

**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ АРМЕНИИ
ИНСТИТУТ ПРОБЛЕМ ИНФОРМАТИКИ И АВТОМАТИЗАЦИИ НАН РА**

Петросян Александр Усикович

**РАЗРАБОТКА КОМПЛЕКСА АЛГОРИТМОВ И ПРОГРАММ,
ПРЕДНАЗНАЧЕННЫХ ДЛЯ ЗАЩИТЫ И ВОССТАНОВЛЕНИЯ ИЗОБРАЖЕНИЙ
С ИСПОЛЬЗОВАНИЕМ МЕТОДОВ САМОВНЕДРЕНИЯ**

**Диссертация на соискание учёной степени кандидата технических наук по
специальности 05.13.04 – “Математическое и программное обеспечение
вычислительных машин, комплексов, систем и сетей”**

Научный руководитель

Д.Г. Асатрян

Доктор технических наук, профессор

ЕРЕВАН - 2016

СОДЕРЖАНИЕ РАБОТЫ

ВВЕДЕНИЕ	4
Г Л А В А I.....	8
МЕТОДЫ ЗАЩИТЫ И ОБРАБОТКИ ИЗОБРАЖЕНИЙ.....	8
1.1 Цифровые системы обработки изображений.....	8
1.2 Информационная безопасность изображений	12
1.3 Технологии встраивания ЦВЗ и самовнедрения	15
1.4 Форматы и методы кодирования.....	26
1.5 Многопоточная модель графических процессоров	30
Г Л А В А II	39
ПОСТРОЕНИЕ ПРОСТРАНСТВЕННОГО АЛГОРИТМА САМОВНЕДРЕНИЯ ИЗОБРАЖЕНИЯ	39
2.1 Основные аспекты разработки алгоритма встраивания ЦВЗ методом самовнедрения.....	39
2.2 Алгоритмы оценки качества изображений.....	44
2.3 Алгоритмы смешивания пикселей	46
2.4 Пространственно-частотный алгоритм самовнедрения изображения	54
Г Л А В А III.....	60
МЕТОД СМЕШАННОГО КОДИРОВАНИЯ И ВСТРАИВАНИЯ ЦВЗ	60

3.1 Метод отклонений (разностей).....	60
3.2 Метод прямоугольных сеток.....	66
3.3 Фрагментное представление изображения	68
 Г Л А В А IV	71
ПРИЛОЖЕНИЯ, ПРОГРАММНАЯ РЕАЛИЗАЦИЯ	71
4.1 Общая характеристика программной системы	71
4.2 Методы обработки изображений в контексте разработанной программной системы	76
4.3 Применение алгоритмов самовнедрения изображения для обеспечения безопасности электронных подписей	82
4.4. Средства для работы с модулями параллельных вычислений	87
 ОСНОВНЫЕ РЕЗУЛЬТАТЫ ДИССЕРТАЦИОННОЙ РАБОТЫ	89
СПИСОК ПУБЛИКАЦИЙ.....	89
СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ.....	91

ВВЕДЕНИЕ

Актуальность темы

Проблема авторского права на изображения (например, авторское право на картину), а также необходимость проверки подлинности тех или иных визуальных данных являются актуальными с давних времён в различных аспектах жизнедеятельности людей. В частности, это может быть причастность некоторых лиц к каким-либо событиям, изображённым на фотографии, электронные подписи, подлинность документов, фактическая достоверность улик, изображённых на фотографиях и т.д.

Начиная с середины XX века изображения в основном обрабатывались аналоговыми методами или же с помощью оптических устройств. На сегодняшний день схожие методы также используются в некоторых областях науки, например, в голографии. Учитывая резкий рост производительности вычислительных систем в последние годы, данные методы в основном были вытеснены цифровыми методами обработки изображений. Цифровые методы обработки графических данных более точны, гибки, надёжны и просты в реализации по сравнению с аналоговыми методами.

На данный момент существует множество методов представления визуальной информации в электронном формате, но, к сожалению, большинство из них не позволяет удостовериться в подлинности того или иного изображения (например, фотография может быть подвергнута искажениям, манипуляциям, что выявить часто крайне затруднительно, а то и не представляется возможным и во многом зависит от человеческого фактора).

Цифровые визуальные данные в большинстве своём представляются, кодируются и распространяются с помощью наиболее распространённых форматов их представления, таких как GIF, JPEG, PNG. Несмотря на значительные преимущества данных форматов кодирования дискретных изображений, недостаток данных форматов заключается в

большой сложности нахождении искажений по сравнению с имеющимся оригиналом, что в большинстве случаев вообще не представляется возможным, ещё более проблематична задача восстановления оригинала.

В настоящее время наиболее распространены методы встраивания дополнительной информации в исходное электронное изображение, такую информацию принято называть ЦВЗ (цифровой водяной знак). Имеется множество разнообразных алгоритмов встраивания ЦВЗ в изображение.

Настоящая диссертационная работа посвящена разработке комплекса алгоритмов и программ для защиты цифрового изображения с возможностью его восстановления.

В данной работе разработан один из усовершенствованных методов защиты и восстановления электронного изображения, основанный на внедрении в него определённого образа самого защищаемого изображения, закодированного новым методом. Предложенный подход обеспечивает также повышенный уровень устойчивости процедуры защиты авторского права к различным типам преднамеренных и не преднамеренных атак. В соответствии с этим был предложен метод для хранения и внедрения ЦВЗ в изображение, в результате которого совмещаются методы сокращённого кодирования изображения и встраивания ЦВЗ.

Цель работы

Целью диссертационной работы является разработка комплекса алгоритмов и программных модулей, основанных на самовнедрении сжатого и закодированного изображения, обеспечивающих одновременно защиту авторского права и восстановления искажённого оригинала.

Для достижения поставленной цели были сформулированы и решены следующие задачи:

- Разработка методов защиты визуальной информации от несанкционированного доступа и искажения данных путём использования методов самовнедрения;

- разработка методов кодирования изображения с повышенной устойчивостью к атакам и искажениям;
- разработка алгоритмов и программных модулей восстановления исходника видеоизображения, основываясь на декомпиляции атакованного изображения.

Научная новизна

- разработаны методы обработки, хранения и передачи графических данных с использованием методов самовнедрения и кодирования электронного изображения;
- разработан усовершенствованный пространственно-частотный алгоритм защиты электронного изображения;
- разработан новый подход для кодирования и хранения изображений, основанный на обработке растровых данных изображения;
- предоставлена процедура оптимизации метода встраивания ЦВЗ в изображение, закодированного в использованном разработанном формате;
- разработаны методы для оптимизации полученных алгоритмов под многопоточные видеочипы типа GPGPU.

Практическая ценность

- Разработаны программные системы и модули для реализации перечисленных выше методов;
- программное обеспечение позволяет обрабатывать графические данные в реальном времени, тем самым реализуя возможность использования его в различных прикладных целях;
- применение алгоритма возможно для статических изображений, так и для динамических видеоданных;

- реализован механизм использования программного обеспечения в мобильных устройствах;
- разработан сетевой сервис обработки электронных изображений;
- реализована возможность использования графических чипов GPGPU с использованием технологии NVIDIA CUDA для производства вычислений описанных алгоритмов и методов.

На защиту выносятся:

Методы защиты и восстановления изображений путём их самовнедрения в качестве ЦВЗ обеспечивающие сохранность данных и устойчивость к различным искажениям и атакам;

- алгоритмы, их программные реализации и экспериментальные результаты исследований в данном направлении;
- алгоритмы и программная реализация многопоточного кодирования учитывающие особенности использования возможностей графических видеочипов типа GPGPU;
- клиент-серверная технология конвертации при использовании мобильных систем.

Апробация работы

Материалы и положения диссертационной работы обсуждались во время семинаров ИПИА, а также на семинарах Российско-Армянского (Славянского) университета. Материалы докладывались на 9-ой международной конференции “Computer Sciences and Information technologies” (CSIT-2013, г. Ереван, Армения), а также на Международной Научно-Практической конференции: «Проблемы и перспективы современной науки» 2016 (г. Москва).

Г Л А В А I

МЕТОДЫ ЗАЩИТЫ И ОБРАБОТКИ ИЗОБРАЖЕНИЙ

1.1 Цифровые системы обработки изображений.

Начиная с середины XX века изображения в основном обрабатывались аналоговыми методами или же с помощью оптических устройств. Такие оптические методы используются и по сей день в некоторых областях, таких как голография. С увеличением производительности компьютеров данные методы были заменены методами цифровой обработки изображений. Цифровая обработка визуальной информации более практична по сравнению аналоговыми методами, такая обработка даёт прирост в скорости выполнения, а также в качестве полученного результата. Во множестве программных систем имеются отдельные модули расширения для обработки изображений (например, пакет Mathematica, Mathcad, MatLab, Maple). Ранее в цифровой обработке в качестве основного инструмента использовался один центральный процессор. На сегодняшний же день наиболее практично использование многопоточных систем в том числе графических процессоров.

На практике графическая информация представляется в виде сигналов, которые передаются через некоторый канал данных. В таком случае алгоритмы обработки применяются для одномерных сигналов. Данные алгоритмы в основном обобщаются так же и для двумерных данных, которыми представляются цифровые фотографии, рассматриваемые в данной работе. При обработке цифровых фотографий представленных в виде двумерных массивов привносятся такие понятия как ротационная инвариантность, связность, которые не имеют смысла для одномерных сигналов. Скажем при обработке двумерных цифровых фотографий широко используются такие алгоритмы, как вейвлет

преобразование, преобразование Фурье, косинусное преобразование, фильтр Габора [1], [2]. Обработку изображений можно разделить на два подкласса – пространственную обработку (гамма, яркость) и частотную (ДКП, ДПФ) [3].

На данный момент большинство офисного программного обеспечения использует ресурсы центрального процессора для расчётов [4], производимых над графическими файлами, хотя имеются более подходящие методы для такого рода вычислений, а именно, произведение вычислений с помощью графических процессоров.

За последние годы в мировой компьютерной индустрии произошёл переворот в области архитектуры центральных процессоров, и в общем архитектуры персональных компьютеров в целом. Переворот заключался в вынужденном переходе процесса построения центрального процессора с одноядерным вычислительным ядром на многоядерную архитектуру вычислений.

Одноядерные системы утверждали свои позиции в течение десятков, а то и сотен лет.

Такой резкий переход был обусловлен не только физическими ограничениями одноядерной архитектуры (например, сильное тепловыделение, резко увеличивается при уменьшении техпроцесса), но и стремительным ростом популярности некоторых многоядерных систем, создавших конкуренцию производителям одноядерных систем.

К таким многоядерным системам относятся графические адаптеры, которые изначально конструировались без малейшего намёка на многоядерную систему обработки вычислений. Но со временем для вывода изображений и 3D (3 Dimensions, что предполагает 3 измерения) информации на экраны понадобилось использование множества «не слишком» производительных вычислительных элементов.

Как оказалось, для обработки графической информации целесообразно использование очень большого количества вычислительных элементов [5].

С годами количество параллельно работающих вычислительных элементов достигло порядка десятков тысяч, с быстрой частотой обработки данных в каждом вычислительном узле.

Вначале производители дали возможность использования вычислительных элементов с помощью специфичных микрокоманд, отдалённо напоминающих язык ассемблера, что создавало множество препятствий для программирования подобных систем. Но, начиная с 2007 года, компания NVIDIA впервые выпустила высокоуровневый язык программирования таких систем – NVIDIA CUDA и ввела понятие GPGPU (GPU общего назначения; General-Purposes Graphic Process Unit) (подробное описание следует в дальнейших главах). Как показали тесты, при правильном программировании таких систем можно получить производительность, на порядок превосходящую производительности современных многоядерных процессоров [5], [6].

Вернёмся к самому исконному понятию изображения, дадим его общее определение для понимания его особенностей, более того, дадим математические характеристики изображения. В работе будем работать с растровым заданием изображения. Иными словами, изображение – это ни что иное, как прямоугольная матрица A_{ij} размера $m \times n$, элементы которой целочисленные и лежат в интервале от 0 до 255. В данной работе мы будем в основном рассматривать серотонные изображения дабы избежать лишних нагрузок в вычислениях, по сути любое цветное изображение представляет собой матрицу, состоящую из тройки целых чисел от 0 до 255. В результате цветное изображение задаётся тремя серотонными изображениями. На самом деле данное понятие растрового изображения возникло из кибернетики и определяется следующим образом: растровое изображение – изображение, представляет собой сетку пикселей — цветных точек (или серотонных) (обычно прямоугольных) на мониторе, бумаге и других отображающих устройствах. Так же в кибернетике и программировании существуют такие определения изображения, как, например, векторное изображение – изображение, которое описывается математически элементарными геометрическими объектами, содержащимися в изображении, обычно называемых примитивами, такими как: точки, линии, сплайны, кривые Безье, круги и окружности, многоугольники [7]. Но в данной работе мы будем иметь дело преимущественно с растровыми изображениями, это связано со спецификой разрабатываемых методов и алгоритмов, которые будут описаны позднее,

поэтому вышеуказанное математическое определение изображения целиком подходит в данном контексте.

Определим такое понятие, как код изображения. Вообще говоря, под кодом изображения мы будем понимать обычный код относительно слов – а именно, это правило или некоторый алгоритм, который, сопоставляет для каждого сообщения строго определённые комбинации знаков, сигналов или символов. Фактически, код для изображений – это правило, при помощи которого матрице изображения A_{ij} (как мы сказали ранее) сопоставляются определённые комбинации символов, знаков или сигналов. Но следует так же отметить что в нашем случае такой код должен быть в определённом смысле обратимым, а именно, нам необходимо из полученного кода получить некоторое представление об исходном изображении или, что тоже самое, о целочисленной матрице A_{ij} . Не обязательно, чтоб код целиком восстанавливал изображение, важно то, чтобы мы получили необходимую информацию об изображении. Например, в одном случае это могут быть всего лишь приблизительные образы, отображённые в изображении, в другом случае это возможно будет текст, символы, очертания, детали. Ясно, что для более детализированного кодирования изображений потребуется больший объём кода, а для приблизительных очертаний – меньший. «Кодированием назовём процессом преобразования сообщений в комбинации символов». «Декодированием – называем процесс восстановления сообщений из комбинаций символов» [8].

Будем говорить, что изображение закодировано «как есть» или «без сжатия», если код ставит в соответствие каждому элементу матрицы аналогичный элемент в коде.

Также следует отметить, что существует бесконечное множество кодов, кодирующих одно и то же изображение. Среди этих кодов существуют такие, которые, определённо, «наилучшим» образом кодируют данное изображение. Под термином «наилучшим» образом мы понимаем наиболее краткое кодирование с максимальной сохранностью изображения при декодировании. Важно также отметить, что алгоритмы кодирования обязательно должны сопоставлять любому изображению, как минимум один соответствующий код, представляющий данное изображение. Тем самым один метод

кодирования может «наилучшим» образом кодировать один тип изображения, но отнюдь не «наилучшим» образом кодировать другой тип изображения.

На данный момент в компьютерной технике существует множество методов кодирования изображений, собственно благодаря им изображение превращается в набор байтов и сохраняется на носителе информации. Набор байтов объединяется в логическую структуру, представляющую собой файл и говорится, что полученный файл обладает некоторым форматом кодирования. Тем самым метод кодирования отождествляется с понятием формат файла или что-то же самое, что и формат изображения. Несмотря на то, что было упомянуто, что существует большое количество форматов изображений (что, то же самое, что и разновидностей кодирований), среди них выделяются некоторые форматы, которые позволяют кодировать изображения достаточно малым количеством информации и малым искажением. Среди таких форматов можно выделить форматы: JPG, GIF, PNG. Также существует достаточно распространённый формат BMP. Данный формат позволяет кодировать изображение «как есть» или «без сжатия» (см. определение стр.11).

