архитектура устройства, реализующего пространственную свертку по модулю $2^n - 1$

Аппаратное моделирование производилось в среде ISE Design Suite 14.7. Целевая плата – Kintex 7 XC7K70T. Параметры синтеза представлены в таблице 4.8.

Целью моделирования было сравнение работы схемы пространственной свертки в двоичной системе счисления (ДСС), в СОК с использованием модулей

специального вида и в СОК с использованием набора модулей произвольного вида [109]. Результаты моделирования представлены в таблицах 4.9 и 4.10.

Из таблицы 4.9 видно, что при использовании произвольного набора модулей наибольшая задержка возникает в вычислительном канале по модулю 17 и составляет 19,518 нс. Из таблицы 4.10 видно, что при использовании модулей специального вида наибольшая задержка возникает в вычислительном канале по модулю 2^9 и составляет 15,909 нс. При использовании ДСС возникает задержка 25,404 нс, устройство использует 495 таблиц соответствия слайсов и 286 занятых слайсов.

На рисунке 4.18 представлены диаграммы временных и аппаратных затрат пространственной свертки в двоичной системе счисления (ДСС), в СОК с использованием набора модулей произвольного вида и в СОК с использованием модулей специального вида. Таким образом, использование модулей специального вида позволяет ускорить работу устройства на 37,4% по сравнению с использованием модулей произвольного вида за счет увеличения аппаратных затрат примерно в 2 раза, и ускорить работу устройства на 18,5% по сравнению с использованием модулей произвольного вида при сократить аппаратных затрат на 61,5%.

Таблица 4.8. Параметры синтеза

Свойство	Значение
Цель оптимизации	Скорость
Качество оптимизации	Высокое
Глобальная цель оптимизации	Максимальная задержка
Стиль реализации	Полностью параллельная
Извлечение ОП	Не выбрано
Извлечение ПЗУ	Не выбрано
Извлечение регистра сдвига	Не выбрано
Использование блоков цифровой обработки сигнала	Нет
Перемещение первой стадии триггера	Не выбрано
Перемещение последней стадии триггера	Не выбрано
Пакет регистров ввода-вывода	Нет
Объединение таблиц соответствия	Нет
Уменьшение набора элементов управления	Нет
Оптимизация конкретных примитивов	Не выбрано

Таблица 4.9. Результаты аппаратного моделирования (свертка для модулей произвольного вида)

m	Задержка, нс	Таблицы соответствия слайсов	Занятые слайсы
3	7,949	49	22
5	9,770	106	44
7	8,188	51	24
11	12,377	268	98
13	13,561	264	98
17	**19,518**	384	214
19	16,191	489	208
23	18,568	477	266
29	15,005	380	162
31	18,411	277	152
37	18,409	468	218
Сумма		3213	1506

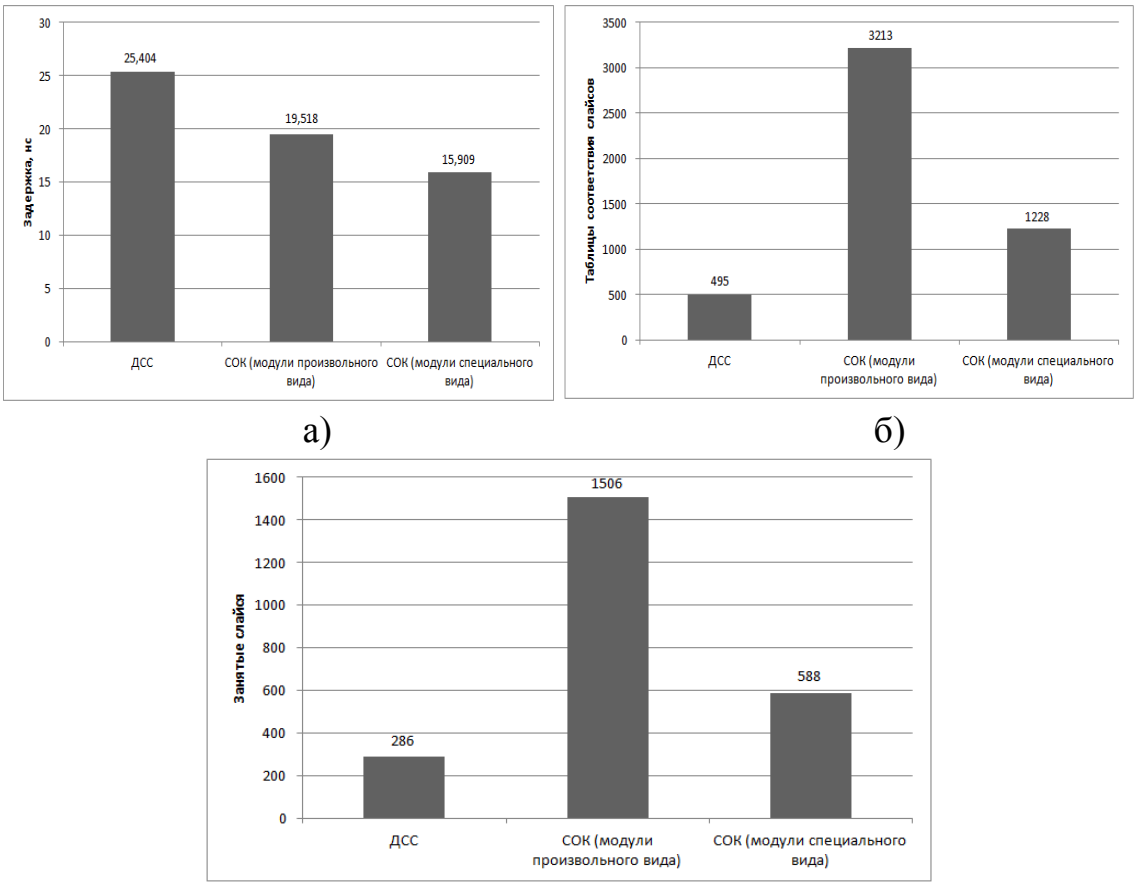
Таблица 4.10. Результаты аппаратного моделирования (свертка для модулей специального вида)

m	Задержка, нс	Таблицы соответствия слайсов	Занятые слайсы
2^5-1	11,757	159	49
2^7-1	12,345	251	122
2^8-1	12,988	270	125
2^9-1	13,230	311	150
2^9	**15,909**	**237**	**142**
Сумма		1228	588

Для аппаратного моделирования прямого и обратного преобразования использовалась целевая плата – Artix7 XC7A200T.

Результаты моделирования блока прямого преобразования из позиционной системы счисления в СОК для предложенной архитектуры СНС и известной архитектуры из [109] представлены в таблице 4.11. Операция прямого преобразования в предложенной архитектуре работает на 92,6 % быстрее и

требует в среднем на 98% меньше аппаратных затрат, чем известная архитектура из работы [109].



а) б)

в)

Рисунок 4.18. Диаграммы временных и аппаратных затрат пространственной свертки в двоичной системе счисления (ДСС), в СОК с использованием набора модулей произвольного вида и в СОК с использованием модулей специального вида: а) задержка устройства; б) количество таблиц соответствия слайсов; в) количество занятых слайсов.

Результаты моделирования блока обратного преобразования из позиционной системы счисления в СОК для предложенной архитектуры СНС и известной архитектуры из [109] представлены в таблице 4.12. Операция обратного преобразования в предложенной архитектуре работает на 61,8% быстрее и требует на 61% меньше аппаратных затрат, чем известная архитектура из работы [109].

Таблица 4.11. Результаты аппаратного моделирования (прямое преобразование из ПСС в СОК)

	Специальный набор модулей	Произвольные модули
Задержка, нс	**16,191**	219,412
Таблицы соответствия слайсов	**380**	21374
Занятые слайсы	**160**	6751

Таблица 4.12. Результаты аппаратного моделирования (обратное преобразование из СОК в ПСС)

	Специальный набор модулей	Произвольные модули
Задержка, нс	**29,081**	37,022
Таблицы соответствия слайсов	**2067**	3920
Занятые слайсы	**611**	1046

Таким образом, предложенная в данной главе архитектура СНС построена исходя из минимизации временных затрат. Для этого была проведена оценка времени работы СНС в зависимости от ее конфигурации и размера входного изображения. Результаты моделирования показали, что с уменьшением размера масок фильтров, разрешения входного изображения и увеличением шага вычислений временные затраты сокращаются. Наилучший результат достигается при использовании третьей конфигурации сети и разрешения изображения 256×192.

Аппаратное моделирование показало, что использование модулей специального вида позволяет увеличить быстродействие устройства и сократить аппаратные затраты.

Сравнивая полученные результаты в ходе данного исследования при построении нейронной сети и ее обучения с результатами из [109] можно сделать вывод, что предлагаемая архитектура СНС дает наилучшие результаты, за счет

модифицирования операций прямого и обратного преобразований в СОК. Прямое преобразование показывает лучшие результаты за счет применения специального набора модулей, обратное преобразование показывает лучшие результаты за счет применения специального набора модулей и не использования вложенной СОК.

4.3 Оценка аппаратурных затрат и быстродействия архитектуры сверточной нейронной сети с вычислениями в СОК

В таблице 4.13 приведены результаты сравнительного анализа аппаратурных затрат и быстродействия предложенной и известной [109] архитектуры СНС, использующих вычисления в СОК.

Анализ данных из таблицы 4.13 свидетельствует о том, что предложенная архитектура СНС использующая в своих вычислениях модули специального вида требует существенно меньших аппаратных затрат по сравнению с синтезом аналогичной архитектуры с произвольным набором модулей СОК. Переход к обработке в СОК с модулями специального вида позволил уменьшить аппаратурные затраты просмотровых таблиц более чем на 20% и занятых слайсов более чем на13% в сумматоре СОК, уменьшить количество занятых слайсов на 99% в блоке прямого преобразователя, уменьшить аппаратурные затраты просмотровых таблиц на 65% и занятых слайсов на 76% в блоке обратного преобразователя по сравнению с аналогичными показателями при использовании произвольного набора модулей.