1.2 Информационная безопасность изображений

Проблема защиты информации от несанкционированного доступа существует с начала развития человечества, так как информация является одним из основных ресурсов развития человечества. Так как информация является своего рода ресурсом, следовательно, возникает вопрос права владения такого ресурса. Отсюда и возникает вопрос авторского права на интеллектуальную собственность. С развитием человечества вопрос стал наиболее актуален в последнем столетии. Обладание необходимой информацией играет решающую роль также в аспектах государственной безопасности, существует множество законов регулирующих данную область.

Первый закон, вводящий авторское право, был принят ещё в 1710 в Британии [9][10]. Это основательно изменило ситуацию распространения цифровых технологий. В

результате множество законов были пересмотрены. Например, в ряде стран введены законы относительно регуляции технических средств защиты авторских прав (DRM) – средства, затрудняющие создание копий произведений, на которые распространяется авторское право (передаваемых в электронной форме), либо позволяющие отследить создание таких копий. За обход или нарушение DRM предусматриваются административные наказания. В США закон был принят в 1998-ом году [11]. В Армении соответствующие законы были приняты в 1999-ом году (ՀՕ-28) и были заменены в 2006-ом году (ՀՕ-142) и 2013-ом (ՀՕ-22). Данными законами пользуются такие ИТ организации, как Microsoft, Amazon, Apple, Sony.

Распространение информации на сегодняшний день осуществляется преимущественно с помощью цифровых технологий, в том числе с помощью передаваемой таким образом визуальной информации. Визуальная информация в свою очередь также делится на ряд подклассов (например, цифровые фотографии, видео файлы, анимационные клипы и т.д.). В конечном итоге такая информация представляется преимущественно в виде растровых данных о которых мы говорили ранее.

В технические средства защиты авторских прав так же включены технологии, относящиеся к встраиванию в мультимедийные файлы (в том числе и изображения) цифровых водяных знаков (ЦВЗ), названные так по аналогии с технологией добавления водяных знаков на бумагу [12], [13]. Методы очень сходны потому, что призваны, во-первых, защитить исходник от подделывания, содержат некоторую информацию, не слишком искажающую оригинал, а также не заметны для невооружённого человеческого глаза. Впервые метод был использован в Италии в 1282-ом году в городе Болонье. Данным методом производитель бумажных изделий идентифицировал свою продукцию. Позже метод начали использовать в государственных документах для защиты их от подделывания. Что же касается направления встраивания Цифровых Водяных Знаков, то как технология и научный термин он появился сравнительно недавно. Впервые термин был использован в 1994 году в работе [14]. Но также, как и мультимедийные технологии и необходимость их защиты, технологии встраивания ЦВЗ получили бурное развитие в течение последних двадцати лет [15]–[25]. Начиная с 1996 года периодически проводятся

конференции, посвящённые ЦВЗ технологиям, стеганографии и криптографии, как во всем мире, так и в Армении [26]–[30],[30]–[34].

Определим понятие ЦВЗ следующим образом.

Определение. Цифровым Водяным Знаком (ЦВЗ) будем называть информацию, которая внедряется в сигнал или изображение (в основном, незаметно) с целью тем или иным образом контролирования его использования [35].

Наиболее распространены методики встраивания невидимых специальных меток в изображение [36]–[40] (также существует множество методов внедрения видимых ЦВЗ [41]–[43]). Методика основывается на модификации яркостных характеристик изображения таким образом, что человек, не знакомый с оригиналом, не сможет выявить факт внедрения некоторой информации, тем более устранить её. Особенно хорошо данная методика работает на изображениях, в которых имеется большое количество неоднородных областей, например, на изображениях листвы и деревьев. Методы встраивания ЦВЗ используются так же для скрытой передачи сообщений. Очень часто для защиты от искажений или подделывания технология используется для внедрения ЦВЗ в официальные документы, права, паспорта, билеты, кредитные карты, счёт-фактуры, медицинские и географические мультимедийные данные. Так же актуальна задача установления права распространения электронных фотографий исходных изображений компьютерного искусства, которые на сегодняшний день имеют значимую ценность, существует множество платформ электронной коммерции подобного рода. Множество методов встраивания ЦВЗ позволяет неоднократное его встраивание. Часто в изображение встраивается дополнительное ЦВЗ с некоторыми метаданными, тогда говорится, что изображение несёт дополнительную информацию.

Говорим, что изображение было атаковано, если относительно изображения со встроенным ЦВЗ были применены алгоритмы, в результате которых хотя бы одно числовое значение, представляющее визуальный облик данного изображения, было изменено.

Определение. Попытки устранения ЦВЗ из изображения называются *атаками*. Различают два типа атак – *преднамеренные*, которые осуществляются целенаправленно со стороны стороннего пользователя, и *непреднамеренные*. Непреднамеренные атаки, в основном, возникают в результате сбоев технических средств.

Существует множество методов встраивания ЦВЗ с различными характеристиками в отношении их устойчивости к тем или иным атакам. В результате встраивания ЦВЗ различными алгоритмами имеем изображение с различными оценочными показателями встроенного ЦВЗ, например, с различными показателями стойкости встроенного ЦВЗ к атакам. Более подробно на этих вопросах мы остановимся во второй главе этой работы.

1.3 Технологии встраивания ЦВЗ и самовнедрения

Технологии встраивания ЦВЗ появились сравнительно недавно, в связи с этим до сих пор терминология данного направления науки устоялась не окончательно. Многие понятия и термины были согласованы на первой международной конференции по сокрытию данных в 1996 году [44]. В основном различают три типа ЦВЗ: хрупкие, полухрупкие или робастные [45], [46]. Наиболее распространёнными являются *робастные ЦВЗ*.

Хрупкие ЦВЗ имеют свойство разрушаться при малейшем искажении изображения со встроенным ЦВЗ. В основном их применяют при сигнальной аутентификации. Хрупкие ЦВЗ так же часто позволяют в случае возникновения искажений указать на них. Отличие таких ЦВЗ от стандартных методов аутентификации заключается в возможности незначительного изменения переносчика данных без разрушения значительной закодированной информации.

Отличительной чертой *полу-хрупких ЦВЗ* же является их сравнительная устойчивость к определённым видам атак и хрупкость по отношению к другим типам воздействий. Основной целью создания таких алгоритмов внедрения ЦВЗ является

закладывание специальной хрупкости к определённым видам искажений. Например, ЦВЗ может быть устойчив к изменениям размеров изображения, но не устойчив к изменениям внутреннего содержимого изображения, тем самым сигнализируя о присутствии преднамеренных воздействий на изображение.

В последние несколько лет среди зарубежной литературы сильное распространение получили методы самовнедрения ЦВЗ (Self-Embedding Watermark). Дадим определение понятия *самовнедрения ЦВЗ*:

Определение. Будем говорить, что изображение закодировано методом *самовнедрения*, если в качестве ЦВЗ встраиваемого в данное изображение берётся такая информация, располагая которой можно получить некоторую (визуальную) информацию об исходном изображении до проведения над ним искажающих манипуляций. [47]–[50]

Как показывает практика, такие методы внедрения ЦВЗ наиболее практичны. Во-первых, при встраивании такого ЦВЗ в изображение решается проблема авторского права на него [51]–[55].

Во-вторых, в случае искажения закодированного изображения мы получаем возможность декодирования и получения информации о прежнем состоянии данного изображения. Наиболее практичны методы встраивания визуальной информации самого изображения. Проблема заключается в ограниченности пространства встраивания данной информации.

Для решения данной проблемы в научных публикациях используются полутоновые изображения [50].

Определение. Изображение будем называть полутоновым, если данное изображение имеет множество значений тона и его непрерывное плавное изменение.

Пиксель такого изображения, каждый раз может кодироваться изменчивым количеством битов. В этом и заключается принцип хранения множества различных полутонов. Метод работает на специфике человеческого зрения. Один цвет в таком кодировании может быть заменён несколькими различными цветовыми пикселями.

Существует множество исследований о конвертации стандартных растровых изображений в полутоновые. Недостаток такого метода заключается в том, что при различных размерах отображающего устройства представление данного изображения должно меняться для сохранения своих визуальных свойств.

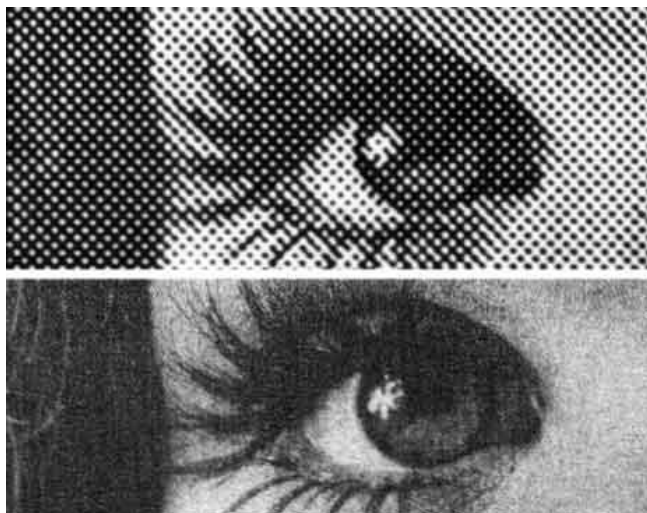


Рис.1. Пример серотонного полутонового изображения (а) снизу оригинал (б) сверху конвертированное полутоновое изображение

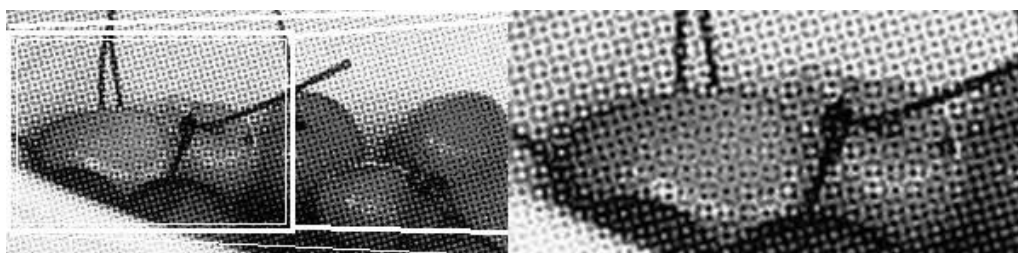


Рис.2. Иллюстрация получения визуального эффекта целостности общей картины полутонового изображения

Ещё одним важным фактором против данного метода является значительная потеря растровых данных после конвертирования изображения в данный полутоновый формат. Кодированное изображение таким образом так же не будет удовлетворять

распространённым критериям схожести изображений, например, PSNR (о котором будет оговорено далее).

Во многом требования к типам встраивания ЦВЗ лежат в конкретных приложениях, в которых они используются (например, в следующих работах рассмотрены разнообразные методы внедрения ЦВЗ в различные форматы изображения [56]–[63]). Далее приведём некоторые требования к **методам встраивания ЦВЗ** для наиболее распространённых приложений.

Критерий необнаруживаемости. Для методов самовнедрения и внедрения ЦВЗ требование необнаруживаемости означает незаметность встраивания информации для человеческой визуальной периферии. Процедура самовнедрения изображения не должна давать явных искажений исходного изображения. Другими словами, изображение, обработанное методом, внедряющим в него специальную информацию не должно визуально отличаться от изображения, над которым не было произведено никаких воздействий. Отличительной чертой этого критерия от аналогичного критерия в стеганографии является то, что там необнаруживаемость означает невозможность установления факта присутствия скрытого сообщения в изображении.

Критерий устойчивости. Одним из важных требований к технологиям внедрения ЦВЗ и следственно к методу самовнедрения является устойчивость внедрённой информации к различного рода атакам. Под данным критерием следует понимать следующее: если изображение, обработанное методами самовнедрения (или внедрения в него ЦВЗ) было подвергнуто искажениям, то извлечённая из него информация, как минимум должна быть визуально узнаваема или удовлетворять специальным показателям качества изображения.

Анализируя устойчивость метода, следует разделить критерии устойчивости самой встроенной информации в изображение и возможность обнаружения присутствия в изображении такой встроенной информации. При некотором преобразовании

изображения (например, при повороте или кадрировании изображения) может быть получена ситуация, в которой декодер не может определить присутствие в изображении ЦВЗ, учитывая то что он в искажённом состоянии содержится в изображении.

Критерий достаточного количества встраиваемой информации. Данный критерий заключается в следующем: алгоритм встраивания ЦВЗ должен позволять встраивание достаточного для конкретного приложения количества данных. Объем или размер ЦВЗ определяется количеством встраиваемых битов (или байтов). Обычно этот критерий характеризуется процентным соотношением между объёмом встраиваемого ЦВЗ и объёмом оригинального объекта [64].

Критерий секретности или безопасности. Согласно данному критерию требуется, чтобы только авторизированные пользователи могли обнаружить или изменить встроенный ЦВЗ. Безопасность ЦВЗ-алгоритма интерпретируется так же, как и для методов кодирования [65], т.е. неуполномоченный пользователь не способен раскодировать информацию, если не знает секретного ключа, с помощью которого было осуществлено встраивание. Фактически, требование означает, что для неуполномоченного пользователя не представляется возможности извлечь встроенные данные в течение разумного времени, даже при том условии, что ему известно, что в сигнал встроено некоторое секретное сообщение, или он знаком с конкретным алгоритмом встраивания.

Отметим, что требования безопасности зависят от конкретного их приложения, и в некоторых случаях данные предварительно зашифровываются прежде, чем встроить их в исходный сигнал. Фактически, ЦВЗ и криптографические методы можно использовать совместно, что ещё более повышает уровень безопасности.

Критерий вычислительной простоты. Согласно данному критерию извлечение ЦВЗ из защищённого объекта для законного пользователя не должно представлять сложностей с вычислительной точки зрения и занимать много времени.

Критерий возможности встраивания нескольких ЦВЗ. В соответствии с данным критерием алгоритм встраивания ЦВЗ должен позволять встраивание в уже защищённый объект дополнительного цифрового водяного знака, в частности, в целях защиты от многократного копирования.

Критерий наличия или отсутствия информации о защищаемом объекте для обнаружения или извлечения ЦВЗ. Различают три вида алгоритмов для извлечения или обнаружения ЦВЗ. В литературе они называются следующим образом: *слепые* алгоритмы, *осведомлённые* и *полуслепые*.

Для случая «слепых» алгоритмов детектор ЦВЗ не имеет доступа к оригиналу объекта, в которое был встроен ЦВЗ, и работает «вслепую», но в этом случае количество встраиваемой информации существенно уменьшается. Алгоритмам второго типа для извлечения ЦВЗ требуется оригинальный мультимедийный объект. Алгоритмам третьего типа для извлечения, встроенного ЦВЗ необходим не оригинал контейнера, а некоторая информация о нем, которая для некоторых приложений может рассматриваться как часть секретного ключа. Считается, что наибольшую область применения имеют «слепые» алгоритмы ЦВЗ.

Обратимся теперь к характеристикам искажений изображений. Различают несколько видов атак, применённых к изображению со встроенным ЦВЗ. Рассмотрим некоторые из них.

Классификация типов атак. Как уже упоминалось на стр. 15, цифровое изображение, защищённое встраиванием ЦВЗ, может быть подвергнуто атакам разного рода, при этом под атакой понимают любое действие, способное удалить или повредить встроенный ЦВЗ. И, как уже отмечалось, одним из основных требований к алгоритмам встраивания ЦВЗ является их устойчивость к атакам, как непреднамеренным, так и преднамеренным (см., например, [66]–[68]).

Непреднамеренные атаки не преследуют каких-либо целей, и возникают, чаще всего, помимо воли человека, в результате операций по хранению, передаче, и обработке изображения при помощи технических средств. Тем не менее, в результате

непреднамеренной атаки на цифровое изображение могут быть устранены меры, предпринятые для его защиты.

Преднамеренные атаки применяются с целью незаконного (несанкционированного) использования информации или присвоения прав на неё.

Многие атаки, незначительно ухудшая качество цифрового изображения, искажают или полностью разрушают встроенный ЦВЗ. Это вполне отвечает требованиям «пиратов», т.к. они зачастую не нуждаются в копиях высшего качества.

Рассмотрим *основные типы атак*.

В работе [35] классификация атак на изображения со встроенным ЦВЗ представлена на основании цели атаки в изображении. Авторы выделяют четыре основных направления атак: атаки, направленные на изображение в целом, атаки, направленные на детектор ЦВЗ, атаки, направленные на протокол использования ЦВЗ, и, наконец, атаки, направленные на само встроенное ЦВЗ.

В первом случае атаки направлены на такие манипуляции с изображением, как сжатие изображений, изменение контрастности, линейная фильтрация, выравнивание гистограммы, добавление шума и т.п.

Второго типа атаки направлены на затруднение или сбой работы детектора. При этом ЦВЗ в изображении не изменяется, но теряется возможность его приёма. К этому типу атак относятся так называемые аффинные преобразования, в частности, сдвиги, масштабирование, повороты также перестановка пикселей, усечение изображения и т.п.

Атаки против самого ЦВЗ направлены на несанкционированную оценку и выделение ЦВЗ из изображения. Среди атак этого вида можно выделить такие атаки, как статистическое усреднение, определённые виды нелинейной фильтрации, очистка сигнала от шумов и т.п.

Перейдём к более подробному описанию наиболее распространённых атак.

Сжатие с потерями. Сжатие является наиболее часто осуществляемой атакой. Наиболее популярные стандарты сжатия изображений JPEG, JPEG2000 имеют широкое применение, позволяя значительно уменьшить объем сохраняемых данных без

существенной потери качества. Это имеет большое значение, учитывая постоянный рост объёмов сохраняемой и передаваемой по Интернету информации.

При сжатии мультимедийный объект со встроенным цифровым водяным знаком может визуально практически не отличаться от оригинала, но ЦВЗ при этом может быть искажён до неузнаваемости [69]. Таким образом, для определения эффективности ЦВЗ-метода он должен, прежде всего, быть протестирован на устойчивость к атаке типа «сжатие».

Кроме того, устойчивость ЦВЗ алгоритма, предназначенного для встраивания в изображение формата BMP, к атакам типа «сжатие» по наиболее распространённым форматам позволяет их применять для файлов и других форматов.

Геометрические атаки. Геометрические атаки включают операции поворота, масштабирования, обрезки (усечения) и т. д., которые математически моделируются, как аффинные преобразования. Они являются характерными для искажения изображений и видео. Эти атаки нацелены на затруднение или невозможность эффективной работы детектора, так как приводят к потере синхронизации при извлечении ЦВЗ. Эти атаки могут быть локальными или глобальными. Атаки такого типа реализованы в программах Unsing (локальные атаки) и StirMark (как локальные, так и глобальные атаки).

Доказано, что атаки типа StirMark успешно удаляют ЦВЗ, встроенные с помощью коммерческих продуктов. Программа StirMark не специфицирована для конкретных известных методов и часто рассматривается, в качестве универсального теста, проверяющего устойчивость разнообразных стеганографических алгоритмов.

Атаки, использующие операции, применяемые при обработке сигналов. Отметим наиболее распространённые типы атак данного вида:

- дискретизация;
- квантование;

- *линейная фильтрация с большой и малой пропускной способностью и нелинейная фильтрация типа медианной*: Подобные атаки предполагают, что ЦВЗ может быть описан в виде статистического шума;

- *уменьшение числа оттенков*;

- *увеличение значений всех пикселей на постоянную величину*;

- *наложение шума*: Отметим, что шумы могут быть наложены на мультимедийный объект непреднамеренно, а в результате выполнения с объектом определённых операций. Например, при передаче по физическому каналу с шумом, при печати на принтере с последующим сканированием (так называемая атака «печать-сканирование»), при конвертации сигналов типа «цифровое-аналоговое» и «аналоговое-цифровое»;

- *локальный обмен значений пикселей*, и т. д.

Отметим, что представленная типология является одной из многих классификаций атак, однако она является одной из наиболее полных типологий. Однако заметим, что некоторые атаки можно отнести к нескольким типам, в частности, например, удаление шума.

Другие типы атак подробно описаны в научной литературе. Подробное исследование типов атак и их характеристик дано, например, в работе [70], поэтому мы их опускаем.

В соответствии со степенью соответствия алгоритма встраивания ЦВЗ вышеописанным критериям к различного рода атакам вернёмся к **методам встраивания ЦВЗ**.

Некоторым основным требованием для методов встраивания ЦВЗ является объём встраиваемой в изображение информации. В результате существует необходимость в таких универсальных методах, которые одновременно обеспечивали бы устойчивость к атакам, были бы не обнаруживаемыми и давали возможность встраивания большого объёма информации. Реально на практике имеем следующую картину зависимости критериев к встраиванию ЦВЗ и самих методов встраивания: (Рис.3)

Как можно заметить на рисунке при увеличении, например, устойчивости к атакам уменьшается количество встраиваемой информации, либо такое усовершенствование будет влиять на заметность ЦВЗ в изображении [71]–[73]. Отсюда можно заметить, что наиболее практично использование комбинированных методов встраивания ЦВЗ, в которых учтены по мере возможностей указанные критерии (как раз такие методы и будут рассматриваться далее в данной работе). Задача сводится к нахождению наиболее оптимального метода для конкретной задачи.

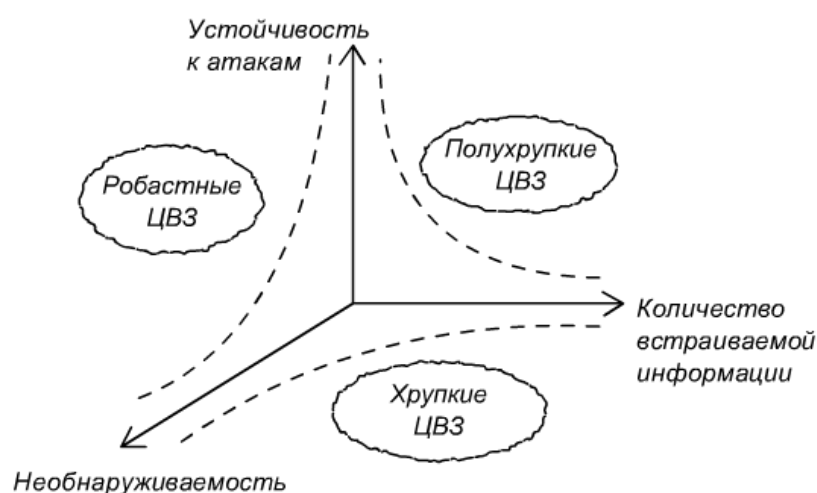


Рис.3. Зависимость типа ЦВЗ от основных требований к методу встраивания.

Также иногда в литературе можно встретить название: “гибридные методы встраивания ЦВЗ” (в следующие работы посвящены таким методам [74]–[77]). Данное понятие идентично с комбинированным методом встраивания ЦВЗ.

Одним из таких комбинированных методов и является пространственно-частотный алгоритм встраивания ЦВЗ [78]–[81], о котором и пойдёт речь в данной работе. Особенностью метода является разложение образа ЦВЗ на частотные гармоники, в результате чего производится сжатие и, тем самым, увеличивается объём встраиваемой таким образом информации [10], [70], [82]–[85].

Рассмотрим некоторые распространённые алгоритмы встраивания ЦВЗ (в следующих работах можно познакомиться с некоторыми известными использованными

методами [27]–[29], [86], [87]). Одним из таких алгоритмов является метод замены наименьшего значащего бита ещё именуемого LSB (Least Significant Bit). Метод заключается в сокрытии информации в изображении путём замены наименьшего значащего бита в каждом пикселе изображения. Для серого изображения имеется 255 цветовых оттенков при замене последнего бита мы теряем один оттенок. Такое изменение практически не уловимо человеческим глазом; тем самым, с использованием последнего бита для такого изображения в определённых целях визуальная информация для восприятия не искажается.

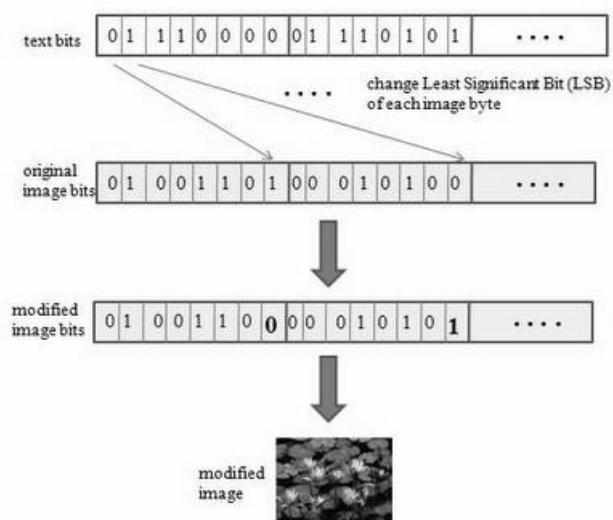


Рис.4. демонстрация работы алгоритма LSB

Допустим имеем серое изображение размером 300x300 пикселей. Это, что значит мы имеем 90 тысяч цветовых значений. Каждое такое значение состоит из 8 бит (число от 0 до 255). Метод резервирует каждый последний бит цветового значения для встраивания ЦВЗ. Тем самым получаем 90 тысяч бит для встраивания ЦВЗ. Указанный размер позволяет хранить 11 КБ информации. Среди недостатков данного метода является то, что он является крайне неустойчивым к различного рода искажениям, например, к сжатию. Тем самым получается невозможным его использование в таких

распространённых форматах как JPEG или аналогичных частотных форматах, предусматривающих сжатие с потерями.

1.4 Форматы и методы кодирования

Представленный в предыдущей главе метод встраивания ЦВЗ является примером хрупкого типа встраивания ЦВЗ. Информация, встроенная в изображение таким образом, не обнаруживается человеческим зрением в изображении, но тем самым, обеспечивая в полной мере критерий необнаруживаемости, в описанном методе страдают остальные показатели – устойчивость и размер встраиваемой информации. Такой метод не подходит для встраивания достаточного количества информации для восстановления исходного изображения. Для поставленных задач, таких как восстановление изображения подходят пространственные, частотные методы встраивания ЦВЗ [46], [88]–[92]. Обратим внимание на преимущества пространственно-частотных методов встраивания ЦВЗ [93], [94]. В чём заключается преимущество таких данных методов? Дело в том, что в основе пространственных методов лежит схожий алгоритм разложения с алгоритмом для сжатия изображений в формате JPEG. Это и позволяет его совместное использование с кодированием и хранением в данном формате. Разложение в частотном формате так же делает ЦВЗ более устойчивым к искажениям, т.к. возможная потеря информации несёт в себе частотную потерю некоторых деталей участка разложенного изображения.

Так же, говоря о форматах кодирования изображений обратим внимание на существующий распространённый формат кодирования изображений для веб-технологий – GIF (Graphics Interchange Format). Дословный перевод звучит следующим образом – «формат обмена изображениями». Данный формат был разработан в 1987 году предприятием под названием “Compu Serve” для передачи растровых данных изображений по компьютерным сетям. Данный формат использует алгоритм Лемпеля – Зива – Велча, так же в зарубежной литературе кратко именуемый LZW. Впервые

целостная версия данного алгоритма была опубликована в 1984 году, алгоритм, в качестве улучшения алгоритма LZ78.

Алгоритм во время процесса сжатия (кодирования) создаёт некоторую таблицу, отвечающую за преобразование строк: словам (некоторой последовательности символов) ставится в соответствие некоторая группа битов определённой длины (часто используют 12 бит). Данная таблица инициализируется каждой 1-символьной строкой (при использовании 8-битовых символов получаем 256 записей). В процессе кодирования данный алгоритм рассматривает содержимое посимвольно, сохраняя каждую последующую, уникальную двух-символьную строку в качестве пары – кода и символа, где код является ссылкой на соответственный начальный символ. После сохранения каждой новой двух-символьной строки в таблице, на вход подаётся код начального символа. На входе читая последующий символ, в таблице для него находится соответствующая уже встретившаяся строка наибольшей длины, после в таблице сохраняем код данной строки со последующим символом на входе; а соответственно на выходе даётся код данной строки, а последующий символ берётся в качестве начального символа следующей строки.

Для работы декодирующего алгоритма необходимо подать на вход только кодированный текст, так как алгоритм воссоздаёт соответственную таблицу непосредственно с помощью кодированного текста.

Далее представим алгоритм работы представленного метода в 6 шагах:

1. Задание словаря со всеми возможными фразами, состоящими из одного символа. Инициализирование подающейся на вход фразы W в качестве начального символа сообщения.
2. Поиск в словаре соответствующую строку W максимальной длины, которая должна совпадать с последними принятыми символами.
3. Производится считывание следующего символа K из закодированного текста.
4. Если встречается символ окончания сообщения, то выдаётся код для W .

5. Если данная фраза WK присутствует в таблице, то входной фразе W присваивается значение WK с последующим переходом к шагу N3, в обратном случае выдаётся код W, и добавляется WK в таблицу, далее входной фразе W присваивается K и вновь возвращаемся к шагу N3.
6. Выход.

LZMA (Lempel – Ziv – Markov chain-Algorithm) – алгоритм кодирования и сжатия информации, используемый с 2001 года. Один из основных алгоритмов распространённого архиватора 7-zip, которым создаются архивы сжатой информации в формате 7z.

Алгоритм функционирует на аналогичной схеме работы с таблицей слов, похожей на использованной в алгоритме LZ78 описанной ранее в данной главе и обеспечивает сравнительно высокий уровень сжатия (превышающий уровень сжатия таких алгоритмов, как алгоритм bzip2). Позволяет использование словарей различного размера, не более 4 гб [95]).

Также распространена функция командной строки с открытыми исходниками, во множестве современных операционных систем, под названием lzma, реализующая данный алгоритм сжатия данных. Данное программное обеспечение работает с файлами расширения «.lzma».

Официальный выпуск данного программного обеспечения начинается с бета версии 4.61 на лицензии CPL в категории общественного достояния. ПО имеет следующие указанные свойства:

Для процессоров частотой более 2ггц серии x86 сжатие должно производиться со скоростью более 1мб/с.

Для процессоров с частотой более 2ггц серии x86 извлечение (декодирование) должно производиться со скоростью более 10-20мб/с

ПО должно обладать поддержкой многопоточных систем.

Объем распаковочного кода LZMA должно составлять не более 5кб.

Расход оперативной памяти устройства должен зависеть непосредственно от заранее определённого размера словаря.