Оценка быстродействия устройств обработки сигналов в СНС с модулями СОК произвольного вида показала, что устройства на основе такого набор модулей уступают в быстродействии аналогичным устройствам на основе СОК с модулями специального вида в блоке прямого преобразования на 96%, в блоке обратного преобразования на 65%.

Таблица 4.13. Сравниваемые показатели разработанной и известной архитектуры [109]

Функциональные блоки	Показатели	Известная архитектура [109]	Предложенная архитектура
Диапазон, бит		103	63
Сумматор СОК	Задержка, нс	7,453	**8,561**
	Просмотровые LUT-таблицы, шт	479	376
	Занятый слайсы, шт	197	160
Прямое преобразование	Задержка, нс	450,238[1]	**13,510**
	Просмотровые таблицы, шт	Превышение ресурсов целевой платы	583
	Занятый слайсы, шт	40042[1]	219
Обратное преобразование	Задержка, нс	80,546	**28,147**
	Просмотровые таблицы, шт	22120	5474
	Занятый слайсы, шт	6773	1591

4.4 Выводы по четвертой главе

1. В главе показана разработка СНС для распознавании образов, которая состоит из четырех слоев и использует фильтры Собеля для извлечения признаков из изображений. Проведено моделирование с целью оценки временных затрат работы СНС, основанных на различных конфигурациях сети (разные размеры масок фильтров, шаг вычислений, разрешение входного изображения). Установлено, что при уменьшении разрешения исходного изображения и размера маски фильтров временные затраты сокращаются. Показано, что наилучшее быстродействие достигается при использовании разрешения 256×192.

2.Проведено моделирование обучения разработанной СНС. Установлено, что оптимальная аппаратная конфигурации, дающая наилучшее количество распознанных изображений достигается при использовании 8 фильтров для операции свертки, 10 нейронов третьего слоя и 16 разрядов дробной части весовых коэффициентов третьего и четвертого слоев.

[1] Данные значения были получены путем анализа каждого вычислительного канала СОК в отдельности.

3.Проведено моделирование с целью сравнения работы схемы пространственной свертки в двоичной системе счисления, в СОК с использованием модулей специального вида и в СОК с использованием набора модулей произвольного вида. Установлено, что использование модулей специального вида позволяет ускорить работу блоков прямого и обратного преобразований соответственно на 92,6% и 61,8% по сравнению с использованием модулей произвольного вида при сокращении аппаратных затрат на 98% и 61% соответственно. Полученный результат достигается за счет использования модулей специального вида 2^n и 2^n-1, а также исключение избыточных преобразований во вложенных СОК.

Заключение

В рамках диссертационной работы реализованы подходы к выполнению операций цифровой фильтрации, лежащих в основе систем, базирующихся на интеграции системы остаточных классов и искусственной нейронной сети, включая цифровые системы видеонаблюдения. Решена актуальная задача поиска подходов к повышению скорости работы сверточной нейронной сети, которая выполняет операцию распознавания образов, за счет использования системы остаточных классов. Получены новые технологии оптимизации выполнения операций цифровой фильтрации, направленные на увеличение скорости работы системы.

Проведенное в диссертации исследование, а также результаты экспериментов и моделирование дали возможность получить следующие основные научные и практические результаты.

1. Разработана математическая модель фильтра с конечной импульсной характеристикой, основанная на вычислениях в системе остаточных классов, которая по сравнению с известными, позволяет увеличить скорость фильтрации на 93%.

2. Разработана математическая модель ошибок, возникающая при цифровой обработке сигналов с вычислениями в системе остаточных классов, позволяющая получать корректные результаты обработки сигнала в процессе цифровой фильтрации. Исследован вопрос о достаточности динамического диапазона системы остаточных классов, необходимого для процесса цифровой обработки изображений. Разработан критерий определения диапазона системы остаточных классов, зависящий от разрядности обрабатываемого изображения и коэффициентов маски фильтра, позволяющий избежать ошибок обработки, возникающих из-за переполнения динамического диапазона вычислительной системы.

3. Разработан новый численный метод для процесса цифровой фильтрации изображений, основанный на вычислениях в системе остаточных

классов, позволяющий сократить время необходимое на выполнение математических операций в ходе обработки. Разработана новая схема процесса фильтрации изображений, позволяющая получать корректные результаты обработки изображений для любых размеров масок и коэффициентов фильтра.

4. Исследован вопрос о влиянии разрядности коэффициентов фильтра при вейвлет-обработки изображений, что позволило установить, что пригодное для практических целей изображение можно получить при снижении разрядности до 12 бит, визуально неотличимое по качеству от 28 бит.

5. Разработана архитектура сверточной нейронной сети с вычислениями в системе остаточных классов, использующая модули специального вида и позволяющая увеличить скорость выполнения прямого и обратного преобразования. Показано преимущество разработанной архитектуры на основе сравнительного анализа работы модулярных сумматоров и устройств прямого и обратного преобразований по сравнению с известными архитектурами. Прямое преобразование данных из позиционной системы счисления в систему остаточных классов для разработанной архитектуры сверточной нейронной сети работает в 21 раз быстрее и требует в 56 раз меньше аппаратных затрат, чем известная архитектура сверточной нейронной сети. Обратное преобразование в разработанной архитектуре сверточной нейронной сети работает на 25% быстрее, чем в известной архитектуре, и требует на 32,5% меньше аппаратных затрат.

6. Разработана сверточная нейронная сеть, состоящая из четырех слоев, способная решать задачу распознавания образов при использовании фильтров Собеля для извлечения признаков из изображений.

7. Проведено моделирование работы сверточной нейронной сети, основанное на получение наименьших временных затрат с использованием различных конфигураций сверточной нейронной сети. Установлено, что при уменьшении разрешения исходного изображения и размера маски фильтров временные затраты сокращаются. Показано, что наилучшее быстродействие достигается при использовании разрешения 256×192.

8. Проведено моделирование обучения разработанной сверточной нейронной сети, основанное на получении наилучшего количества распознанных изображений. Установлено, что оптимальная аппаратная конфигурации, дающая наилучшее количество распознанных изображений достигается при использовании 8 фильтров для операции свертки, 10 нейронов третьего слоя и 16 разрядов дробной части весовых коэффициентов третьего и четвертого слоев.

9. Проведено моделирование, основанное на сравнении работы схемы пространственной свертки в двоичной системе счисления и СОК с использованием модулей специального вида и с использованием набора модулей произвольного вида. Установлено, что использование модулей специального вида позволяет ускорить работу блоков прямого и обратного преобразований соответственно на 92,6% и 61,8% по сравнению с использованием модулей произвольного вида при сокращении аппаратных затрат на 98% и 61% соответственно. Полученный результат достигается за счет использования модулей специального вида 2^n и 2^n-1, а также исключение избыточных преобразований во вложенных СОК.

Обозначения и сокращения

ДСС – двоичная система счисления

ИНС – искусственная нейронная сеть

КТО – Китайская теорема об остатках

ПЛИС – программируемая логическая интегральная схема

ПСС – позиционная система счисления

СОК – система остаточных классов

СНС – сверточная нейронная сеть

ЦОИ – цифровая обработка изображений

ЦОС – цифровая обработка сигналов

ASIC – специализированная интегральная схема

FPGA – программируемые вентильные матрицы

LUT – look-up table, просмотровая таблица

Список литературы

1. Айфичер, Э. Цифровая обработка сигналов. Практический подход. 2 изд. Пер. с англ. / Э. Айфичер, Б. Джервис. – М.: Вильямс, 2004. – 992 с.

2. Акушский, И.Я. Машинная арифметика в остаточных классах / И.Я. Акушский, Д.И. Юдицкий. – М.: Советское радио, 1968. – 440 с.

3. Альгин, О.Г. Особенности проектирования цифровых систем видеонаблюдения / О. Г. Альгин / Актуальные вопросы эксплуатации систем охраны и защищенных телекоммуникационных систем: материалы Всероссийской научно-практической конференции. – Тамбов, 2014. – С. 13-14.

4. Бабаев, В. Централизованная система управления IP-видеонаблюдения / В. Бабаев // Первая миля. – 2016. – №4(57). – С. 62-63.

5. Балухто, А.Н. Нейросетевая фильтрация и сегментация цифровых изображений / А. Н. Балухто, Л. Е. Назаров // Нейрокомпьютеры в прикладных задачах обработки изображений. – 2007. – № 25. – С. 7-24.

6. Белодедов, М.В. Методы проектирования цифровых фильтров: Учебное пособие / М.В. Белодедов. – Волгоград: Издательство Волгоградского государственного университета, 2004. – 60 с.

7. Блаттер, К. Вейвлет-анализ. Основы теории / К. Блаттер. – М.: Техносфера, 2006. – 272 с.

8. Воробьёв, В.И. Теория и практика вейвлет-преобразования / В.И. Воробьёв, В.Г. Грибунин. – СПб.: ВУС, 1999. – 204 с.

9. Галушкин, А.И. Нейрокомпьютеры для обработки изображений / А.И. Галушкин, Н.С. Томашевич, Е.И. Рябцев // Нейрокомпьютеры в прикладных задачах обработки изображений. – 2007. – № 25. – С. 74-109.

10. Ганьжа, Д. Видеонаблюдение не только для обеспечения безопасности / Д. Ганьжа // Журнал сетевых решений LAN. – 2016. – № 6. – С. 10-14.

11. Гонсалес, Р. Цифровая обработка изображений. Издание 3-е, исправленное и дополненное / Р. Гонсалес, Р. Вудс. – М.: Техносфера. 2012. – 1104 с.

12. Гоулд, Б. Цифровая обработка сигналов / Б. Гоулд, Ч. Рэйдер; пер. с англ. под ред. А.М. Трахтмана. – М.: Сов. радио, 1973. – 438 с.

13. Гусев, А.Л. Функциональная предобработка входных сигналов нейронной сети / А.Л. Гусев, Ф.М. Черепанов, Л.Н. Ясницкий // Нейрокомпьютеры: разработка, применение. – 2013. – №5. – С. 19-21.

14. Дамьяновски, В. CCTV. Библия охранного телевидения: пер. с англ. / В. Дамьяновски. – Москва: Ай-Эс-Пресс, 2003. – 344 с.

15. Добеши, И. Десять лекций по вейвлетам / И. Добеши. – Ижевск: НИЦ «Регулярная и хаотическая динамика», 2001. – 464 с.

16. Дремин, И.М. Вейвлеты и их использование / И.М. Дремин, О.В. Иванов, В.А. Нечитайло // Успехи физических наук. – 2001. – № 5. – С. 465-501.

17. Дьяконов, В.П. Вэйвлеты. От теории к практике / В.П. Дьяконов. – М.: СОЛОН-Р, 2002. – 446 с.

18. Желтов, С.Ю. Обработка и анализ изображений в задачах машинного зрения / С. Ю. Желтов. – М.: Физматкнига, 2010. – 672 с.

19. Зайцев, С.В. Обработка изображений при распознавании образов в видеонаблюдении / С.В. Зайцев, П.И. Карасев, Ю.А. Губсков / Охрана, безопасность, связь: материалы международной научно-практической конференции. – Воронеж. – 2015. – С. 168-171.

20. Исупов, К.С. Метод выполнения немодульных операций в системе остаточных классов на основе интервальных позиционных характеристик / К.С. Исупов // Фундаментальные исследования. – 2013. – № 4-3. – С. 566-570.

21. Как выбрать камеру видеонаблюдения. Статья. – Режим доступа: http:www.dgsv.ruArticlesdigital_processing_1.html

22. Калита, Д.И. Применение системы остаточных классов в энергосберегающих приложених цифровой обработки сигналов / Д.И. Калита, Н.И. Червяков / Инфокоммуникационные технологии в науке, производстве и

образовании (Инфоком-6): материалы Шестой международной научно-технической конференции. – Ставрополь: 2014. – С. 227-235.

23. Калита, Д.И. Анализ производительности КИХ фильтров реализованных в ПСС и СОК / Д.И. Калита, Н.И. Червяков / Инновации, качество и сервис в технике и технологиях: материалы 4-ой Международной научно-практической конференции. – Курск , 2014. – С. 259-264.

24. Калита, Д.И. Техника изоморфизма в архитектуре параллельного КИХ фильтра / Д.И. Калита, Н.И. Червяков / Наука, образование, общество: проблемы и перспективы развития: материалы Международной научно-практической конференции. – Тамбов, 2014. – С. 93-96.

25. Калита, Д.И. Аналитический обзор методов распознавания образов / Д.И. Калита, Т.Р. Раджабов / Основные направления развития научного потенциала в свете современных исследований: теория и практика: материалы одиннадцатой международной заочной научной конференции. – Ставрополь, 2017. – С. 235-238.

26. Калмыков, И.А. Математические модели нейросетевых отказоустойчивых вычислительных средств, функционирующих в полиномиальной системе классов вычетов / И.А. Калмыков. – М.: ФИЗМАТЛИТ, 2005. – 276 с.

27. Калмыков, И.А. Метод уменьшение диапазона представления данных в модулярных цифровых фильтрах / И.А. Калмыков, А.В. Велигоша, А.В. Гапочкин и др. // Теория и техника радиосвязи. – 2016. – №1. – С. 52-57.

28. Качановский, Ю.П. Предобработка данных для обучения нейронной сети / Ю.П. Качановский, Е.А. Коротков // Фундаментальные исследования. – 2011. – №12. – С. 117-120.

29. Косовская, Т.М. Иерархическое описание классов и нейросетевое распознавание сложных объектов / Т.М. Косовская, А.В. Тимофеев // Нейрокомпьютеры: разработка, применение. – 2007. – №6. – С. 30-33.

30. Крисилов, В.А. Методы ускорения обучения нейронных сетей / В.А. Крисилов, Д.Н. Олешко. – М.: Гардарика, 2005. – 1042 с.

31. Кузнецов, О.П. Псевдооптические нейронные сети – прямолинейные модели / О.П. Кузнецов // Автоматика и телемеханика. – 1996. – №12. – С. 160-172.

32. Курячий, М.И. Цифровая обработка сигналов: учеб. пособие для вузов / М.И. Курячий. – Томск: Томск. гос. Университет систем упр. и радио-электроники, 2009. – 190 с.

33. Лемешко К.Г. Системы автоматизированного распознавания объектов через видеонаблюдение / К. Г. Лемешко // Форум молодых ученых. – 2016. – №4(4). – С. 556-558.

34. Малла, С. Вейвлеты в обработке сигналов / С. Малла. – М.: Мир, 2005. – 671 с.

35. Мезенцева, О.С. Влияние предобработки изображений на качество обучения нейронной сети для их распознавания / О.С. Мезенцева, Н.А. Лагунов // Вестник СКФУ. – 2015. – №1. – С. 51-58.

36. Мезенцева, О.С. Фильтрация сильно зашумленных изображений / О.С. Мезенцева, А.А. Андреев // Обозрение прикладной и промышленной математики. – 2008. – Т.15. – №1. – С. 176-178.

37. Мерков, А. Распознавание образов. Введение в методы статистического обучения / А. Мерков. – М.: Едиториал, УРСС, 2011. – 256 с.

38. Назаров, Л.Е. Нейросетевые алгоритмы обнаружения, классификации и распознавания объектов на изображениях / Л.Е. Назаров, Н.С. Томашевич, А.Н. Балухто // Нейрокомпьютеры в прикладных задачах обработки изображений. – 2007. – № 25. – С. 25-54.

39. Никитин В. Особенности использования видеокомпресии MPEG-4 и ее применение в сетевом видеонаблюдении / В. Никитин, М. Ефимов // Алгоритм безопасности. – 2006. – №2. – С. 16-19.

40. Никколс, Д.Г. От нейрона к мозгу: Пер. с англ. Изд. 3-е. / Д.Г. Никколс, А.Р. Мартин, Б. Дж. Валлас, П.А. Фукс. – М.: Книжный дом «ЛИБРОКОМ», 2012. – 672 с.

41. Новиков И.Я. Теория всплесков / И. Я. Новиков, В. Ю. Протасов, М. А. Скопина. – М.: ФИЗМАТЛИТ, 2005. – 616 с.

42. Нуссбаумер Г. Быстрое преобразование Фурье и алгоритмы вычисления сверток. Пер. с англ. / Г. Нуссбаумер. – М.: Радио и Связь, 1985. – 248 с.

43. Оцоков Ш.А. Ускорение высокоточных вычислений за счет распараллеливания операций округления в комплексе систем счисления / Ш.А. Оцоков // Информационные технологии. – 2015. – Т. 21. – №5. – С. 352 – 356.

44. Потапов, А.А. Новейшие методы обработки изображений / А.А. Потапов, А.А. Пахомов, С.А. Никитин. – М.: Физматлит, 2008. – 496 с.

45. Р 78.36.002–99 «Выбор и применение телевизионных систем видеоконтроля»

46. Рассел, С. Искусственный интеллект: современный подход, 2-е изд.: Пер. с англ. / С. Рассел, П. Норвиг. – М.: Издательский дом «Вильямс», 2006. – 1408 с.

47. Сергиенко, А.Б. Цифровая обработка сигналов / А.Б. Сергиенко. – СПб.: Питер, 2003. – 604 с.

48. Смоленцев, Н.К. Основы теории вейвлетов. Вейвлеты в MATLAB / Н.К. Смоленцев. – М.: ДМК Пресс, 2005. – 304 с.

49. Стемпковский, А.Л. Особенности реализации устройств цифровой обработки сигналов в интегральном исполнении с применением модулярной арифметики / А.Л. Стемпковский, А.И. Корнилов, М.Ю. Семенов // Информационные технологии. – 2004. – № 2. – С. 2-9.

50. Суздальцев, В.А. Нейросетевой метод определения пространственных координат мобильного объекта при видеонаблюдении / В.А. Суздальцев, Р.Л. Чумарин / Инновационные технологии XXI века: материалы международной научно-практической конференции. – Казань, 2015. – С. 23-25.

51. Тархов, Д.А. Нейронные сети. Модели и алгоритмы. Книга 18 / Д.А. Тархов. – М.: Радиотехника, 2005. – 256 с.

52. Торгашев, В.А. Система остаточных классов и надежность ЦВМ / В.А. Торгашев. – М.: Советское радио, 1973. – 120 с.

53. Финько, О.А. Модулярная арифметика параллельных вычислений: монография / О.А. Финько. – М.: Институт проблем управления им. В.А. Трапезникова РАН, 2003. – 224 с.

54. Фомин, Я. А. Распознавание образов: теория и практика. Издание третье, дополненное / Я. А. Фомин. – М.: ФАЗИС, 2014. – 460 с.

55. Фрейзер, М. Введение в вэйвлеты в свете линейной алгебры / М. Фрейзер. – М.: БИНОМ. Лаборатория знаний, 2008. – 487 с.

56. Червяков, Н.И. Нейрокомпьютеры в остаточных классах / Н.И. Червяков, П.А. Сахнюк, А.В. Шапошников, А.Н. Макоха. – М.: Радиотехника, 2003. – 272 с.

57. Червяков, Н.И. О математических моделях фильтров для цифровой обработки изображений / Н.И. Червяков, П.А. Ляхов, Д.И. Калита, Н.В. Попова / Основные направления развития научного потенциала в свете современных исследований: теория и практика: материалы одиннадцатой международной заочной научной конференции. – Ставрополь, 2017. – С. 238-241.

58. Червяков, Н.И. Методы, алгоритмы и техническая реализация основных проблемных операций, выполняемых в системе остаточных классов / Н.И. Червяков // Инфокоммуникационные технологии. – 2011. – №4. – С. 4-12.

59. Червяков, Н.И. Принципы построения модулярных сумматоров и умножителей / Н.И. Червяков, И.В. Дьяченко // Сборник научных трудов Ставропольского государственного университета. – 2006. – №. 1. – С. 26-39.