Существует так же возможность установления авторского права посредством расчёта контрольных сумм изображения. Далее рассмотрим наиболее распространённые такие алгоритмы.

AES (Advanced Encryption Standart), метод также известен под названием Rijndael - симметрический алгоритм поблочного шифрования (в качестве стандарта взят размер блока равный 128 битам, а размер ключа равный одному из приведённых значений 128 / 192 / 256 битов). Данный алгоритм принят стандартом шифрования правительством США по результатам конкурса проведённой на конференции посвящённой безопасности правительственных данных. Данный алгоритм широко анализирован и сейчас активно используется, аналогично с его предшественником DES (алгоритм блочного шифрования). Спецификация данного алгоритма была опубликована национальным институтом стандартов и технологий США (NIST – National Institute of Standards and Technology) в 2001 году в ноябре. До того в течение пяти лет производилось создание анализ подобных методов, кроме AES было представлено еще 15 методов. В 2002 году же в качестве международного стандарта шифрования был назначен алгоритм AES. В 2009 году алгоритм AES попал в списки наиболее распространённых алгоритмов шифрования [96]. Поддержка данного алгоритма введена ведущими производителями электронных схем, таких как Intel. Несмотря на это производительность данного алгоритма для большого для большого объёма данных требует значительных вычислительных ресурсов (например, для побитового шифрования изображений).

Алгоритм DES был американским государственным стандартом начиная с 1977 года. В 2000 году по результатам конкурса было объявлено, что алгоритм DES не уже не является наиболее оптимальным вариантом и его место было занято алгоритмом Rijndael. С того времени был начат процесс стандартизации алгоритма. В 2001 году алгоритм AES был принят в качестве FIPS 197. Историческая ретроспектива представлена в работе NIST [97].

Так же среди алгоритмов шифрования известен алгоритм Message Digest 5 (более распространено название MD5) – алгоритм хеширования (128-битный). Впервые был представлен Ривестом Л.Р. - профессором из института MIT (Massachusetts Institute Technologies; Масачусатский Технический Университет) с 1991 года. Идея алгоритма заключается в создании некоторых так называемых отпечатков сообщения любой длины и дальнейшей проверки подлинности данных сообщений. Часто применяется для удостоверения подлинности информации или хранения паролей в зашифрованном виде.

Данный алгоритм состоит среди серии алгоритмов по построению так называемых «дайджестов» (отпечатков) сообщения. Был представлен в качестве усовершенствованного варианта алгоритма-предшественника MD4 [98]. Описание алгоритма можно встретить в [98]. В дальнейшем профессор Г. Доббертин обнаружил серьёзные недостатки в алгоритме MD4 тем самым ставя точку в вопросе его использования. В литературе можно встретить описания ряда недостатков данного алгоритма MD5. Среди таких недостатков можно отметить «многозначную» обратимость алгоритма, т.е. можно получить множество кодов, получающих универсальную последовательность.

1.5 Многопоточная модель графических процессоров

В работе используются методы программирования графических процессоров вида GPGPU для реализации алгоритмов самовнедрения изображений. В данном параграфе будет представлена и обоснована целесообразность использования графических процессоров в поставленных задачах на иллюстрации вычислительных возможностей подобных графических систем. Введём некоторые основные понятия такие как GPGPU. GPGPU (GPU, GPGP, General Purpose Computing for Graphics Processing Unit, специальные вычисления на графических процессорах) – технология которая использует графический процессор видеокарты. Обычно на нем имеют место вычисления, связанные с компьютерной графикой, для выполнения расчётов в ПО для вычислений общего

назначения, которые часто производит центральный процессор. Данная технология обрела возможность реализации благодаря введению программных шиферных блоков и более точных арифметических расчётов растровых конвейеров, что позволило разработчикам программного обеспечения программировать процессоры потоков в неграфических целях.

С годами производителями наращивались частоты универсальных единичных процессоров, что привело к проблеме физических ограничений данной технологии и высокому энергопотреблению. Тем самым вынуждено для увеличения частоты вычислительного ядра процессора приходится размещать несколько вычислительных ядер в одном процессоре. Доступные на сегодняшний день центральные процессоры имеют поддержку до тридцати двух ядер в экстремальных версиях моделей центральных процессоров, и рост производительности с каждым разом значительно уменьшается. В таблице приведены статистические данные роста частот с 2000 года по 2014 год.

Год	2000	2003	2005	2007	2010	2014
Кодовое название ядра	Willamette	Gallatin	Prescott	Yonah	Bloomfield	Broadwell
Частота ядра	1.5 ГГц	3.3 ГГц	3.8 ГГц	1.9 ГГц x2	3 ГГц x4	3.8 ГГц x4

Таблица 1. Демонстрация роста частот одного вычислительного ядра в формате «Год» - «Частота» [99]

Доступные на сегодняшний день процессоры в большинстве имеют 2-4 ядра, предназначенные для обработки стандартных приложений, использующие архитектуру MIMD – множественные потоки команд и данных. Одно ядро в данной архитектуре

работает отдельно от других ядер, выполняя различные инструкции для различных процессоров.

Среди первых необходимых инструкций были представлены специальные векторные вычислительные возможности центральных процессоров SSE2 и SSE3 для двойной точности (двухкомпонентных) и одинарной точности (четырёхкомпонентных) вычислений с плавающей точкой из-за увеличивающихся требований к графическим приложениям. По этой причине для задач, рассматриваемых в данной работе гораздо более выгодно использование GPU (графического процессора), так как GPU ориентированы именно на работу с графическими данными, которые обрабатываются алгоритмами, разрабатываемыми в данной работе.

Например, в видеоустройствах компании NVIDIA главный блок представляет собой многопроцессорный блок с десятками основных ядер и сотнями вычислительных арифметических блоков ALU в целом, а также устройство имеет тысячи регистров для быстрой обработки чисел с плавающей точкой и имеет сравнительно небольшое количество разделяемой памяти (общей памяти). Также каждый видеоадаптер имеет память, выделенную специально для констант, имеет быструю глобальную память, которая доступна для каждого мультипроцессора, а также в распоряжении имеется локальная память для каждого из мультипроцессоров (которая быстрее глобальной).

Основной особенностью является то, что основные ядра мультипроцессора в GPU разработаны на основе архитектуры SIMD (один командный поток, множество потоковых данных). В данной архитектуре все ядра одновременно выполняют одни и те же инструкции. Подобная архитектура является типичской при обработке графических данных и аналогичных обрабатываемых алгоритмов, используемых во многих научных задачах. Несмотря на графическую ориентированность данных процессоров, даже при реализации программ соответствующей тематики, процессоры требуют специфического программирования, в котором учитывается архитектура SIMD. Благодаря такому подходу возможно значительно нарастить количество блоков исполнения за счёт упрощения.

Перечислим основные архитектурные различия между GPU (графический процессор) и CPU (центральный процессор). Каждое ядро CPU ориентировано на

выполнение одного потока команд (последовательных инструкций) с наибольшей производительностью, в свою очередь GPU спроектированы для быстрого выполнения значительного количества параллельных выполняющихся потоков команд (инструкций). Каждый универсальный процессор видеочипа оптимизирован для достижения максимальной производительности одного единого потока инструкций, обрабатывающего числа с плавающей точкой и целочисленные значения. При этом происходит так называемый «рандомизированный» доступ к памяти.

В отличие от архитектуры GPU, архитектура CPU наращивает наибольшую возможность исполнения значительного количества команд (инструкций) параллельно в каждом ядре, для увеличения производительности. Именно для данной цели начиная с процессоров серии Pentium (от компании Intel, и аналогичных) производителем было внедрено так называемое «суперскалярное» выполнение команд, обеспечивающее выполнение одновременно двух инструкций за один такт работы центрального процессора. В серии процессоров Pentium Pro было добавлено внеочередное выполнение инструкций. Предложенная архитектура обладала множеством недостатков. Проблема заключается в том, что при параллельном выполнении последовательного потока команд имеются определённые базисные ограничения. При увеличении количества выполняющих блоков не удавалось добиться кратного увеличения скорости.

В основе же архитектуры видеочипа уже заложена идеология параллельной работы микрокоманд. Каждое видеоядро получает на вход группу полигонов, после чего производит множество необходимых операций и на выход передаёт обработанные пиксели. Существенным моментом в данном случае является тот факт, что обработка множества полигонов и обработка пикселей независимы друг от друга, они могут быть обработаны отдельно друг от друга, параллельным образом. Именно по этой причине изначальной параллельной организации архитектуры GPU может быть использовано одновременно значительное количество исполняющих блоков, которые могут быть загружены гораздо меньшими усилиями, чем это происходит в архитектуре CPU при последовательном обрабатываемом потоке инструкций. Более того в современных GPU реализована технология «суперскалярного» выполнения команд, о которой говорилось

ранее, тем самым возможно выполнение более одной команды за такт (dual issue). Например, в архитектуре графических процессоров Tesla в некоторых случаях запускается исполнение операций MUL и MAD или SFU и MAD одновременно.

Отметим, что CPU отличается от GPU также и по принципу доступа к виртуальной памяти. В графических чипах доступ к памяти предсказуем и является связным. Например, когда из памяти видеоадаптера считывается текстель текстуры, то естественно, что через некоторый промежуток времени последует обработка соседних текстелей. Аналогичным образом происходит и запись в видеопамять – когда пиксель записывается в фреймбуфер, то естественно, что через некоторое количество тактов будут записываться пиксели, находящиеся по соседству. Исходя из сказанного можно сделать выводы, насколько принципиально различны архитектуры организации памяти центрального и графического процессоров. Для видеоадаптеров в отличие от центральных процессоров кэш-память большого размера практически не требуется, она лишь необходима для считывания текстур. В видеоадаптерах для этого предусматривается до 128-256 килобайт.

В работе с памятью CPU и GPU отметим следующие основные отличия. Во-первых, не каждый центральный процессор имеет встроенный контроллер памяти, а каждый графический процессор обязан иметь как правило несколько контроллеров. Например, в адаптере NVIDIA GT200 использовано до восьми 64-битных каналов. В связи с тем, что видеоадаптер является одним цельным устройством, он обладает и более скоростной памятью, чем в разделённой архитектуре CPU. Тем самым видеоадаптеру доступна на порядок высшая пропускная способность памяти. Для параллельных расчётов обработки графической информации, в которых часто используются большие массивы данных, пропускная способность памяти играет критичную роль в вопросе производительности и параллельности таких расчётов.

В универсальных центральных процессорах большое количество транзисторов и объем чипа тратится на буфер команд, на предсказание аппаратного ветвления, а также размер чипа расходуется на колоссальные объёмы чиповой памяти – кэш. Указанные аппаратные блоки необходимы для увеличения производительности выполнения

немногочисленного количества потоков инструкций. В свою очередь в видеочипах транзисторы расходуются в значительной степени на исполнительные блоки, на блоки управления потоков, на общую память сравнительно небольшого объёма и на многоканальные контроллеры памяти. Описанная структура не даёт прироста производительности для отдельных потоков, но даёт возможность обработки чипом сотен тысяч потоков, одновременно выполняющихся на одном чипе и требующих высокую пропускную производительность памяти.

В процессе кеширования следует отметить следующие отличия. В универсальных центральных процессорах используется кэш-память, увеличивающая производительность благодаря снижению задержек во время доступа к памяти, в свою очередь в GPU кэш память используется для увеличения объёма информации пропускания шины. В CPU задержка при доступе к памяти снижается с помощью памяти кэша сравнительно большого размера, в том числе, благодаря предсказанию ветвления кода. Данные аппаратные части занимают значительную часть площади устройства чипов потребляя большое количество энергии. В видеоадаптерах подобные проблемы задержек доступа к памяти обходятся благодаря одновременному исполнению большого количества потоков. Когда каждый из потоков ожидает на вход данных из запоминающего устройства, видеочип выполняет вычисление другого потока без задержек и ожидания.

Среди множества различий в поддержке многопоточности чипов. CPU выполняет несколько потоков в каждом процессорном ядре, а видеоадаптер поддерживает до 1024 потоков для каждого мультипроцессора, которых в адаптерах содержится по несколько штук. И в случае, когда при переключении потоков для CPU используется большое количество тактов, то для GPU аналогичное переключение использует один такт для переключения нескольких потоков.

Кроме этого, универсальные так как универсальные центральные процессоры используют архитектуру SIMD (одна команда выполняется над многими данными) блоки для вычислений векторного типа, а видеоадаптеры используют соответственно технологию SMT (единая инструкция на несколько потоков) для скалярного

обработки потоков. В архитектуре SIMT нет необходимости преобразования данных в вектор разработчиками и также допускается произвольное ветвление в потоках.

Резюмируя описанное, можно сказать, что отличием современного универсального процессора CPU от графического процессора GPU, является то, что GPU больше предназначен для параллельных вычислений с одновременным выполнением большого количества арифметических операций. В GPU основное количество транзисторов работает по их прямому назначению обработки данных, а не управлением использованием немногочисленного количества последовательных потоков вычислений (flow control). Далее представим схему объёма используемого пространства для разнообразной логики в CPU и GPU:

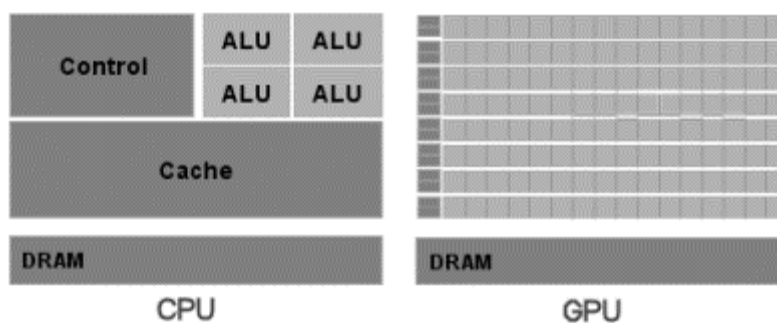


Рис.5. Различия архитектур микрочипов CPU и GPU

Резюмируя, можно сказать, что для эффективного использования мощности GPU в научных и схожих не обязательно графических целях, является наличие возможности распараллеливание процесса выполнения алгоритма на множество исполнительных блоков, которые доступны в видеочипе. Например, множество приложений для молекулярного моделирования отлично приспособляется для расчёта на видеоадаптерах, так как они требуют одновременного большого количества вычислительной мощности тем самым удобны для преобразования в параллельные вычисления. Подобные вычисления наглядно продемонстрированы в работах института информатики и автоматизации РАН. [100], [101]

В алгоритмах же в которых происходит распараллеливание процесса обработки изображений уже заложена архитектура процесса распараллеливания, предложенная в графических адаптерах так как они разрабатывались именно для таких целей – обработка графической информации, в которой необходимо одновременная большая вычислительная мощность. В современных системах так же возможно использование одновременно нескольких графических процессоров GPU (например, технология SLI), что даёт ещё большие вычислительные возможности при решении поставленных задач.

При выполнении расчётов на графическом процессоре были получены внушительные результаты в алгоритмах обработки графической информации, в которых активно используется параллельная обработка данных. Обоснованием увеличения производительности является то, что при обработке одной и той же последовательности математических операций к большому объёму данных часто встречаются одинаковые инструкции, которые одновременно выполняются в нескольких арифметических блоках видеочипа, что и является особенностью архитектуры SIMT. При этом следует отметить, что наилучшие результаты могут быть достигнуты в том случае, если количественное соотношение арифметических операций к числу обращений к памяти сравнительно велико. При таком количественном соотношении предъявляются меньшие требования к блоку «flow control» (управления исполнением), а большой объём данных, и большая плотность математических операций уменьшает необходимость присутствия кэша большого размера, которые используются в CPU.