60. Червяков, Н.И. Четырехточечный фильтр Добеши / Н.И. Червяков, Д.И. Калита, Е.С. Карнаухова, М.А. Дерябин / Параллельная компьютерная алгебра и ее приложения в новых инфокоммуникационных системах: материалы I международной научной конференции. – Ставрополь, 2014. – С. 440-444.

61. Червяков, Н.И. Развитие методов быстрого вейвлет-преобразования с помощью фильтров Добеши / Н.И. Червяков, Ю.В. Кондрашов // Научные

ведомости Белгородского государственного университета. Серия: Экономика. Информатика. – 2009. – №15 (70). – С.112-117.

62. Червяков, Н.И. Реализация КИХ-фильтров в системе остаточных классов / Н.И. Червяков, П.А. Ляхов // Нейрокомпьютеры: разработка, применение. – 2012. – №5. – С. 15-24.

63. Червяков, Н.И. О выборе диапазона системы остаточных классов для цифровой обработки изображений / Н.И. Червяков, П.А. Ляхов, Д.И. Калита // Инфокоммуникационные технологии. – 2016. – Т. 14. – №2. – С. 111-122.

64. Червяков, Н.И. Влияние ограничения разрядности коэффициентов фильтра на качество вейвлет-обработки изображений в оттенках серого / Н.И. Червяков, П.А. Ляхов, Д.И. Калита, К.С. Шульженко // Наука. Инновации. Технологии. – 2016. – №2. – С. 61-76.

65. Червяков, Н.И. Принцип сжатия изображений на основе дискретного вейвлет-преобразования / Н.И. Червяков, П.А. Ляхов, Д.И. Калита, К.С. Шульженко // Наука. Инновации. Технологии. – 2016. – №3. – С. 97-118.

66. Червяков, Н.И. Выбор оптимального набора модулей системы остаточных классов для повышения производительности фильтров с конечной импульсной характеристикой / Н.И. Червяков, П.А. Ляхов, Д.И. Калита // Информационные технологии. – 2015. – Т.21. – №12. – С.923-929.

67. Червяков, Н.И. Применение сумматоров с параллельно-префиксной архитектурой для перевода чисел из двоичной системы счисления в систему остаточных классов / Н.И. Червяков, П.А. Ляхов, Н.Ф. Семенова, М.В. Валуева // Нейрокомпьютеры: разработка, применение. – 2016. – №10. – С. 31-39.

68. Червяков, Н.И. Архитектура сверточной нейронной сети с вычислениями в системе остаточных классов с модулями специального вида / Н.И. Червяков, П.А. Ляхов, Д.И. Калита, М.В. Валуева // Нейрокомпьютеры: разработка, применение. – 2017. – №1. – С. 3-15.

69. Червяков, Н.И. Модулярные параллельные вычислительные структуры нейропроцессорных систем / Н.И. Червяков, П.А. Сахнюк, А.В. Шапошников, С.А. Ряднов. – М.: ФИЗМАТЛИТ, 2003. – 288 с.

70. Хайкин, С. Нейронные сети: полный курс, 2-е издание / С. Хайкин. – М.: Вильямс, 2006. – 1104 с.

71. Afsheh, A. An improved reverse converter for moduli set $\left(2^n-1,2^n,2^n+1\right)$/ A. Afsheh, A. Mojoodi // ISCIT 2010 International Symposium on Communications and Information Technologies Proceedings. – 2010. – №5 – P. 928 – 933.

72. Ammar, A. A secure image coding scheme using residue number system / A. Ammar, A. Kabbany, M. Youssef, A. Amam // Proc. National Radio Sci. Conf. – 2001. – № 2. – P. 399 – 405.

73. Antao, S.F. RNS based Elliptic Curve Point Multiplication for Massive Parallel Architectures / S.F. Antao, J.C. Bajard, L. Sousa // Computer Journal. – 2012. – № 5. – Vol.55. – P. 629 – 647.

74. Bailey, D.G. Design for embedded image processing on FPGA / D.G. Bailey. – John Wiley & Sons (Asia) Pte Ltd, 2011. – 497 p.

75. Bovik, A. Handbook of Image and Video Processing / A. Bovik. – Second Edition Elsevier Academic Press, 2005. – 1429 p.

76. Cao, B. A new efficient reverse converter for the 4-moduli set $\left\{2^n-1,2^n,2^n+1,2^{2n}+1\right\}$ / B. Cao, C.H. Chang, T. Srikanthan // IEEE Trans. on Circuits and Systems-I: Fundamental Theory and Applications. – 2014. – №2. – Vol. 8. – P. 328 – 332.

77. Cao, B. A residue-to-binary converter for a new five-moduli set / B. Cao, C. Chang, T. Srikanthan // IEEE Trans. on Circuits and Systems-I: Regular Papers. – 2007. – №5. – Vol. 54. – P. 1041 – 1049.

78. Cardarilli, G.C. Residue number system for low-power DSP applications / G.C. Cardarilli, A. Nannarelli, M. Re // Proc. 41st Asilomar Conf. Signals, Syst., Comput. – 2007. – Vol. 57. – P. 1412 – 1416.

79. Chang, C.-H. Residue Number Systems: A New Paradigm to Datapath Optimization for Low-Power and High-Performance Digital Signal Processing Applications / C.-H. Chang, A.S. Molahosseini, A.A.E. Zarandi, T.F. Tay // IEEE Circuits and Systems Magazine. – 2015. – Vol. 15. – № 4. – P. 26 – 44.

80. Chervyakov, N.I. An Approximate Method for Comparing Modular Numbers and its Application to the Division of Numbers in Residue Number Systems / N.I. Chervyakov, M.G. Babenko, P.A. Lyakhov, I.N. Lavrinenko // Cybernetics and Systems Analysis. – 2014. – №50. – Vol.6. – P. 977 – 984.

81. Chervyakov, N.I. High-speed smoothing filter in the residue number system / N.I. Chervyakov, P.A. Lyakhov, A.S. Ionisyan, M.V. Valueva / 2016 3rd Internation Processing, Data Mining, amd Wireless Communications, DIPDMWC 2016. – Moscow, 2016. – C. 121 – 126.

82. Chervyakov, N.I. Digital filtering of images in a residue number system using finite-field wavelets / N.I. Chervyakov, P.A. Lyakhov, M.G. Babenko // Automatic Control and Computer Sciences. – 2014. – № 3. – P. 180–189.

83. Chervyakov, N.I. An efficient method of error correction in fault-tolerant modular neurocomputers / N.I. Chervyakov, P.A. Lyakhov, M.G. Babenko, A.I. Garyanina, I.N. Lavrinenko, A.V. Lavrinenko // Neurocomputing. – 2016. – Vol. 205. – P. 32 – 44.

84. Chervyakov, N.I. Effect of RNS dynamic range on grayscale image filtering / N.I. Chervyakov, P.A. Lyakhov, D.I. Kalita, K.S. Shulzhenko // 2016 XV International Symposium Problems of Redundancy in Information and Control Systems (REDUNDANCY). – St. Petersburg, 2016. – P. 33 – 37.

85. Chervyakov, N.I. Effect of RNS moduli set selection on digital filter performance for satellite communications / N.I. Chervyakov, P.A. Lyakhov, D.I. Kalita, K.S. Shulzhenko / 2015 International Siberian Conference on Control and Communications (SIBCON). – Omsk, 2015. – P. 1 – 7.

86. Fan, J. Human Tracking Using Convolutional Neural Networks / J. Fan, W. Xu, Y. Wu // IEEE Transactions on Neural Networks. – 2010. – №. 10. – Vol. 21. – P. 1610 – 1623.

87. Farabet, C. Hardware accelerated convolutional neural networks for synthetic vision systems / C. Farabet, B. Martini, P. Akselrod, S. Talay, Y. LeCun, E. Culurciello / Int'l Symp. on Circuits and Systems (ISCAS2010). – Paris, 2010. – P. 257 – 260.

88. Goh, V.T. Multiple Error Detection and Correction Based on Redundant Residue Number Systems / V.T. Goh, M.U. Siddiqi // IEEE Transactions on Communications. – 2008. – № 3. – Vol. 56. – P. 325 – 330.

89. Goswami, J.C. Fundamentals of Wavelets. Theory, Algorithms, and Applications / J.C. Goswami, A.K. Chan // Wiley. – 2011. – Vol.2. – 359 p.

90. Hariri, A. A new high dynamic range moduli set with efficient reverse converter / A. Hariri, K. Navi, R. Rastegar // Computers & Mathematics with Applications Journal. – 2008. – Vol. 55. – P. 660 – 668.

91. Hernandez, N.R. Bits planes technique for digital image processing / N.R. Hernandez, J.L.R. Quirarte / 5th International Conference on Electrical Engineering, Computing Science and Automatic Control. – Mexico, 2008. – P. 186 – 191.

92. Hiasat, A.A. VLSI implementation of new arithmetic residue to binary decoders / A.A. Hiasat // IEEE Trans. on VLSI Systems. – 2005. – Vol. 13. – P. 153 – 158.

93. Hung, C.Y. An approximate sign detection method for residue numbers and its application to RNS division / C.Y. Hung, B. Parhami // Computers &Mathematics with Applications. – 1994. – №4. – Vol. 27. – P. 23 – 35.

94. Huynh-Thu, Q. Scope of validity of PSNR in image/video quality assessment / Q. Huynh-Thu, M. Ghanbari // Electronics Letters. – 2008. – №13. – Vol. 44. – P. 800–801.

95. Ingle, V. K. Digital Signal Procesing Using MATLAB / V.K. Imgle, J.K. Proakis / ser. BookWare Companion Series, New York, Madrid: Brooks/Cole Publishing Company. – Stamford, 1999. – 218 p.

96. Juxian, M. Based on the fourier transform and the wavelet transformation of the digital image processing / M. Juxian / 2012 International Conference on Computer Science and Information Processing (CSIP). – Xian, Shaanxi, 2012. – P. 1232 – 1234.

97. Khan, H. Efficient eyes and mouth detection algorithm using combination of viola jones and skin color pixel detection / H. Khan, M. Abdulla, Bin Zainal Shamian // International Journal of Engineering and Applied Sciences. – 2013. – №4. – Vol. 3. – P. 234 – 245.

98. Kogge, P.M. A Parallel Algorithm for the Efficient Solution of a General Class of Recurrence Equations / P.M. Kogge // IEEE Transaction on computers. – 1973. – №8. – Vol. 22. – P. 786 – 793.

99. Lecun, Y. Gradient-based learning applied to document recognition / Y. Lecun, L. Bottou, Y. Bengio, P. Haffiner // Proc. of the IEEE. – 1998. – №11. – Vol.86. – P. 2278 – 2324.

100. Meyer-Baese, U. Digital Signal Processing with Field Programmable Gate Arrays / U. Meyer-Baese / Springer-Verlag Berlin Heidelberg. – Berlin, 2001. – 422 p.

101. Mitra, S. Digital Signal Processing: A Completed-Based Approach / S. Mitra / 3 edition. UCSB: Mc Graw Hill. – Santa Barbara, 2006. – 960 p.

102. Mohan, P.V.A. RNS-to-binary converter for a new three moduli set $\{2^{n+1}-1,2^n,2^n-1\}$ / P.V.A Mohan // IEEE Trans. on Circuits and Systems-II: Express Briefs. – 2007. – Vol. 54. – P. 775 – 779.

103. Mohan, P.V.A. RNS-to-binary converters for two four-moduli sets $\{2^n-1,2^n,2^n+1,2^{n+1}-1\}$ and $\{2^n-1,2^n,2^n+1,2^{n+1}+1\}$ / P.V.A. Mohan, A.B. Premkumar // IEEE Trans. on Circuits and Systems-I: Regular Papers. – 2007. – Vol. 54. – P. 1245 – 1254.

104. Mollahosseini, A.S. A new five moduli set for efficient hardware implementation of the reverse converter / A.S. Molahosseini, C. Dadkhan, K. Navi // IEICE Electronics Express. – 2009. – Vol. 6. – P. 1006 – 1012.

105. Mollahosseini, A.S. Efficient reverse converter designs for the new 4-moduli sets $\{2^n-1,2^n,2^n+1,2^{2n+1}-1\}$ and $\{2^n-1,2^n+1,2^{2n},2^{2n}+1\}$ based on new CRTs / A.S. Molahosseini, K. Navi, C. Dadkhan, O. Kavehei, S. Timarchi // IEEE Trans. on Circuits and Systems-I: Regular Papers. – 2010. – Vol. 57. – P. 823 – 835.

106. Mollahosseini, A.S. A new residue to binary converter based on mixed-radix conversion / A.S. Molahosseini, K. Navi, M.K. Rafsanjani // 3rd International Conference on Information and Communication Technologies: From Theory to Applications (ICTTA). – 2008. – Vol. 23. – P. 1 – 6.

107. Mollahosseini, A.S. Research Challenges in Next-Generation Residue Number System Architectures / A.S. Molahosseini, S. Sorouri, A.A. Zarandi / 7th International Conference on Computer Science & Education. – Melbourne, 2012. – P. 1658–1661.

108. Mollahosseini, A.S. Embedded systems dasing with special arithmetic and number systems / A.S. Molahossini, L.S. Sousa, C.H. Chang // Springer. – 2017. – № 1. – 388 p.

109. Nakahara, H. A deep convolutional neural network based on nested residue number system / H. Nakahara, T. Sasao / 25th International Conference on Field Programmable Logic and Applications (FPL). – London, 2015. – P. 1 – 6.

110. Omondi, A. Residue Number Systems: Theory and Implementation / A. Omondi, B. Premkumar / Imperial College Press. – Singapore, 2007. – 296 p.

111. Parhami, B. Computer Arithmetic: Algorithms and Hardware Designs / B. Parhami. – New York, Oxford University Press, 2010. – 641 p.

112. Patronik, P. Energy-Effcient Constant-Coeffcient FIR Filters Using Residue Number System / P. Patronik, K. Berezowski, S.J. Piestrak, J. Biernat // Low Power Electronics and Design (ISLPED) International Symposium, IEEE. – Taipei, Taiwan, 2011. – P. 385 – 390.

113. Peemen, M. Memorycentric accelerator design for convolutional neural networks / M. Peemen, A.A. Setio, B. Mesman, H. Corporaal // 31st International Conference on Computer Design (ICCD2013). – Asheville, NC, 2013. – P.13 –19.

114. Piestrak, S. J. A high-speed realization of a residue to binary number system converter / S.J. Piestrak // IEEE Trans. on Circuits and Systems-II: Analog and Digital Signal Processing. – 1995. – Vol. 42. – P. 661 – 663.

115. Salomon, D. Data Compression / D. Salomon // Springer-Verlag. – London, 2007. – 1092 p.

116. Sankaradas, M. A massively parallel coprocessor for convolutional neural networks / M. Sankaradas, V. Jakkula, S. Gadami, S. Chakradhar, I. Durdanovic, E. Cosatto, H.P. Graf / 20th IEEE International Conference on Application-specific Systems, Architectures and Processors (ASAP2009). – Boston, 2009. – P.53 – 60.

117. Shahana, T.K. Performance Analysis of FIR Digital Filter Design: RNS Versus Traditional / T.K. Shahana, R.K. James, B.R. Jose, K.P. Jacob, S. Sasi / ISCIT 2007 International Symposium on Communications and Information Technologies Proceedings. – Sidney, 2007. – P.1 – 5.

118. Shih, F.I. Image processing and pattern recognition / F.I. Shih / Institute of Electrical and Electronics Engineers, Inc. – New Jersey, 2010. – 552 p.

119. Stamenković, N. Digital FIR Filter Architecture Based on the Residue Number System / N. Stamenković // Facta Universitatis, Ser.: Elec. Energ. – 2009. – №1. – Vol. 22. – P. 125 – 140.

120. Taleshmekaeil, D.K. Using residue number system for edge detection in digital images processing / D.K. Taleshmekaeil, H. Mohamamdzade, A. Mousavi / IEEE 3rd international conference on communication software and networks. – Xian, 2011. – P. 249–253.

121. Taleshmekaeil, D.K. The use of residue number system for improving the digital image processing / D.K. Taleshmekaeil, A. Mousavi / In Proc. of 10th International Conference on Signal Processing. – Pecin, 2010. – P. 775 – 780.

122. Tan L. Digital Signal Processing, Second Edition: Fundamentals and Applications / L. Tan , J. Jiang // Academic Press. – 2013. – № 34. – 876 p.

123. Vasalos, E. RNS Assisted Image Filtering and Edge Detection / E. Vasalos, D. Bakalis, H.T. Vergos / Digital Signal Processing (DSP), 2013 18th International Conference. – Fira, Santorini, 2013. – P. 1 – 6.

124. Vergos, H.T. On Modulo 2^n+1 Adder Design, H.T. / H.T. Vergos // IEEE Trnsactions on computers. – 2012. – №2. – Vol 61. – P. 173 – 186.

125. Wang, Z. Image quality assessment: from error visibility to structural similarity / Z. Wang // IEEE Transactions image processing. – 2004. – №4. – Vol 13. – P. 600–612.

126. Wang, W. A high-speed residue-to-binary converter for three-moduli $\left(2^k, 2^k-1, 2^{k-1}-1\right)$ RNS and a scheme for its VLSI implementation / W. Wang, M.N.

Swamy, M.O. Ahmad, Y. Wang // IEEE Trans. on Circuits and Systems-II: Analog and Digital Signal Processing. – 2000. – №12. – Vol. 47. – P. 1576–1581.

127. Wang, W. A study of the residue-to-binary converters for the three-moduli sets / W. Wang, M.N.S. Swamy, M.O. Ahmad, Y. Wang // IEEE Trans. Circuits Syst. I: Fundamental Theory Appl. – 2000. – Vol. 50. – P. 235 – 243.

128. Wang, W. RNS application for digital image processing / W. Wang, M.N.S. Swamy, M.O. Ahmad / Proceedings of the 4th IEEE International Workshop on System-on-Chip for Real-time Application. – Canada, 2004. – P. 77 – 80.

129. Younes, D. Comparative Study on Different Moduli Sets in Residue Number System / D. Younes, P.A. Steffan / Dept. of Microelectronics Brno University of Technology Brno, Czech Republic. – Brno, 2013. – P. 47 – 50.

130. Younes, D. Efficient image processing application using Residue Number System / D. Younes, P. Steffan / 20th International Conference on Mixed Design of Integrated Circuits and Systems. – Gdynia, 2013. – P. 20 – 22.

131. Zarei, B. Residue Number System for Low-Power DSP Applications / B. Zarei, V. Muthukkumarasay, Wu. Xin-Wen / 27th International Conference on Advanced Information Networking and Applications (AINA), IEEE. – Barselona, 2013. – P. 197 – 204.

132. Zhang, W. An efficient design of residue to binary converter for four moduli set $\{2^n-1,2^n+1,2^{2n}-2,2^{2n+1}-3\}$ based on new CRT-II / W. Zhang, P. Siy // Information Sciences Journal. – 2008. – Vol. 178. – P. 264 – 279.

133. Chang, C.-H. Residue Number Systems: A New Paradigm to Datapath Optimization for Low-Power and High-Performance Digital Signal Processing Applications / C.-H. Chang, A.S. Molahosseini, A.A.E. Zarandi, T.F. Tay // IEEE Circuits and Systems Magazine. – 2015. – №4. – Vol. 15. – P. 26 – 44.

ПРИЛОЖЕНИЕ 1

Программный комплекс моделирования цифровой фильтрации в системе остаточных классов с модулям специального вида, осуществляющая преобразование входного сигнала методом свертки с коэффициентами фильтра.