В результате описанных выше отличительных характеристик, производительность видеоадаптеров в процессе обработки графической информации выше производительности CPU. Компанией NVIDIA была опубликована следующая информация роста производительности GPU относительно GPU за последние годы:

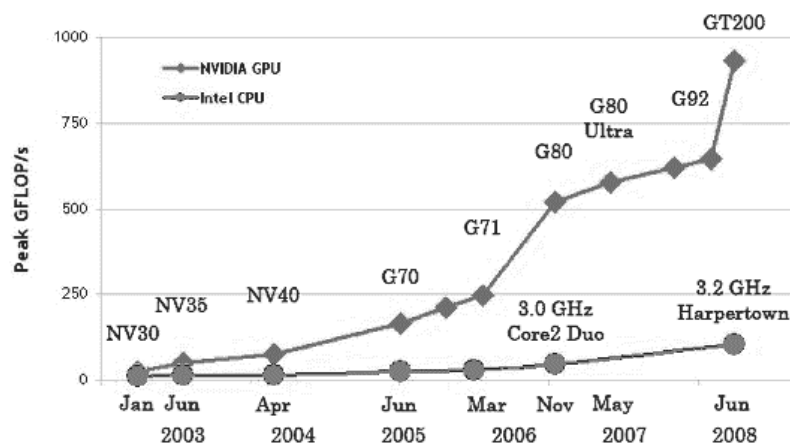


Рис.6. Рост производительности CPU по сравнению с GPU.

Формат «Год» - «Производительность» [102]

Следует также учитывать, что на CPU на практике гораздо легче достигается теоретический предел точности, а также числовые показатели приведены в случае GPU – одинарной точности, а для CPU – двойной точности. Но в любом случае, для параллельных вычислений одинарной точности достаточно, а разница в производительности между CPU и GPU значительно велика.

Попытки использования видеоадаптеров в математических расчётах присутствовали сравнительно давно. Самые первые из попыток такого рода были довольно примитивными и были ограничены использованием нескольких аппаратных функций, таких, как Z-буферизирование и растеризирование. Но на сегодняшний день, с появлением шейдерных блоков, начали ускоряться вычисления числовых матриц. Начиная с 2003 года на конференции SIGGRAPH была выделена отдельная секция для вычислений на GPU, и она получила название GPGPU (GPU, GPGP, General Purpose Computing for Graphics Processing Unit, специальные вычисления на графических процессорах).

Среди наиболее известных компиляторов следует отметить компилятор BrookGPU. Это компилятор созданный для выполнения не обязательно графических вычислений на видеоадаптерах. Данный компилятор компилирует исходный код потокового языка программирования под названием Brook. До появления данного компилятора перед

разработчиками стоял выбор между не столь разнообразными распространёнными API: OpenGL или Direct3D. Подобный выбор сильно ограничивал применение GPU, так как в трёхмерной графике активно используются текстуры и шейдеры, информации о которых не имели специалисты по программированию параллельных вычислений, но без подобной информации было невозможно использование ядер и потоков. Язык Brook сильно облегчил проблематичность данной задачи. Данные потоковые расширения к языку программирования C, разрабатываемые Стенфордским университетом, прятали от разработчиков сложные трёхмерные API OpenGL и Direct3D и позволяли программирование на видеочипе, как на параллельном сопроцессоре. Компилятор обрабатывает файлы с расширением br (написанные на языке C++) преобразуя код привязывая его к библиотекам с поддержкой OpenGL, DirectX или (в то время) x86.

У данного компилятора Brook была масса недостатков. Но самое его появление вызвало большой прирост инициативы компаний производителей GPU ATI и NVIDIA в процессе доступа производства вычислений и программирования графических процессоров. Дальнейшее развитие данной инициативы кардинально изменило положение в области параллельных вычислений и положило основу новому сектору под названием – вычисления на графических видеоадаптерах.

Приводя историческую справку отметим что многие исследователи из проекта Brook влились в команду разработчиков компании NVIDIA, для представление аппаратно-программной стратегии параллельных вычислений. В результате и была получена система, благодаря которой нет необходимости использования трёхмерного графического API, а работа с аппаратных обеспечением осуществляется напрямую с использованием самого драйвера устройства. В результате усилий данной команды и появилась технология NVIDIA Compute Unified Device Architecture (CUDA) – аппаратно-программная архитектура для параллельных вычислений на NVIDIA GPU. Которая и была взята за основу при разработке модулей расширения для ускорения алгоритмов, разрабатываемых в данной работе, описанные в следующих главах.

В представленном параграфе использованы данные из следующих источников: [4]–[7], [82]

Г Л А В А II

ПОСТРОЕНИЕ ПРОСТРАНСТВЕННОГО АЛГОРИТМА САМОВНЕДРЕНИЯ ИЗОБРАЖЕНИЯ

2.1 Основные аспекты разработки алгоритма встраивания ЦВЗ методом самовнедрения

В данной главе будут описаны детали комбинированных пространственно-частотных методов встраивания дополнительной информации, и построения соответствующей модели встраивания ЦВЗ в изображение. Разрабатывается частотный алгоритм встраивания ЦВЗ в изображение, основанный на известном алгоритме линейного встраивания ЦВЗ [103].

В качестве дискретного изображения рассматривается объект-матрица $A = \{a_i^j\}$, где $1 \leq i \leq M, 1 \leq j \leq N$ и каждое значение a_i^j есть натуральное число, удовлетворяющее условию $1 \leq a_i^j \leq h$, где h – некоторая постоянная (рассматриваются черно-белые, серотонные изображения, где $h=255$. Аналогичным образом предложенный алгоритм может быть рассмотрен и для цветных дискретных изображений, в которых каждое a_i^j – тройка чисел от 1 до 255, а h есть тройка (255,255,255)). Параметры M, N, h определяют структуру рассматриваемого дискретного изображения. Аналогично определяется изображение ЦВЗ для встраивания $W = \{w_{kl}\}$.

Следует заметить, в качестве ЦВЗ W мы будем рассматривать изображение визуальное сходное с изображением исходником A (сходная с изображением, но не обязательно в точности та же самая), т.е. рассматриваемые далее методы будут

удовлетворять определению данному в первой главе и могут быть именованы методами самовнедрения.

Далее при извлечении ЦВЗ мы будем иметь некоторое представление об оригинальном изображении.

Рассмотрим линейные алгоритмы встраивания ЦВЗ. Стандартные линейные алгоритмы подразумевают следующее правило для алгоритма встраивания ЦВЗ:

$$\tilde{a}_i^j = a_i^j + \alpha * w_{kl} \quad (1)$$

где α некоторое действительное число-коэффициент встраивания ЦВЗ в изображение.

$\tilde{A} = \{\tilde{a}_i^j\}$ изображение со встроенным ЦВЗ.

Как видим описанный в первой главе метод встраивания LSB является частным случаем данного линейного алгоритма, при $\alpha = 1$.

Можно заметить что при достаточно больших значениях α встроенная информация в изображение будет заметна для человеческого глаза, например, при $\alpha = 255$ и $a_i^j = 0$ – черный цвет в изображении превращается в белый. Следовательно, для сохранения критерия необнаруживаемости ЦВЗ k должно быть достаточно мало для сохранения конспирации.

Также существует проблема переполнения цветового буфера изображения. Допустим имеем числовое значение. Допустим имеем белый цвет в конкретном пикселе изображения $a_i^j = 255$, $\alpha = 1$ а соответствующее значение водяного знака так же равно $w_{ij} = 255$. В данном случае имеем полную потерю информации о ЦВЗ т.к. максимальное значение $\tilde{a}_i^j \leq 255$. В литературе такие случаи называют необратимыми алгоритмами встраивания ЦВЗ.

Известны методы адаптивного встраивания ЦВЗ в которых учитываются особенности изображения источника для встраивания. Таким образом алгоритм динамично видоизменяется для конкретных участков изображения. В этом и заключается идея термина, адаптивность – адаптация к различным изображениям.

Приведём несколько существующих адаптивных алгоритмов встраивания ЦВЗ [104], [105].

В алгоритме [105] используется метод разбиения оригинального изображения на блоки. Для каждого из блоков ведутся соответствующие расчёты учитывающие особенности данного блока для наиболее оптимального встраивания ЦВЗ. Существует множество модифицированных методов, основанных на данном алгоритме.

Среди недостатков данного алгоритма основным является, то что сила встраивания ЦВЗ зависит от особенности изображения и не регулируется разработчиком. В результате сильно снижается область использования данного алгоритма.

Для обхода некоторых подобных проблем предлагается использование вместо формулы (1) использовать следующую формулу:

$$\tilde{a}_i^j = (1 - \alpha) * a_i^j + \alpha * w_{kl} \quad (2)$$

аналогично формуле (1) в данной формуле α некоторое действительное число-коэффициент встраивания ЦВЗ в изображение. $\tilde{A} = \{\tilde{a}_i^j\}$ изображение со встроенным ЦВЗ. Следует наложить ограничение на число-коэффициент α - $0 \leq \alpha \leq 1$. При таком подходе в зависимости от значения числа w_{kl} уменьшается или увеличивается число получившегося изображения $\tilde{a}_i^j$ со встроенным ЦВЗ тем самым обеспечивая адаптивность данного метода.

Сделаем небольшие уточнения относительно разрабатываемого метода. Для простоты возьмём ширину и высоту изображения ЦВЗ W кратную размерам изображения, в которое оно встраивается, это всегда возможно сделать, дополнив изображение ЦВЗ необходимым количеством нулей.

Делим изображение в которое встраивается ЦВЗ делим на $K \times L$ непересекающихся блоков. K, L – ширина и высота изображения ЦВЗ встраиваемого в изображение A . Получаем $M/K \times N/L$ непересекающихся блоков исходного изображения. Каждому блоку по правилу (2) ставится в соответствие число-пиксель ЦВЗ w_{kl} . Здесь может быть применено некоторое правило смешивания пикселей, т.е. при сопоставлении может быть сопоставлен пиксель ЦВЗ исходя из некоторого правила соответствия. Такой метод позволяет улучшить качественные характеристики алгоритма в

нескольких аспектах. Построение такого алгоритма сопоставления будет проведено позже в данной главе.

Пусть w_{kl} некоторый пиксель ЦВЗ, который встраивается в блок изображения $A_{kl} = \{a_{k'l'}\}$, где k', l' индексы блока k, l изображения A . Тем самым получаем следующую формулу получаемого изображения $\tilde{A}$:

$$\tilde{a}_{k'}^{l'} = (1 - \alpha) * a_{k'}^{l'} + \alpha * w_{kl}$$

где $\tilde{a}_{k'}^{l'}$ пиксель изображения $\tilde{A}$. Заметим, что получаемое изображение не меняет своих размеров относительно его оригинала. Т.е. в результате каждому пикселю оригинального изображения ставится в соответствие преобразованный пиксель нового изображения. Данный сопоставленный пиксель будет искажаться в соответствии с силой встраивания ЦВЗ α , тем самым для обеспечения критерия необнаруживаемости ЦВЗ в получаемом изображении значение α должно быть достаточно мало. Учитывая экспериментальную информацию можно получить те значения α , которые оптимальны для определённого контекста.

Таким образом получаем закодированное изображения $\tilde{A}$ с содержащимся в нем ЦВЗ W . Допустим изображение было подвергнуто атаке. Обозначим полученное изображение через $\tilde{A}^\tau$, где τ символизирует атаку.

При извлечении ЦВЗ из атакованного изображения $\tilde{A}^\tau$ используется метод наименьших квадратов (МНК).

Мы знаем, как выглядит искомая функция в данном методе. Обозначим ее через Ξ . Разность значений квадратов должна быть минимальной.

По методу МНК можно получить

$$\Xi(k, l) = \sum_{k'} \sum_{l'} \left(\tilde{a}_{k'}^{l'} - (1 - \alpha) * a_{k'}^{l'} + \alpha * w_{kl}^\tau \right)^2$$

где k', l' индексы пикселей блока k, l изображения.

Можем получить уравнение МНК:

$$-2\alpha * \sum_{k'} \sum_{l'} \left(\tilde{a}_{k'}^{l'} - (1 - \alpha) * a_{k'}^{l'} - \alpha * w_{kl}^\tau \right) = 0$$

Получаем:

$$w_{kl}^{\tau} = \frac{\mu_{kl}^{\tau} - (1 - \alpha) * \mu_{kl}}{\alpha}$$

где $\mu_{kl}, \mu_{kl}^{\tau}$ среднеарифметические цветовые значения пикселей блока k, l изображения

$$\mu_{kl} = \frac{KL}{MN} \sum_{k'} \sum_{l'} a_{k'}^{l'}$$

$$\mu_{kl}^{\tau} = \frac{KL}{MN} \sum_{k'} \sum_{l'} \tilde{a}_{k'}^{\tau l'}$$

Таким образом, мы получили возможность извлечения самого изображения.

2.2 Алгоритмы оценки качества изображений

Вернёмся к вопросу, который был оговорён ещё в первой главе. Приведём описание алгоритмов оценки качества изображений. Существует множество алгоритмов оценки качества изображений. Мы же используем такие алгоритмы, которые на вход получают два изображения и дают на выходе числовую характеристику сходства данных изображений.

Построим алгоритм оценки качества изображения, основанный на линейном расчёте количества различающихся пикселей. В качестве числовой характеристики будем принимать соответствие f сопоставляющее двум изображениям A и $\tilde{A}$ числовое действительное значение $l, 0 \leq l \leq 100$.

Иными словами f - функция представляющаяся следующим образом:

$$f(A, \tilde{A}) = l$$

При $f(A, \tilde{A}) = l = 100$ будем считать что изображения идентичны.

При $f(A, \tilde{A}) = l = 0$ не имеют ни малейшего сходства.

Реализация такого алгоритма не представляет особого труда. Допустим количество пикселей изображений $V(A) = V(\tilde{A}) = M \times N$.

Тогда при линейном сравнении соответствующих пикселей исходного и искажённого изображений $A, \tilde{A}$ получаем количество несовпадающих пикселей, обозначим через C .

Получаем:

$$f(A, \tilde{A}) = \frac{M \times N - C}{M \times N} * 100$$

Данная формула и представляет искомую числовую характеристику.

Недостатком данного алгоритма является, то что он не учитывает визуальных особенностей изображений. Например, два изображения могут на глаз не отличаться друг от друга, но данный алгоритм будет возвращать значение 0. Это может быть в том случае если изображения отличаются небольшим оттенком, не заметным для глаза.

Следует так же определить известные из литературы понятия сходства двух изображений, математически описываемые критерии расчёта среднеквадратического отклонения (MSE) и расчёта пикового отношения сигнала к шуму (PSNR), которые позволяют численно оценить насколько сходны два изображения. Имея два изображения, представляемые матрицами числовых значений $A = \{a_i^j\}$ и $A' = \{a'^j_i\}$ значение среднеквадратического отклонения и пиковое отношение PSNR будет выглядеть следующим образом:

$$PSNR = 10 \cdot \log_{10} \left(\frac{MAX_{A,A'}}{MSE^2} \right) = 20 \cdot \log_{10} \left(\frac{MAX_{A,A'}^2}{\sqrt{MSE}} \right) \text{ дБ, и } MSE = \frac{1}{MN} \cdot \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} |a_i^j - a'^j_i|^2$$

где $MAX_{A,A'}$ -динамический диапазон расхождения между значениями пикселей изображений A и A' . Критерий PSNR показывает на сколько сходны визуально изображения, представленные матрицами значений A и A' , измеряется в логарифмической шкале в децибелах. Стандартные значения PSNR для сжатия изображений лежат в интервале от 30 до 40 дБ. Показывается, что, если значения PSNR для изображений выше 30 дБ то тогда искажения не заметны для человеческого глаза¹.