Программа состоит из 1 модуля, который является основным (модуль основной программы).

```
#include "stdafx.h"
#include <iostream>
#include "conio.h"
#include "math.h"

using namespace std;

int main()
{
 int h[10] ;
int x[10];
int y[10],y1[10],y2[10],y3[10],result[10];
int N=10,Diapazon=504,M1=72,M2=63,M3=56,k1=2,k2=7,k3=2,B1,B2,B3;
   setlocale(LC_CTYPE, "Russian");
     cout<<"Введите 10 целых чисел входного сигнала\n";
for(int i=0;i<N;i++){
     cin>>x[i];}
```

```
for(int i=0;i<N;i++) {
     cout<<"\n x["<<i<<"] = "<<x[i];}
     cout<<"\nВведите 10 целых чисел обозначающих
коэффициенты\n";
for(int i=0;i<N;i++){
     cin>>h[i];}
for(int i=0;i<N;i++) {
     cout<<"\n h["<<i<<"] = "<<h[i];}


  for (int k = 0; k < N; k++)
  {
    y1[k] = (x[k]*h[k])% 7;
  }
  for (int l = 0; l < N; l++)
  {
    y2[l] = (x[l]*h[l])% 8;
  }
  for (int m = 0; m < N; m++)
  {
    y3[m] = (x[m]*h[m])% 9;
  }


  for(int i=1;i<7;i++) {
     if ((k1*i)%7==1) {
          B1=i*M1;
```

```
          break;}  }

  for(int q=1;q<8;q++) {
     if ((k2*q)%8==1) {
          B2=q*M2;
          break;}}

  for(int t=1;t<9;t++) {
     if ((k3*t)%9==1) {
          B3=t*M3;
          break;}}

   for(int i=0;i<N;i++) {
     result[i]=(y1[i]*B1+y2[i]*B2+y3[i]*B3)%504;}

  cout<<"\n B1 = "<<B1;
      cout<<"\n B2 = "<<B2;
      cout<<"\n B3 = "<<B3;
 for(int i=0;i<N;i++) {
     cout<<"\n signal["<<i<<"] = "<<result[i];
 }        getch();
    return 0;
}
```

ПРИЛОЖЕНИЕ 2

Программный комплекс моделирования распознавания изображений с использованием сверточных нейронных сетей

Листинг СНС в Matlab

СНС

```
function out = neural_network_new( image)
%
img_3 =level_3( image, 3, 8, 3);
input = level_4_generate( img_3, 2);
out = myNeuralNetworkFunction_3(input);

end
```

Первый слой (свертка)

```
function img_out = level_3( img_in, step, count_filters, kernel)
%функция осуществляет операцию свертки
%img_in - исходное изображение формата RGB
%step - шаг вычислений
%count_filters - количество фильтров
%kernel - размер маски фильтра
%img_out - обработанный набор изображений

%вычисляем размер исходного изображения
 size_img_in = size(img_in);
%вычисляем размер обработанного изображения
```

```
 size_img_out_1 = fix(size_img_in(1)/step);
 size_img_out_2 = fix(size_img_in(2)/step);
%генерируем нулевой массив обработанных изображений
 img_out = double(zeros( size_img_out_1, size_img_out_2, count_filters));
%генерируем временный массив для промежуточных вычислений
 img_tmp = double(zeros( size_img_in(1)+kernel-1, size_img_in(2)+kernel-1, size_img_in(3)));
%вычисляем необходимый размер рамки из 0
 frm = fix(kernel/2);
%загружаем фильтры из файла
 %load filter_3_8 filters_3;
 filters_3 = double(zeros( kernel, kernel, size_img_in(3), count_filters));
 filters_3(:,:,1,1) = [1 2 1; 0 0 0; -1 -2 -1];
 filters_3(:,:,2,1) = [1 2 1; 0 0 0; -1 -2 -1];
 filters_3(:,:,3,1) = [1 2 1; 0 0 0; -1 -2 -1];
 filters_3(:,:,1,2) = [1 0 -1; 2 0 -2; 1 0 -1];
 filters_3(:,:,2,2) = [1 0 -1; 2 0 -2; 1 0 -1];
 filters_3(:,:,3,2) = [1 0 -1; 2 0 -2; 1 0 -1];
 filters_3(:,:,1,3) = [2 1 0; 1 0 -1; 0 -1 -2];
 filters_3(:,:,2,3) = [2 1 0; 1 0 -1; 0 -1 -2];
 filters_3(:,:,3,3) = [2 1 0; 1 0 -1; 0 -1 -2];
 filters_3(:,:,1,4) = [0 1 2; -1 0 1; -2 -1 0];
 filters_3(:,:,2,4) = [0 1 2; -1 0 1; -2 -1 0];
 filters_3(:,:,3,4) = [0 1 2; -1 0 1; -2 -1 0];
 filters_3(:,:,1,5) = [0 -1 -2; 1 0 -1; 2 1 0];
 filters_3(:,:,2,5) = [0 -1 -2; 1 0 -1; 2 1 0];
 filters_3(:,:,3,5) = [0 -1 -2; 1 0 -1; 2 1 0];
```

```
filters_3(:,:,1,6) = [-2 -1 0; -1 0 1; 0 1 2];
filters_3(:,:,2,6) = [-2 -1 0; -1 0 1; 0 1 2];
filters_3(:,:,3,6) = [-2 -1 0; -1 0 1; 0 1 2];
filters_3(:,:,1,7) = [-1 -2 -1; 0 0 0; 1 2 1];
filters_3(:,:,2,7) = [-1 -2 -1; 0 0 0; 1 2 1];
filters_3(:,:,3,7) = [-1 -2 -1; 0 0 0; 1 2 1];
filters_3(:,:,1,8) = [-1 0 1; -2 0 2; -1 0 1];
filters_3(:,:,2,8) = [-1 0 1; -2 0 2; -1 0 1];
filters_3(:,:,3,8) = [-1 0 1; -2 0 2; -1 0 1];
%добавляем  рамку из 0
 for n = 1:1:size_img_in(3)
       img_tmp(: ,: ,n) = frm_zeros(img_in(: ,: ,n),frm);
 end
%выполняем операцию свертки
for m = 1:1:count_filters
    for i=1:1: size_img_out_1
      for j=1:1: size_img_out_2
         img_out_tmp = 0;
          for n = 1:1:size_img_in(3)
            for k=1:1:kernel
               for l=1:1:kernel
                    img_out_tmp = img_out_tmp+img_tmp(step*(i-1)+k,step*(j-
1)+l,n)*filters_3(k,l,n,m);
               end
            end
            img_out(i,j,m) = img_out(i,j,m) + img_out_tmp ;
         end
```

```
        end
    end
 end

end
```

Второй слой (выбор максимальных элементов)

```
function img_4 = level_4_generate(  img_in, step)
%Функция выполняет операцию выбора максимального элемента
%img_in - исходный набор изображений
%step - шаг вычислений
%img_4 - обработанное изображение в виде вектора
%вычисляем размер исходного изображения
 size_img_in = size(img_in);
%вычисляем размер обработанного изображения
 size_img_out_1 = fix(size_img_in(1)/step);
 size_img_out_2 = fix(size_img_in(2)/step);
%генерируем нулевой двумерный массив обработанных изображений
img_out = double(zeros(  size_img_out_1, size_img_out_2, size_img_in(3)));
 %операция выбора максимального элемента по всем слоям
 for n = 1:1:size_img_in(3)
    for i=1:1: size_img_out_1
       for j=1:1: size_img_out_2
          for k=1:1:step
             for l=1:1:step
```

```
                if  img_out(i,j,n) < img_in(step*(i-1)+k,step*(j-1)+l,n)
                 img_out(i,j,n) = img_in(step*(i-1)+k,step*(j-1)+l,n);
                end
            end
        end
      end
    end
 end
%определяем размер обработанного изображения в виде вектора
size_w = fix(size_img_out_1*size_img_out_2);
%генерируем нулевой вектор
img_4 = zeros(size_w,1,'double');
count = 1;
%заполняем обработанное изображение в виде вектора
for i=1:1: size_img_out_1
      for j=1:1: size_img_out_2
         for n = 1:1:size_img_in(3)
           img_4(count) = img_4(count)+ img_out(i,j,n);
         end
         count = count +1;
      end
end
end
```

Третий и четвертый слои (нейронная сеть)

```
function [y1] = myNeuralNetworkFunction_3(x1)

%MYNEURALNETWORKFUNCTION neural network simulation function.

%

% Generated by Neural Network Toolbox function genFunction, 26-Feb-2017 22:02:56.

%

% [y1] = myNeuralNetworkFunction(x1) takes these arguments:

%   x = 1344xQ matrix, input #1

% and returns:

%   y = 8xQ matrix, output #1

% where Q is the number of samples.


%#ok<*RPMT0>


  % ===== NEURAL NETWORK CONSTANTS =====


  % Input 1

  x1_step1_xoffset                                                          =
[10560;6364;6310;6336;6555;6491;6746;6680;6508;6416;6622;6455;6591;6690;6559;
6539;6435;6520;6435;6438;6620;6709;6938;6874;7004;6999;7058;6898;6790;6771;6
922;6708;6523;6374;6454;6371;6253;6157;6212;6139;6340;6372;6414;0;0;0;0;0;0;0;0
;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;6384;0;0;0;0;0;0;0;0;0;
0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;6098;0;0;0;0;0;0;0;0;0;0;0
;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;6313;0;0;0;0;0;0;0;0;0;0;0;0;
0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;6523;0;0;0;0;0;0;0;0;0;0;0;0;0;0
;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;6643;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;
0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;6627;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0
;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;7401;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;
0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;7518;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0
;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;7661;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;
0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;7747;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0
```