¹ Nagar, Atulya, Mohapatra Proceedings of 3rd International Conference on Advanced Computing, Networking and Informatics ICACNI 2015, Volume 2 pp. 254 ISBN: 9788132225386 8132225384

На основании описанных критериев и будет производиться сравнение полученных результатов с оригиналом. Данные алгоритмы реализованы в системе, приложенной к диссертации. Об этом будет так же оговорено в главе IV.

2.3 Алгоритмы смешивания пикселей

Для обеспечения дополнительной безопасности внедряемого ЦВЗ используются алгоритмы или правила смешивания пикселей. В результате, если декодировщик не осведомлён о данном правиле, даже при успешном выделении ЦВЗ из изображения он получает несуразный набор пикселей. Так же метод может быть использован для увеличения качества изображения. Метод даёт наилучший результат в особенности для метода, разлагающего изображение в спектральное представление, так как в таком представлении в каждом блоке имеются повторяющиеся участки с наибольшими значениями. При смешивании данных областей равномерно по всему изображению они менее заметны для человеческих визуальных рецепторов.

В литературе за последние годы часто можно встретить использование метода преобразования Арнольда [106]–[110].

Данное преобразование в каком-то смысле схоже с использованием псевдослучайной последовательности для перемешивания пикселей ЦВЗ перед и после встраивания, которое так же описано в данной главе. Как правило, в литературе, данный процесс применяется для того [111], чтобы максимально усложнить декодирование встроенной информации в изображение (ЦВЗ) из возможно извлечённого образа неавторизованными лицами [108].

Далее представим сам процесс смешивания пикселей изображения методом преобразования Арнольда. Как было оговорено ранее о в данной главе имеем изображение представленное в виде матрицы чисел $A = \{a_i^j\}$, которое преобразуется в числовую матрицу-изображение $A' = \{a_{i'}^{j'}\}$ (в данном случае будем рассматривать

квадратные матрицы размера M , матрицу любого размера можно привести к квадратному виду добавлением необходимого количества нулей) по следующему правилу:

$$\begin{pmatrix} i' \\ j' \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix} * \begin{pmatrix} i \\ j \end{pmatrix} \bmod M$$

где $i, j = 0, 1 \dots M - 1$.

Процесс производится с помощью нескольких итераций т.е. нескольких преобразований указанного вида. Его можно представить с помощью следующего вида

$$P_{ij}^{n_{iter}} = B P_{ij}^{n_{iter}-1}$$

где $n_{iter} = 1, 2, \dots$;

$$B = \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix}; P_{ij} = (i, j)^T.$$

Для определённого количества итераций T , зависящего от размера рассматриваемого изображения можно восстановить начальный образ изображения. Другими словами, можно сказать, что алгоритм является обратимым и периодическим. Все же метод не лишён ряда недостатков, среди них следует отметить существенный недостаток данного метода, который заключается в невозможности однозначного задания числа периодов при котором изображение возвращается к своему первоначальному облику. Даже для квадратных изображений различного размера данный период существенно отличается. В литературе, однако можно встретить некоторые экспериментальные данные значения периода преобразования T , для изображений определённого размера. Например, в работе [112] получены результаты, представленные в следующей таблице:

Размер изображе- ния	60×60	100×100	120×120	128×128	256×256	480×480	512×512
T	60	150	60	96	192	240	384

Таблица 2. Приведены экспериментальные значения периода для алгоритма преобразования Арнольда в формате «Размер изображения» – «Период» [112].

Также можно встретить методы расчёта периода преобразования Арнольда для специальных случаев. Например, в [113] предлагается использование следующих формул при расчёте периода, при удовлетворении следующих критериев, при размере изображения $N \times N$

- Если $N = 2 * 5^k$, при $k = 1, 2 \dots$, то тогда:

$$T = 3 * N$$

- Если $N = 5^k$ или $N = 6 * 5^k$, при $k = 1, 2, \dots$, то тогда:

$$T = 2 * N$$

- Если изображение имеет другой размер для периода имеет место следующее неравенство

$$T \leq \frac{12 * N}{7}$$

Результатом экспериментов, в некоторых источниках также были получены следующие числовые закономерности для периода работы алгоритма Арнольда:

- Если $N = 2^k$, при $k = 2, \dots, 10$, то тогда:

$$T = \frac{3 * N}{4}$$

В результате можно проследить некоторую общую закономерность среди приведённых полученных экспериментальных данных. Несмотря на это общей формулы для расчёта периода преобразования Арнольда на момент написания данной работы получено не было.

Можно обеспечить дополнительный уровень безопасности применяемого метода, усложнив задачу злоумышленника, если применить преобразование Арнольда не ко всему изображению сразу, а к некоторым частям изображения по отдельности. Тем самым атакующему придётся узнавать размеры частей изображения. Причиной столь широкого

распространения данного метода является простота его реализации в особенности к приложениям встраивания ЦВЗ.

Ещё одним значительным недостатком данного метода является то, что при кодировании изображения частотными алгоритмами сохраняется относительное расположение чётности соседних элементов друг другу. Скажем, если на всех чётных местах в матрице содержится число 4, тогда после применения данного метода на чётных местах так же будет содержаться число 4. В частотных же алгоритмах наивысшие значения достигаются на чётных позициях, и после применения преобразования Арнольда заметны яркостные скачки с периодичностью на чётных позициях. Тем самым может быть нарушен критерий необнаруживаемости встроенного ЦВЗ пространственно-частотными методами. Для этого предлагается использование метода, основанного на генерации псевдослучайных чисел PRNG (pseudorandom number generator), где за основу взят линейно-конгруэнтный метод. Д.Г. Лемер опубликовал данный метод в 1949 году [114].

Метод заключается в следующем: вычисляется численная последовательность X_n , где выполняется следующее соотношение:

$X_{n+1} = (aX_n + c) \bmod m$, где m – количество значений из которых формируется последовательность $m \geq 2$

$\bmod$ – остаток от деления

a – множитель $0 \leq a < m$

c – приращение $0 \leq c < m$

X_0 – начальное значение $0 \leq X_0 < m$

Такая последовательность X_n называется линейно конгруэнтной последовательностью.

Например, для $m = 10, X_0 = a = c = 7$ получаем следующую последовательность [115]

7, 6, 9, 0, 7, 6, 9, 0 ...

Линейная конгруэнтная последовательность, определённая числами m, a, c, X_0 повторяется с некоторым периодом, с периодом не более чем m . Частота периода определяется числом m тогда и только тогда, когда [115]

- числовые значения m и c взаимно просты
- $a - 1 = b$ кратно числу p для каждого простого числа p , которое есть делитель числа m
- если m кратно четырем то и число b кратно четырём,

Наличие этого свойства для случая $m = 2^e$, где e – число битов в машинном слове, было доказано М. Гринбергом [116]. Присутствие данного свойства в общем случае и условие достаточности было доказано Т.Е. Халлом и А.Р. Добеллом [117].

Метод называется мультипликативным тогда, когда генерация конгруэнтной последовательности осуществляется, при $c = 0$. В случае когда $c \neq 0$ определяется как смешанный линейный конгруэнтный метод. Когда $c = 0$ получаемые числа имеют небольшой период, сравнительно со случаем когда $c \neq 0$, однако при специальном задании последовательности можно получить частоту повторения длины $m - 1$, когда m является простым числом. Факт того, что $c \neq 0$ приводит к тому что появляются периоды большей длины. Он был установлен Томсоном В.Е., а также Ротенбергом А., независимо [115]. Для гарантирования наибольшего цикла повторений последовательностей при $c = 0$, требуется в качестве значений параметров m выбирать простые числа. Наиболее известный генератор подобного рода был предложен Миллером и Парком в январе 1988 года. Генератор называется минимальным стандартным генератором случайных чисел. Он определяется числами $m = 2147483647$ и $a = 16807$ [118], [119].

Часто в литературе встречается так называемый смешанный конгруэнтный метод, генерирующий последовательность псевдослучайных чисел.

Когда коэффициенты линейного конгруэнтного метода постоянны позволяет обобщить его на случай, в которых коэффициенты, c и a различные. Когда переменные коэффициенты различные линейный конгруэнтный метод задаётся следующим уравнением:

$$X_n = (a_{n-1}X_{n-1} + c_{n-1}) \bmod m$$

Из случая переменных коэффициентов линейно конгруэнтного метода получаются его известные разновидности:

Квадратичным конгруэнтным методом называется, при

$$X_n = (aX_{n-1}^2 + bX_{n-1} + d) \bmod m$$

Кубический конгруэнтный метод:

$$X_n = (aX_{n-1}^3 + bX_{n-1}^2 + cX_{n-1} + d) \bmod 2^e$$

Регулярный рандомизированный метод с коэффициентами, зависящими от предыдущих элементов последовательности:

$$X_n = (A_{n-1}X_{n-1} + C_{n-1}) \bmod m$$

Среди особенностей характерных регулярному рандомизационному методу следует отметить зависимость при переходе нелинейного участка от изначальных условий, последовательность за счёт саморегуляции, достигает наибольшего периода m и далее на границе каждого из периодов ведёт себя без повторений [120].

Нам наиболее подходит последний предложенный метод с внесением смешанных характеристик. Линейный конгруэнтный метод имеет ряд недостатков с криптографической точки зрения. Для этого мы внесём ряд поправок в его реализацию. Поправки сходны по своей идеологии с инверсивным конгруэнтным методом. Далее приведён листинг соответствующего модуля генерации псевдослучайной последовательности на языке C++:

```
static unsigned int seed = 1;

void srand (int newseed) {
    seed = (unsigned)newseed & 0x7fffffffU;
}

int rand (void) {
    seed = (seed * 1103515245U + 12345U) & 0x7fffffffU;
    return (int)seed;
}
```

}

Переменная `seed` в данном случае отвечает за определение всей последовательности псевдослучайных чисел, таким образом обладая данным числом, можно восстановить всю исходную псевдослучайную последовательность. Рекомендуется использование достаточно больших чисел в качестве переменной `seed`.

Основой для криптографической стойкости будет послойный подход генерации дайной последовательности, а именно в качестве случайной составляющей будет играть роль позиции пикселя изображения.

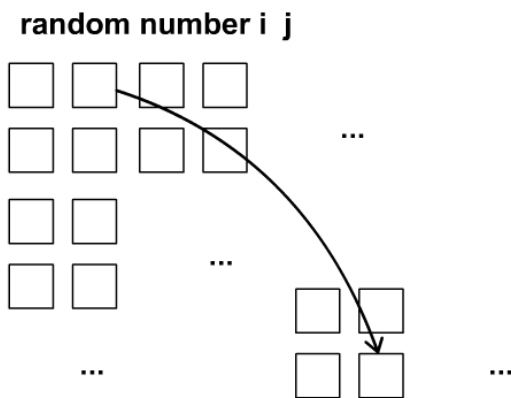


Рис.7. Создание смешивающей матрицы

Строится матрица переходов. Каждый элемент матрицы содержит универсальный номер элемента данной матрицы. В начале каждый элемент указывает сам на своё местоположение, это означает что перемешивания пока не было. Для каждого числа генерируются 2 случайных числа его новой позиции. После чего значения транспонируются, тем самым указывая на новое местоположение элемента.

Данную матрицу можно назвать матрицей перехода или матрицей, смешивающей пиксели. При применении данной на изображение пиксели смешиваются в случайном порядке.

Продemonстрируем принцип работы алгоритма смешивания пикселей:

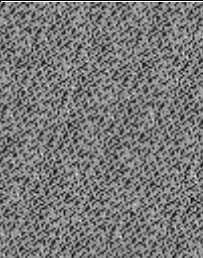
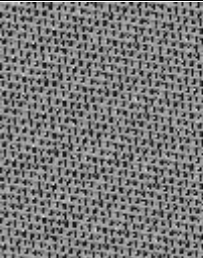
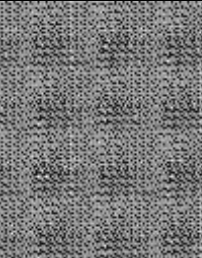
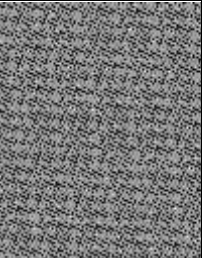
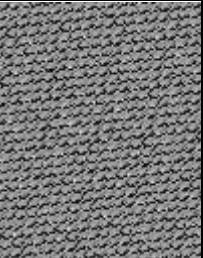
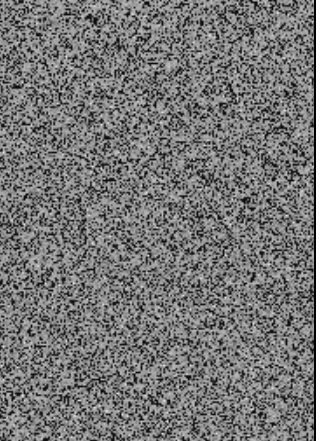
Количество итераций 0	Количество итераций 8	Количество итераций 16	Количество итераций 24	Количество итераций 32	Количество итераций 40
					

Таблица 3. Демонстрация работы одного из алгоритмов смешивания пикселей

Оригинальное изображение	10 % пикселей преобразованы с помощью псевдослучайного правила	50% пикселей преобразованы с помощью псевдослучайного правила	100% пикселей преобразованы с помощью псевдослучайного правила
			

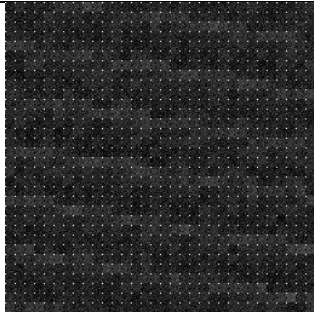
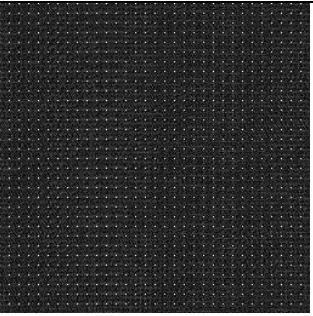
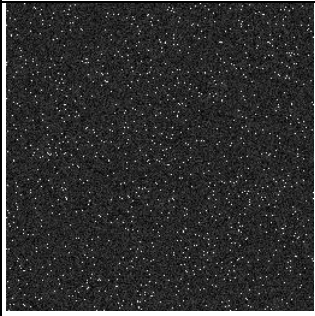
Частотное представление изображения	Частотное изображение преобразованы с помощью преобразования Арнольда (шаг 32)	Частотное изображение преобразованы с помощью псевдослучайного правила (100% п.)	-
			-

Таблица 4. Демонстрация работы псевдослучайного алгоритма смешивания пикселей

2.4 Пространственно-частотный алгоритм самовнедрения изображения

Первые шаги схожи с линейным методом встраивания ЦВЗ, описанным ранее в данной главе. Вновь введём информацию об объектах, которые будут использоваться далее в данном методе. В качестве изображения рассматривается объект-матрица $A = \{a_i^j\}$, где $1 \leq i \leq M, 1 \leq j \leq N$ и каждое значение a_i^j есть натуральное число, удовлетворяющее условию $1 \leq a_i^j \leq h$, где h – некоторая постоянная (рассматриваются черно-белые, серотонные изображения, где $h=255$). Аналогичным образом предложенный алгоритм может быть рассмотрен и для цветных дискретных изображений, в которых каждое a_i^j – тройка чисел от 1 до 255, а h есть тройка (255,255,255)). Параметры M, N, h определяют структуру рассматриваемого дискретного изображения. Аналогично определяется изображение ЦВЗ для встраивания $W = \{w_{kl}\}$.

Следует заметить, что в качестве ЦВЗ W мы будем рассматривать изображение, визуально сходное с изображением источника A (сходное с изображением, но не

обязательно в точности то же самое), т.е. рассматриваемые далее методы будут удовлетворять определению, данному в первой главе и могут быть именованы методами самовнедрения.

Далее, при извлечении ЦВЗ мы будем иметь некоторое представление об оригинальном изображении.

Пусть $W = \{w_{kl}\}$ – матрица ЦВЗ для встраивания, сходная, с оригинальным изображением A , размерами $K \times L, k = 0, 1, \dots, K - 1, l = 0, 1, \dots, L - 1$. Рассматриваются K и L кратные M и N соответственно (среди сходных алгоритмов следует отметить работу [121]). Далее ЦВЗ делится на блоки размером $b \times b$ пикселей, где b может принимать значения 4, 8 и 16, основываясь на размере изображения (такое разбиение вида 2^n необходимо в связи со спецификой дальнейшего быстрого декодирования, можно использовать и другие значения дополнив оставшуюся часть нулями). Для каждого из выделенных блоков ЦВЗ рассчитывается матрица коэффициентов двумерного Дискретного Косинусного Преобразования (ДКП, DCT – Discrete Cosine Transform). Существует несколько разновидностей ДКП в литературе мы же будем использовать разновидности DCT-II и обратное преобразование DCT-III.

$$X_k^{DCTII} = \sum_{n=0}^{N-1} x_n \cos \left[\frac{\pi}{N} \left(n + \frac{1}{2} \right) k \right] \quad k = 0 \dots N - 1$$

$$X_k^{DCTIII} = \frac{1}{2} x_0 + \sum_{n=1}^{N-1} x_n \cos \left[\frac{\pi}{N} n \left(k + \frac{1}{2} \right) \right] \quad k = 0 \dots N - 1$$

аналогичным образом обобщается 2D DCT-II и 2D DCT-III

$$X_{k_1, k_2}^{2DDCTII} = \sum_{n_1=0}^{N_1-1} \sum_{n_2=0}^{N_2-1} x_{n_1, n_2} \cos \left[\frac{\pi}{N_1} \left(n_1 + \frac{1}{2} \right) k_1 \right] \cos \left[\frac{\pi}{N_2} \left(n_2 + \frac{1}{2} \right) k_2 \right]$$

$$X_{k_1, k_2}^{2DDCTIII} = \frac{1}{2} x_{0,0} + \sum_{n_1=0}^{N_1-1} \sum_{n_2=0}^{N_2-1} x_{n_1, n_2} \cos \left[\frac{\pi}{N_1} n_1 \left(k_1 + \frac{1}{2} \right) \right] \cos \left[\frac{\pi}{N_2} n_2 \left(k_2 + \frac{1}{2} \right) \right] \quad n_1 \neq 0 \text{ and } n_2 \neq 0$$

Для некоторых практических приложений рассматривается также двумерное Быстрое Преобразование Фурье (БПФ, FFT). На вход преобразование получает значения с заданной вещественной частью, мнимая же часть содержит нулевые значения. В

результате преобразования получаем матрицу мнимых чисел, размера $b \times b$ (по размеру величины блока разбиения ЦВЗ), в которой в действительной части в интервале $\left[(0,0), \left(b, \frac{b}{2}\right)\right]$ принимает значения, равные значениям в интервале $\left[\left(0, \frac{b}{2}\right), (b, b)\right]$. Аналогично получаем значения в мнимой матрице с тем отличием, что значения в интервалах $\left[(0,0), \left(b, \frac{b}{2}\right)\right]$, $\left[\left(0, \frac{b}{2}\right), (b, b)\right]$ принимают значения равные по модулю. Для восстановления матрицы действительной составляющей, в каждом блоке записываются значения в интервале $\left[(0,0), \left(b, \frac{b}{2}\right)\right]$, а для восстановления матрицы с мнимыми коэффициентам записываются значения в интервале $\left[(0,0), \left(b, \frac{b}{2}\right)\right]$ и маска значений $(-1)^n$ отображающая первую половину значений матрицы на $\left[\left(0, \frac{b}{2}\right), (b, b)\right]$, длиной $\frac{b}{2} \cdot b$, по сути представляющую собой маску битовых значений. Например, если размер блока равен 8, то объём маски для блока будет занимать 32 бита = 4 байта. При размере изображения 1024 x 1024 размер маски при размере блока 8 без сжатия будет составлять не более 64 кб. В результате одного из преобразований ДКП или БПФ, получаем матрицу коэффициентов интенсивности спектра пикселей водяного знака.

Каждый блок представляется следующим видом $S = (S_i^j) \quad i, j = 0, 1 \dots b - 1$
 $t \times t$ коэффициентов спектрального преобразования каждого блока сохраняются, остальные игнорируются.

Оставшиеся коэффициенты каждого блока выписываются в сжатую матрицу коэффициентов спектрального представления ЦВЗ $S^{comp} = (S_{k',l'}^{comp})$ размером $K' \times L'$, где $(l' = 0, 1 \dots L' - 1, \quad k' = 0, 1 \dots K' - 1)$

Целые числа из интервала $[0 \dots 255]$ присваиваются каждому элементу матрице S^{comp} по следующей формуле

$$s'_{k',l'} = \left\lceil 255 \cdot \frac{S_{k',l'}^{comp} - Min}{Max - Min} \right\rceil, (1)$$

где Max, Min соответственно максимальные и минимальные элементы матрицы S^{comp} .

Полученную матрицу (1) обозначаем через SPW и называем интенсивностью спектрального шаблона водяного знака (Spectral Pattern Watermark).

Приведём непосредственно алгоритм встраивания SPW в изображение.

Основываясь на идее алгоритма линейного встраивания ЦВЗ в изображение, произведём встраивание. Для встраивания ЦВЗ (SPW) в исходное изображение $A = \{a_i^j\}$ размером $M \times N$; $m = 0,1 \dots M - 1$; $n = 0,1 \dots N - 1$, изображение делится на $K' \times L'$ блоков A_ξ размером $(M/K') \times (N/L')$, где $\xi = 0,1 \dots (K' \times L') - 1$. Если K' и L' не множители M и N , то тогда матрица спектральных значений пикселей заполняется соответствующим количеством нулей. Производится смешивание пикселей исходного изображения с преобразованным изображением водяного знака. Рассмотрим вопрос соотношения качества водяного знака при декодировании, если вместо линейного взаимно однозначно соответствия пикселей используется некоторое правило Θ , о котором говорилось ранее в данной главе, устанавливающее соответствие между пикселями исходного изображения и ЦВЗ. Соответствие устанавливается на нескольких этапах. Первый этап производит смешивание пикселей перед обработкой ЦВЗ. Второй этап производится после спектральной обработки ЦВЗ. Подобный подход двухэтапного смешивания показывает улучшенные показатели качества ЦВЗ после его восстановления. Такое смешивание также производится как для увеличения секретности.

В случае, если в качестве Θ берётся псевдослучайное правило, описанное ранее в данной главе, то тогда можно заметить следующую закономерность при его использовании:

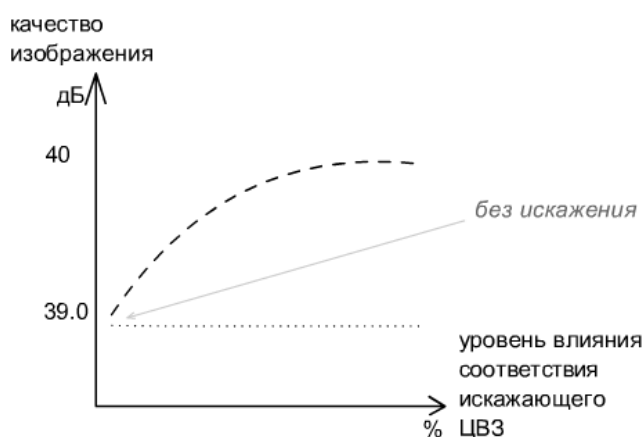


Рис.8. Показатель качества относительно мощности перемешивания пикселей

Как говорилось ранее, псевдослучайное правило строится на основании некоторого ключа, описывающего данное псевдослучайное правило. Таким образом, только в случае обладания ключом можно расшифровать информацию, закодированную в исходном изображении, что увеличивает безопасность встраиваемой информации. Ещё раз подчеркнём, что необходимо использование достаточно больших значений в качестве ключа подобного рода для усложнения дальнейшего его несанкционированного декодирования методом примитивного перебора.

Допустим w'_ξ ξ -ый пиксель водяного знака SPW . Пусть w'_ξ встроен в блок $A_\xi = \{a_{m_\xi n_\xi}\}$. В данной главе рассматривается следующее правило встраивания ЦВЗ в изображение:

$$a_{m_\xi n_\xi}^{SPW} = (1 - \alpha)a_{m_\xi n_\xi} + \alpha w'_\xi$$

где $a_{m_\xi n_\xi}^{SPW}$ соответствующий пиксель изображения A^{spw} в которое встраивается ЦВЗ. Значение α здесь достаточно малое действительное число, лежащее в интервале $[0,1]$ и представляющее собой мощность встраивания водяного знака в изображение. α едино для всех блоков встраиваемого изображения. При высоких значениях α оригинальное изображение сильно искажается в связи с этим используются сравнительно не большие значения α . В среднем подходящие значения лежат в отрезке $(0, \frac{1}{2})$.

При извлечении ЦВЗ имеем изображение со встроенным водяным знаком и возможно подвергнутое атаке. Обозначим матрицу описывающую такое изображение через $A^{spw\tau} = \{a_{ij}^{spw\tau}\}$, где τ – символизирует отображение-атаку на изображение.

Изображение $A^{spw\tau} = \{a_{ij}^{spw\tau}\}$ делится на блоки по количеству пикселей встроенного водяного знака. Обозначим среднюю интенсивность пикселей блоков изображения-контейнера, через $\{\mu_\xi\}$, где

$$\mu_\xi = \frac{KL}{MN} \sum_{m_\xi} \sum_{n_\xi} a_{m_\xi n_\xi}$$

Обозначим среднюю интенсивность пикселей ξ -ого блока атакованного изображения водяного знака через

$$\mu_{\xi}^{spwX} = \frac{KL}{MN} \sum_{m_{\xi}} \sum_{n_{\xi}} a_{m_{\xi}n_{\xi}}^{spwX}$$

Приближение элемента SPW с присутствующей атакующей составляющей τ вычисляется по следующей формуле

$$w_{\xi}^{'X} = \frac{\mu_{\xi}^{spwX} - (1 - \alpha)\mu_{\xi}}{\alpha}.$$

Приведённая формула получается методом наименьших квадратов так же как в параграфе 2.1.

Полученная информация трансформируется следующим образом:

$$S_{k'l'}^{comp} = w_{k'l'}' \frac{(Max - Min)}{255} + Min$$

Матрица с полученными элементами $S_{k'l'}^{comp}$ делится на блоки размером $t \times t$, каждый блок заполняется необходимым количеством нулей до размера $b \times b$.

Каждый полученный блок размера $b \times b$ подвергается обратному спектральному преобразованию в зависимости от применённого преобразования.

Полученные данные комбинируются в матрицу и конвертируются в изображение исходного размера водяного знака.

После преобразований пиксели изображения подвергаются обратному смешивающему преобразованию в соответствии с алгоритмом смешивания применённым во время встраивания ЦВЗ.

ГЛАВА III

МЕТОД СМЕШАННОГО КОДИРОВАНИЯ И ВСТРАИВАНИЯ ЦВЗ

3.1 Метод отклонений (разностей)

В качестве изображения рассматривается объект-матрица $A = \{a_i^j\}$, где $1 \leq i \leq M, 1 \leq j \leq N$ и каждое значение a_i^j есть натуральное число, удовлетворяющее условию $1 \leq a_i^j \leq h$, где h – некоторая постоянная (рассматриваются черно-белые, серотонные изображения, где $h=255$). Аналогичным образом предложенный алгоритм может быть рассмотрен и для цветных дискретных изображений, в которых каждое a_i^j – тройка чисел от 1 до 255, а h есть тройка $(255, 255, 255)$. Параметры M, N, h определяют структуру рассматриваемого дискретного изображения.

Как было указано выше, имеем матрицу изображения $\{a_i^j\}$, допустим начальный элемент матрицы a_0^0 лежит в левом верхнем углу матрицы, кодирование начинается именно с данного элемента. Для дальнейшего облегчения вычислений преобразуем матрицу $\{a_i^j\}$ к виду, в котором каждый элемент матрицы является четным числом, иными словами, каждый элемент матрицы заменяется самым близким меньшим четным числом. Легко убедиться в том, что при таком преобразовании изменения в изображении не будут ощутимы человеческим глазом. Обозначим полученную матрицу через $\{b_i^j\}$. Для каждого b_i^j рассмотрим разность $c_i^j = (b_i^{j+1} - b_i^j)$. Заметим, что каждый элемент b_i^j будет принимать значения в интервале $[-255, 255]$, так как мы условились рассматривать серотонные изображения. Тем самым, если мы возьмём матрицу $\left\{\frac{c_i^j}{2}\right\}$, то значения будут

лежать в интервале $[-127, 127]$, причем каждое $\frac{c_i^j}{2}$ будет целым числом, т.к. c_i^j – четное (в чем легко убедиться). Теперь прибавим к каждому $\frac{c_i^j}{2}$ число 127, тем самым получим третью матрицу $\{e_i^j\}$, где $e_i^j = \left(\frac{c_i^j}{2} + 127\right)$ значения которой лежат в интервале $[0, 255]$. Важно заметить, что для дальнейшего восстановления исходного изображения, кодируются все строки кроме последней, т.е. для $i \neq M$. Полученная матрица $\{e_i^j\}$ и есть закодированная матрица методом отклонений. Примечательно, то что тем самым кодируются только отклонения (или разности), тем самым уменьшая численный разброс цветовой гаммы. Т.е. информация сжимается в случае, если разброс не слишком велик. Например, перекодирование изображения, в котором чередуются строки со значениями 0 и 255, не имеет смысла, так как в закодированном виде оно ничем не будет отличаться от оригинала, в чём легко убедиться. Однако такое изображение практически не встречается среди электронных фотографий, для наглядности ниже приводится пример такого изображения (см. Рис.9.):

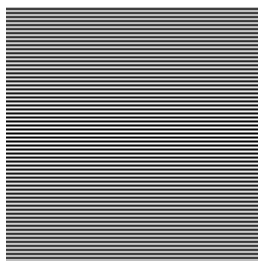


Рис.9. Пиксели со значениями оттенков 0 и 255 распределённые по строчно, первая строка заполнена цветовым оттенком 255, вторая оттенком 0 и так далее...

Замечание. Выше мы рассмотрели кодирование методом отклонений по оси Y, но ни что не мешает кодировать разности и по оси X. Закодированное таким образом изображение примет следующий вид:



Рис.10. Пиксели со значениями оттенков 0 и 255 распределённые по строчно, закодированные методом отклонений по оси Y

Таким образом получается небольшой прирост в экономии свободного места по сравнению с кодированием по оси Y, которое не меняет данное изображение.