;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;8121;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;217;180;8
1;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;7941;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;132;19
2;96;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;8254;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;116
;96;98;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;8302;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;1
05;102;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;8247;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0
;35;74;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;8352;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;
0;16;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;8286;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;
0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;8270;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0
;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;8323;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;
0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;8323;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0
;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;8450;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;
0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;8476;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0
;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;8538;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;
0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;8524;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0
;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;8606;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;
0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;8691;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0
;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;8728;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;
0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;8709;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0
;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;8577;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;
0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;8536;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0
;0];

x1_step1_gain=[0.000122070312500000;0.000228881835937500;0.000228881835937
500;0.000228881835937500;0.000228881835937500;0.000228881835937500;0.0002
28881835937500;0.000228881835937500;0.000228881835937500;0.0002288818359
37500;0.000228881835937500;0.000228881835937500;0.000228881835937500;0.00
0228881835937500;0.000228881835937500;0.000228881835937500;0.00022888183
5937500;0.000228881835937500;0.000228881835937500;0.000228881835937500;0.
000228881835937500;0.000228881835937500;0.000244140625000000;0.000244140
625000000;0.000244140625000000;0.000244140625000000;0.000244140625000000;
0.000244140625000000;0.000228881835937500;0.000228881835937500;0.00024414
0625000000;0.000228881835937500;0.000228881835937500;0.00022888183593750
0;0.
7929687500;0.000671386718750000;0.000778198242187500;0.00030517578125000
0;0.0026550292968750;0.0029754638671875;0.0027923583984375;0.003585815
42968750;0.0032958984375000;0.000854492187500000;0.000961303710937500;0.
000808715820312500;0.0010375976562500;0.0017395019531250;0.00108337402
343750;0.000411987304687500;0.000183105468750000;0.000122070312500000;0.0
0016784667968750;0.000122070312500000;0.000167846679687500;0.0001525878
90625000;0.000167846679687500;0.000122070312500000;0.000167846679687500;0
.000183105468750000;0.000167846679687500;0.000198364257812500;0.000183105

468750000;0.00018310546875000;0.00016784667968750;0.00028991699218750;
0.00022888183593750;0.00038146972656250;0.00085449218750000;0.00123596
191406250;0.00077819824218750;0.00076293945312500;0.00094604492187500;
0.00071716308593750;0.00135803222656250;0.00129699707031250;0.0035247802
7343750;0.00477600097656250;0.00372314453125000;0.00028991699218750;0.00
27465820312500;0.00326538085937500;0.00297546386718750;0.00321960449218
750;0.00230407714843750;0.00180053710937500;0.00097656250000000;0.000900
268554687500;0.00094604492187500;0.00059509277343750;0.00085449218750000
000;0.00086975097656250;0.00074768066406250;0.00021362304687500;0.0003
20434570312500;0.00027465820312500;0.00027465820312500;0.0003204345703
12500;0.00025939941406250;0.00018310546875000;0.00025939941406250;0.00
0244140625000000;0.00022888183593750;0.00027465820312500;0.00033569335
9375000;0.00054931640625000;0.00077819824218750;0.00062561035156250;0.
0011596679687500;0.00083923339843750;0.00077819824218750;0.0005950927
73437500;0.00070190429687500;0.00085449218750000;0.00080871582031250;0
.00178527832031250;0.00125122070312500;0.00263977050781250;0.002044677734
37500;0.00267028085937500;0.00393676757812500;0.00028991699218750;0.0029
1442871093750;0.00335693359375000;0.00413513183593750;0.0039367675781250
0;0.00335693359375000;0.00331115722656250;0.00201416015625000;0.001983642
57812500;0.00167846679687500;0.00158691406250000;0.00134277343750000;0.00
07629394531250000;0.00050354003906250;0.00090026855468750;0.00071716308
5937500;0.00077819824218750;0.00097656250000000;0.00041198730468750;0.
00039672851562500;0.00028991699218750;0.00048828125000000;0.000610351
562500000;0.00094604492187500;0.00071716308593750;0.00091552734375000;
0.00047302246093750;0.00051879882812500;0.00099182128906250;0.00067138
6718750000;0.00112915039062500;0.00059509277343750;0.00085449218750000;
0.00125122070312500;0.00163269042968750;0.00155639648437500;0.00285339355
468750;0.00181579589843750;0.00357055664062500;0.00306701660156250;0.0030
9753417968750;0.00300598144531250];

x1_step1_ymin = -1;

% Layer 1

b1 = [1.395553588867188;-1.109436035156250;-0.767379760742188;-
0.463806152343750;-0.128952026367188;0.157348632812500;-
0.485107421875000;0.775115966796875;-1.106063842773438;1.410690307617188];

IW1_1 =[-0.0546417236328125 -0.0559082031250000 -0.00973510742187500
0.0298156738281250 -0.0271911621093750 -0.0280609130859375
0.0492553710937500 -0.0555114746093750 0.0433959960937500 -
0.0502471923828125 0.0677490234375000 0.0612945556640625

0.0222625732421875 -0.0550537109375000 -0.00979614257812500 -
0.0141601562500000 -0.0227050781250000 0.0416107177734375 -
0.0122985839843750 -0.0224914550781250 0.0600280761718750 -
0.0297851562500000 -0.0263061523437500 0.00917053222656250
0.0511016845703125 0.0195312500000000 0.0328979492187500 -
0.0351715087890625 0.0250244140625000 0.00189208984375000 -
0.0157165527343750 -0.0106658935546875 0.0550079345703125
0.0259094238281250 0.00819396972656250 -0.0167694091796875 -
0.0268859863281250 0.0514221191406250 -0.0551300048828125
0.0192260742187500 -0.0137939453125000 0.0558471679687500 -
0.0291442871093750 -0.0765380859375000 0.0222320556640625
0.0194549560546875 0.0114593505859375 0.00375366210937500 -
0.00251770019531250 0.00360107421875000 -0.0164337158203125 -
0.0695648193359375 -0.0629425048828125 -0.0247955322265625 -
0.0203552246093750 0.0514526367187500 0.0445861816406250 -
0.0669555664062500 -0.00514221191406250 0.0526733398437500
0.00410461425781250 -0.0152740478515625 -0.00791931152343750 -
0.0541381835937500 -0.00483703613281250 -0.0643157958984375 -
0.0552215576171875 -0.0550231933593750 0.0286407470703125 -
0.00695800781250000 -0.0335693359375000 -0.0663299560546875 -
0.0587310791015625 -0.0636138916015625 -0.0638885498046875 -
0.0762786865234375 -0.00312805175781250 -0.0192108154296875 -
0.0597534179687500 -0.0464935302734375 -0.0303955078125000 -
0.0670471191406250 0.0317535400390625 0.0569458007812500 -
0.000671386718750000 -0.0840454101562500 -0.0607604980468750 -
0.0596008300781250 -0.0539703369140625 -0.0226593017578125 -
0.0339355468750000 -0.0760040283203125 0.0214538574218750
0.00364685058593750 0.0219421386718750 0.0432281494140625 -
0.0463562011718750 0.0287475585937500 -0.0348815917968750
0.0166473388671875 -0.0611114501953125 -0.0965423583984375 -
0.0724182128906250 -0.0926818847656250 -0.0791473388671875 -
0.0126953125000000 -0.0230255126953125 0.0426483154296875
0.00344848632812500 -0.0610046386718750 -0.0565032958984375 -
0.0460662841796875 -0.00550842285156250 0.000503540039062500 -
0.0367584228515625 0.0104370117187500 -0.0454864501953125
0.0466003417968750 -0.0110931396484375 0.00576782226562500 -
0.0645599365234375 -0.0265655517578125 -0.0696411132812500 -
0.0160675048828125 0.0112609863281250 -0.00196838378906250
0.0157165527343750 0.0368347167968750 0.0146331787109375 -
0.0120697021484375 -0.0524291992187500 0.0189819335937500 -
0.0366821289062500 0.0302124023437500 0.0135192871093750 -

0.0355987548828125 -0.00666809082031250 -0.0302276611328125
0.00445556640625000 -0.0291137695312500 -0.0393066406250000 -
0.0361938476562500 -0.00172424316406250 -0.0126037597656250 -
0.00840759277343750 0.000549316406250000 -0.00820922851562500 -
0.0288391113281250 -0.0464019775390625 0.0238189697265625 -
0.0499877929687500 -0.00961303710937500 0.0179595947265625 -
0.00521850585937500 0.0192108154296875 -0.0415344238281250 -
0.0804901123046875 -0.00149536132812500 -0.0260314941406250
0.00817871093750000 -0.0213317871093750 -0.0392761230468750 -
0.00630187988281250 -0.0504760742187500 -0.00204467773437500
0.0317535400390625 0.0378265380859375 0.0525207519531250
0.0204925537109375 0.0101470947265625 0.00822448730468750
0.0450286865234375 0.0489654541015625 -0.0224761962890625
3.05175781250000e-05 -0.0223999023437500 -0.0631103515625000 -
0.0788879394531250 -0.0595703125000000 -0.0340728759765625
0.0168457031250000 -0.0178680419921875 0.0199127197265625
0.0481109619140625 0.0169067382812500 0.0130157470703125 -
0.0291442871093750 -0.0809631347656250 0.00596618652343750 -
0.0228271484375000 -0.0279388427734375 0.00291442871093750 -
0.0255584716796875 -0.0265045166015625 0.0709533691406250
0.0632171630859375 -0.00711059570312500 0.0323791503906250
0.0531311035156250 0.0492706298828125 0.0483398437500000
0.0253753662109375 -0.0194396972656250 0.00775146484375000
0.0258331298828125 -0.0638427734375000 -0.0484008789062500 -
0.0396728515625000 -0.00704956054687500 0.0121307373046875
0.0306701660156250 -0.0396118164062500 -0.0180053710937500
0.00326538085937500 -0.00820922851562500 0.0324707031250000 -
0.0286865234375000 -0.0770111083984375 0.0290069580078125
0.00187683105468750 0.0460510253906250 0.0208129882812500
0.0231933593750000 0.00769042968750000 0.0139465332031250 -
0.00233459472656250 0.0584106445312500 0.0398559570312500
0.0626068115234375 0.0518798828125000 0.0436706542968750
0.0265045166015625 0.0149536132812500 0.0403747558593750
0.0348663330078125 -0.0874328613281250 0.0417938232421875
0.0259094238281250 0.00166320800781250 0.00791931152343750 -
0.0359954833984375 -0.0320892333984375 0.0106811523437500 -
0.0268859863281250 0.0421142578125000 -0.0103759765625000
0.0330505371093750 -0.0137786865234375 -0.0117645263671875 -
0.0240936279296875 -0.00949096679687500 -0.00827026367187500 -
0.0770874023437500 0.0160827636718750 -0.0812835693359375 -
0.0486450195312500 0.0230255126953125 -0.0455627441406250 -

0.0961151123046875 -0.0502319335937500 -0.0287475585937500 -0.0190277099609375 -0.0419158935546875 0.0149841308593750 0.0395355224609375 -0.0755004882812500 -0.0207061767578125 0.0315246582031250 -0.00523376464843750 -0.0174713134765625 -0.0377349853515625 -0.0539550781250000 -0.0115203857421875 0.00372314453125000 0.00440979003906250 -0.0283813476562500 -0.0425720214843750 -0.0755767822265625 0.0117187500000000 -0.0533905029296875 0.0295562744140625 -0.0669555664062500 0.0425567626953125 -0.00576782226562500 0.0333862304687500 -0.0607910156250000 0.000900268554687500 -0.0630340576171875 -0.00982666015625000 -0.0454864501953125 -0.0955963134765625 -0.104354858398438 -0.00117492675781250 -0.0854339599609375 0.0430145263671875 0.0709533691406250 -0.000335693359375000 0.0508880615234375 -0.0349273681640625 -0.0431671142578125 0.0314636230468750 0.0270538330078125 -0.0655670166015625 -0.0291442871093750 -0.0153045654296875 -0.0695648193359375 -0.0179595947265625 0.0556335449218750 0.0473022460937500 -0.0368957519531250 -0.0474090576171875 -0.0568695068359375 0.0302124023437500 -0.0375061035156250 0.00642395019531250 -0.000244140625000000 -0.0291442871093750 0.0427093505859375 0.0196533203125000 0.00871276855468750 -0.0748596191406250 -0.0785522460937500 -0.0697479248046875 -0.0629730224609375 -0.0812683105468750 -0.0559539794921875 -0.0665588378906250 0.0316314697265625 -0.0606231689453125 -0.0748748779296875 -0.0123748779296875 -0.109848022460938 -0.0346069335937500 -0.0961761474609375 -0.0862426757812500 -0.0224761962890625 -0.00740051269531250 0.0414123535156250 -0.0266723632812500 -0.0407562255859375 -0.0485382080078125 -0.0504608154296875 0.0186462402343750 0.0415649414062500 -0.0104980468750000 0.00544738769531250 0.0161743164062500 -0.0272064208984375 0.0132751464843750 0.0553741455078125 -0.0312347412109375 -0.0453186035156250 -0.047729492187500 -0.0132446289062500 0.0259704589843750 0.0119323730468750 -0.00788879394531250 0.00685119628906250 0.00268554687500000 -0.0263214111328125 0.0378723144531250 0.0291290283203125 -0.0151672363281250 0.0218811035156250 -0.0496520996093750 0.0327453613281250 -0.0334320068359375 -0.0362548828125000 0.0230255126953125 -0.0625762939453125 -0.0719604492187500 0.0140686035156250 -0.0942993164062500 -0.0233154296875000 0.0355529785156250 0.0184173583984375 -0.0206451416015625 -0.0717620849609375 -0.0672607421875000 0.0249481201171875 -0.0364074707031250 -

0.00624084472656250 0.00587463378906250 0.0627136230468750 -0.0403137207031250 0.0358276367187500 -0.0541381835937500 0.0529937744140625 0.0321502685546875 0.0169372558593750 0.00083923339843750 -0.0486602783203125 -0.0122833251953125 0.0160827636718750 -0.0267181396484375 0.0497741699218750 -0.0488739013671875 -0.0365753173828125 0.0302886962890625 -0.0189208984375000 -0.0612030029296875 0.0187225341796875 0.0186462402343750 -0.0154266357421875 -0.0220336914062500 -0.0055389404296875 -0.0267944335937500 -0.00508117675781250 0.0312652587890625 -0.0561676025390625 0.0444183349609375 0.0303039550781250 -0.0496978759765625 0.0433959960937500 0.0522003173828125 -0.0279235839843750 -0.0244750976562500 -0.0125885009765625 -0.0120086669921875 -0.0663299560546875 -0.0540771484375000 -0.00515747070312500 -0.0255432128906250 -0.0358428955078125 -0.00186157226562500 0.0759277343750000 0.0647583007812500 0.0472259521484375 0.0138397216796875 0.0364532470703125 0.00302124023437500 -0.0254669189453125 0.0187377929687500 0.0556793212890625 -0.0541534423828125 0.0286102294921875 -0.0262908935546875 0.0137786865234375 -0.0561065673828125 -0.0256958007812500 -0.0505828857421875 -0.0356903076171875 -0.0427398681640625 0.00833129882812500 0.00784301757812500 0.0241546630859375 -0.0297698974609375 0.0265350341796875 0.0223999023437500 -0.0195465087890625 0.000289916992187500 0.0149841308593750 -0.0667724609375000 0.0283355712890625 0.0214691162109375 0.0163269042968750 0.0334777832031250 -0.0277709960937500 0.0170593261718750 0.0471191406250000 0.0178375244140625 0.0119323730468750 -0.0261535644531250 -0.00344848632812500 0.0599212646484375 -0.0200042724609375 -0.0675811767578125 0.0657806396484375 0.0184631347656250 0.0196685791015625 -0.0520324707031250 0.00155639648437500 0.0681152343750000 0.0226745605468750 0.00311279296875000 -0.00994873046875000 -0.0345916748046875 0.00900268554687500 0.00799560546875000 0.0321197509765625 -0.0756225585937500 -0.0421752929687500 -0.00793457031250000 -0.0232849121093750 -0.0545349121093750 0.0450439453125000 0.0469207763671875 0.0451507568359375 -0.0339965820312500 -0.0601654052734375 0.0324249267578125 -0.0571441650390625 0.0423278808593750 -0.0485229492187500 -0.0588989257812500 -0.0236816406250000 0.0601043701171875 -0.0532989501953125 0.0267333984375000 -0.0145111083984375 0.0560760498046875 0.0732574462890625 0.0116271972656250 0.0493011474609375];

```
% Layer 2

b2 = [-0.488906860351563;-0.237075805664063;0.381256103515625;0.174377441406250;0.331634521484375;-0.374023437500000;-0.501922607421875;-0.466857910156250];

LW2_1 = [-0.100173950195313 0.314727783203125 -0.277374267578125 -1.06237792968750 -0.260391235351563 0.834609985351563 -0.904510498046875 -0.822616577148438 0.0708465576171875 -1.00050354003906;-0.0214080810546875 -0.615753173828125 0.860000610351563 -0.263504028320313 1.13009643554688 0.357437133789063 -0.220382690429688 0.400054931640625 -0.822250366210938 0.826324462890625;-0.790618896484375 -1.21560668945313 -0.822555541992188 -0.491058349609375 0.357711791992188 -0.917495727539063 0.391647338867188 -1.05348205566406 0.155517578125000 -0.617095947265625;-1.02108764648438 0.370162963867188 -0.902893066406250 1.12228393554688 0.802993774414063 0.236846923828125 -0.329147338867188 0.541213989257813 -0.817886352539063 -0.239181518554688;0.447494506835938 -0.505477905273438 0.0640258789062500 0.0700683593750000 -0.802612304687500 -0.592544555664063 -0.204940795898438 -1.30377197265625 -0.481460571289063 1.18649291992188;-0.583099365234375 0.483551025390625 -0.904785156250000 -0.231658935546875 -0.456832885742188 -0.284912109375000 -0.773056030273438 -0.0162811279296875 1.08419799804688 0.747085571289063;0.694183349609375 0.628311157226563 0.650329589843750 -0.350311279296875 -1.13566589355469 0.466888427734375 1.15011596679688 0.365447998046875 -1.11372375488281 0.741973876953125;-0.255050659179688 0.523910522460938 -1.08854675292969 0.0383758544921875 0.683593750000000 -0.819686889648438 0.701889038085938 -0.634185791015625 0.587249755859375 1.14945983886719];


%load const_16 x1_step1_gain b1 IW1_1 b2 LW2_1;


% ===== SIMULATION ========


% Dimensions

Q = size(x1,2); % samples


% Input 1
```

```
  xp1 = mapminmax_apply(x1,x1_step1_gain,x1_step1_xoffset,x1_step1_ymin);

  % Layer 1
  a1 = tansig_apply(repmat(b1,1,Q) + IW1_1*xp1);

  % Layer 2
  a2 = softmax_apply(repmat(b2,1,Q) + LW2_1*a1);

  % Output 1
  y1 = a2;
end

% ===== MODULE FUNCTIONS ========

% Map Minimum and Maximum Input Processing Function
function y = mapminmax_apply(x,settings_gain,settings_xoffset,settings_ymin)
  y = bsxfun(@minus,x,settings_xoffset);
  y = bsxfun(@times,y,settings_gain);
  y = bsxfun(@plus,y,settings_ymin);
end

% Competitive Soft Transfer Function
function a = softmax_apply(n)
  nmax = max(n,[],1);
  n = bsxfun(@minus,n,nmax);
  numer = exp(n);
```

```
  denom = sum(numer,1);
  denom(denom == 0) = 1;
  a = bsxfun(@rdivide,numer,denom);
end

% Sigmoid Symmetric Transfer Function
function a = tansig_apply(n)
  a = 2 ./ (1 + exp(-2*n)) - 1;
end
```

ПРИЛОЖЕНИЕ 3

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2016610061

Среда моделирования цифровой фильтрации в системе остаточных классов с модулями специального вида

Правообладатель: ***Федеральное государственное автономное образовательное учреждение высшего профессионального образования «Северо-Кавказский федеральный университет» (RU)***

Авторы: ***Шульженко Кирилл Сергеевич (RU), Ляхов Павел Алексеевич (RU), Червяков Николай Иванович (RU), Калита Диана Ивановна (RU)***

Заявка № **2015660741**
Дата поступления **09 ноября 2015 г.**
Дата государственной регистрации
в Реестре программ для ЭВМ ***11 января 2016 г.***

Руководитель Федеральной службы
по интеллектуальной собственности

Г.П. Ивлиев