Декодирование производится аналогичным методом. Для каждого элемента матрицы $\{e_i^j\}$ считается следующая сумма $a_i^j = e_{i+1}^j + (e_i^j - 127) * 2$ для каждого элемента начиная с $i = M - 1$ до $i = 0$. Таким образом, изображение возвращается к своему исходному виду.

Применим данный алгоритм к изображению – «Ararat».



Рис.11. «Ararat» оригинал

изображение закодировано форматом gif размером в 83.1 КВ.

После применения алгоритма кодирования изображение выглядит следующим образом:

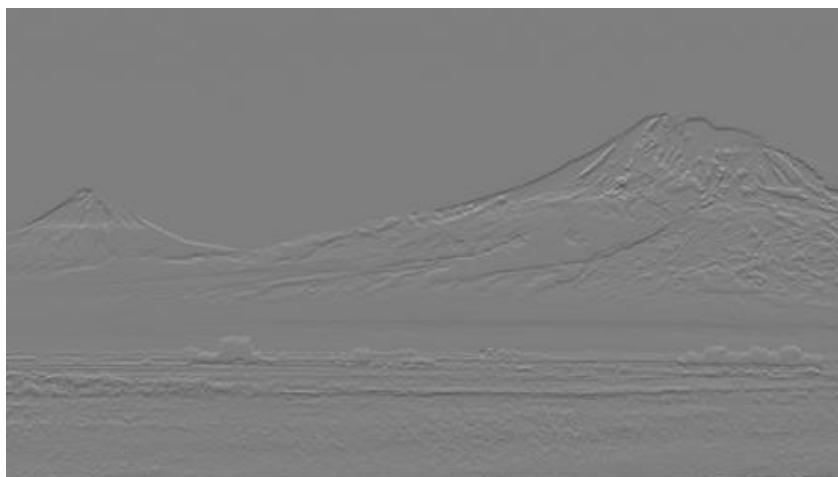


Рис.12. «Ararat» закодирован методом отклонений

Как видно из рис. 12 новое изображение в основном представлено в средних тонах. Далее, если сохранить изображение в исходном формате gif, оно займёт 53.5 KB тем самым обеспечивая сжатие в 35% относительно оригинала.

Применяя алгоритм декодирования, возвращаем изображение к исходному виду:



Рис.13. «Ararat» после декодирования

Посчитаем значение среднеквадратического отклонения, определённые ранее, для оригинального и декодированного изображения, оно будет выглядеть следующим образом:

$$PSNR = 10 \cdot \log_{10} \left(\frac{MAX_{A,A'}}{MSE^2} \right) = 20 \cdot \log_{10} \left(\frac{MAX_{A,A'}^2}{\sqrt{MSE}} \right) \text{ дБ}, \quad MSE = \frac{1}{MN} \cdot \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} |a_i^j - a_i'^j|^2.$$

где A оригинальное изображение A' декодированное. В результате получаем отклонение, равное **51.13 дБ**. Как было указано в первой главе и в работах [122], [123] – знаем, что искажения заметны для глаза при значениях PSNR меньших чем 30-40 дБ,. Следовательно, качества хватает с избытком.

Рассмотрим следующий пример крупного изображения 2508x1254 с множеством деталей – «Ruins» (by Roland Yeghiazaryan, изображение используется с согласия автора²):



Рис.14. «Ruins» оригинал

размер 496 КВ. После конвертации оно примет следующий вид:

² <https://plus.google.com/+RolandYeghiazaryan/photos/photo/5913759708537069538?pid=5913759708537069538&oid=108391226186149093445>

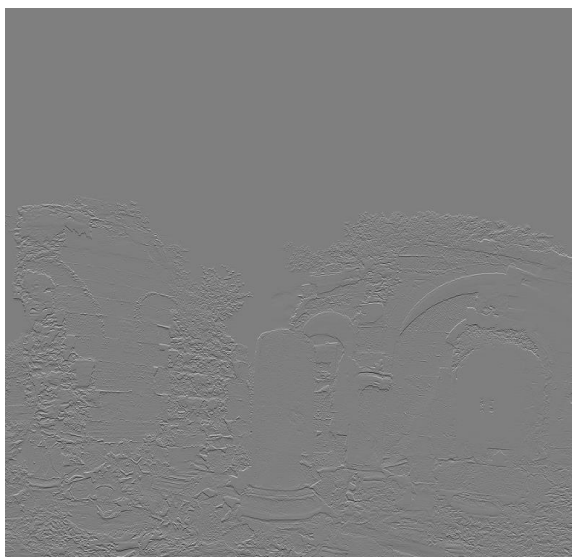


Рис.15. «Ruins» закодировано методом разностей

Размер полученного изображения 342 КВ, тем самым мы сэкономили 31% места, занимаемого данным изображением. Заметим, что кодирование и декодирование производится без значительных потерь, ощутимых человеческим глазом.

После декодирования изображение примет следующий вид:



Рис.16. «Ruins» после декодирования

При подсчёте значений PSNR получаем численные значения равные 51.16дБ, из чего заключаем, что искажения не заметны для человеческого глаза.

Для достижения наилучшего результата сжатия метод разностей следует применять не для всего изображения в целом, а для фрагментов изображения, полученных путём последовательного разбиения изображения. Описание подобного алгоритма фрагментации представлено ниже.

3.2 Метод прямоугольных сеток

Основным объектом, входящим в способ кодирования изображения, является последовательность ординат $c_1 < c_2 < \dots < c_k$ таких, что $c_1 = 0, \dots, c_k = N_n$ для каждого i удовлетворяющего условию $1 \leq i \leq k - 1$, задается система $D_i = (d_{i1}, d_{i2}, \dots, d_{il})$, удовлетворяющая условиям $d_{i1} \leq d_{i2} < \dots < d_{il}$ $d_{i1} = 1, d_{il} = M$ (разумеется, при этом l зависит от i). В результате изображение разбивается на прямоугольники

$$G_{ij} = \{(x, y) | (c_i \leq y < c_{i+1}) \& (d_{ij} \leq x < d_{i,j+1})\}, \quad (1)$$

Где $1 \leq i < k$, и при каждом i в указанных пределах выполнено неравенство $1 \leq j < l$ (здесь l зависит от i).

Выбор параметров указанного разбиения должен осуществляться специальным алгоритмом, о котором пойдёт речь ниже. Если параметры разбиения уже выбраны, то выработка способа сокращённого кодирования изображения осуществляется на основании следующего подхода. (1) выделяются прямоугольники G_{ij} , в которых значения элементов матрицы, задающей изображение, либо совпадают (в одном из вариантов метода), либо отличаются содержащимися числовыми значениями на некоторую константу δ (в другом варианте метода).

Выбор одного из указанных вариантов метода, а так же выбор константы δ (в случае второго варианта) осуществляется специальным алгоритмом, идея которого основана на обнаружении фрагментов G_{ij} матрицы, задающей изображение, таких, где совокупность значений G_{ij} либо постоянная (в случае первого варианта метода), либо достаточно мало отличается от постоянной.

Возвращаясь к алгоритму выбора указанных выше параметров c_i и d_{ij} , отметим, прежде всего, что применение этого алгоритма основано на результатах работы другого алгоритма, при помощи которого выделяются участки заданного изображения, в которых значения элементов матрицы задающей изображение, либо совпадают (в случае первого варианта метода), либо отличаются друг от друга на некоторую достаточно малую константу δ (в случае второго варианта метода); при этом выбор константы δ зависит от подробностей избранного способа кодирования изображений.

Описание указанных участков изображения может производиться различными способами (например, эти участки могут кодироваться в виде многоугольников с рациональными вершинами). Далее описывается один из таких методов.

Применим немного видоизменённый алгоритм подсчёта разностей, описанный выше (метод отклонений). Отличие будет заключаться в том, что при подсчёте элементов $c_i^j = (b_i^{j+1} - b_i^j)$ будем брать модуль разности т.е.

$$c_i^j = |b_i^{j+1} - b_i^j|$$

Назовём полученные элементы переходным весом по оси Y для заданного j. Для каждого j в матрице $\{c_i^j\}$ посчитаем сумму

$$v_j = \sum_{i=1}^M c_i^j$$

Обозначим, через v^{max} максимальный элемент среди v_j , а через v^{min} минимальный. Обозначим, через $w_j = 255 * w_j * (w^{max} - w^{min})$. w_j представляет собой численное выражение в интервале $[0, 255]$, которое определяет собой мощность переходов по оси Y для заданного j. С помощью данных коэффициентов и будем производить разбиение $c_1 < c_2 < \dots < c_k$. Там где w_j достаточно велико деление нужно производить достаточно часто, там же где мало или близко к 0 деления вовсе не необходимы. Для наглядности рассмотрим пример в таблице такого разбиения для уже знакомого нам изображения «cameraman»:

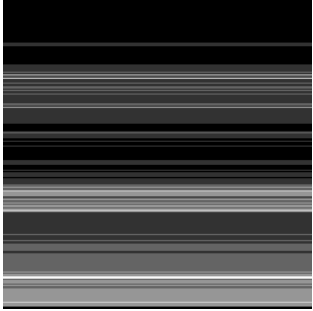
Серотонные показатели оттенков изображения	Весовые показатели изображения по оси Y
	

Таблица 5. «Cameraman» и его переходной вес по оси Y

В таблице в поле весовых показателей демонстрируются значения w_j для каждого j , чем светлее оттенок тем больше переходов на данном участке и тем чаще нужно производить дробление.

3.3 Фрагментное представление изображения

После проведения действий указанных в предыдущих двух параграфах, основной прямоугольник изображения разбивается на два прямоугольника, расположенных соответственно выше и ниже прямой $y = c_i$. Дальнейшие действия производятся аналогичным образом для частей, на которые разбивается прямоугольник.

Построение параметров c_i продолжается до того, как их совокупность окажется в соответствии с условиями завершения построения системы значений c_i ; такие условия естественно выработать заранее с учетом взаимного расположения областей совпадения или приблизительного совпадения значений основной матрицы. Естественным образом строится система горизонтальных полос, достаточно хорошо аппроксимирующая систему областей совпадения значений основной матрицы. Указанная система горизонтальных полос основная информация для построения системы параметров c_i .

Если система параметров $c_1 < c_2 < \dots < c_k$ уже построена, то для каждого i , удовлетворяющего условию $1 \leq i < k$ строится система $d_{i1} < d_{i2} < \dots < d_{il}$ (где l зависит от i). Такая система строится таким же методом, каким строилась система параметров c_i . В результате получается разбиение основного прямоугольника на прямоугольники G_{ij} , где $1 \leq i < k$, и при каждом i таком, что $1 \leq i < k$ имеет место $1 \leq j < l_i$.

Если прямоугольники G_{ij} уже построены, то способ сокращенного кодирования основной матрицы выглядит следующим образом. Если прямоугольник G_{ij} не содержится в областях совпадения или приблизительного совпадения значений основной матрицы, то значения элементов основной матрицы, попавших в G_{ij} кодируются так же, как это делалось в основной матрице. Если значения элементов основной матрицы, попавших в G_{ij} , совпадают, то кодируется лишь общее значение указанных элементов (с соответствующими дополнительными указателями). Если же значения элементов основной матрицы, попавших в G_{ij} совпадают лишь приблизительно, то в случае первого варианта рассматриваемого метода значения элементов матрицы, попавших в такой прямоугольник G_{ij} кодируются так же, как это делается в общем случае; во втором варианте рассматриваемого метода кодируется среднее арифметическое значений матрицы, попавших в G_{ij} , и кроме того, для каждого значения указанного вида кодируется его отклонение (в положительную или отрицательную сторону) от упомянутого среднего арифметического.

Может быть реализован метод сокращённого кодирования изображений, где роль рассматриваемых выше прямоугольников G_{ij} играют многоугольники с рациональными вершинами.

Смешанный метод работает следующим образом – имея изображение, представленное в виде разностей, определяем в них области с минимально изменяющимися значениями. Рассмотрим работу алгоритма с использованием следующего примера. На рисунке визуальны представлены цветовые разностные значения в некоторой области изображения.

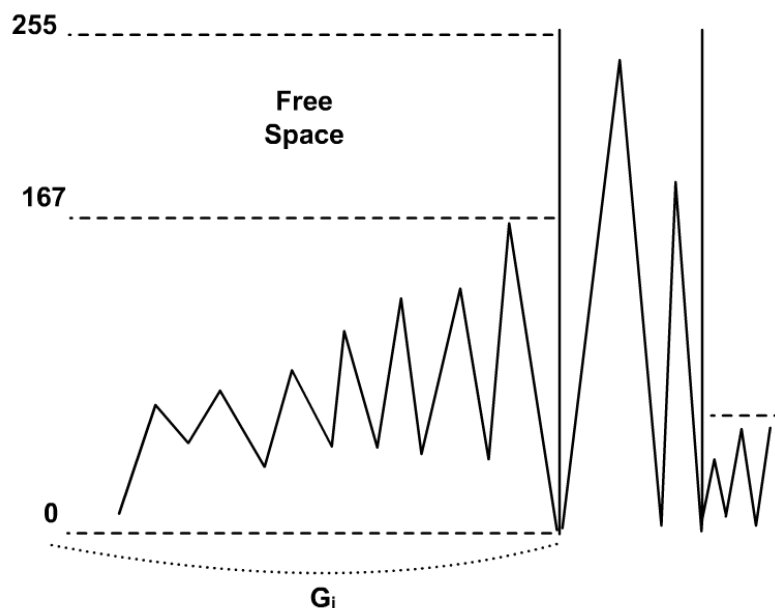


Рис.17. Демонстрация алгоритма фрагментации памяти

G_i в данном случае выделенный прямоугольник. Для него запоминается число 167. Производится расчёт возможной используемой свободной памяти. Для это производится расчёт $255 - 167 = 88$. Так как память в компьютере представляется битами в двоичном виде нам необходимо рассчитать сколько возможно использовать бит в данной свободной области. Для этого находим минимальную степень приближающую данное число. На примере это число 88. Минимальная степень двойки наилучшим образом приближающее число 88 это степень 6, так как $2^6 = 64$, степень 7 уже не подходит так как $2^7 = 128 > 88$. Обозначим количество пикселей участка G_i через некоторое число C . Тем самым на данном участке G_i получаем $C * 6$ бит свободной памяти для встраивания ЦВЗ. Сжатие пространственным алгоритмом происходит таким образом, чтоб заполнить свободное пространство всех выделенных областей.

Ещё раз подчеркнём, что описанный выше метод позволяет сократить объем кода не для всякого изображения. Однако опыт показывает, что для многих реальных цифровых фотографий и изображений сокращение объёма кода изображения достигается представленным методом.

Г Л А В А IV

ПРИЛОЖЕНИЯ, ПРОГРАММНАЯ РЕАЛИЗАЦИЯ

4.1 Общая характеристика программной системы

Разработан комплекс программ модулей для обработки изображений. В том числе разработана модульная программная система. Введём некоторые определения.

Определение. «Модульное программирование – это организация программы, как совокупности небольших независимых блоков, называемых модулями, структура поведения которых подчиняются определённым правилам» [124].

Использование модульного программирования позволяет использование разработанных модулей в различных иных системах, обеспечивает переносимость кода. Так же она позволяет упростить тестирование программного обеспечения и нахождение проблемных участков. В таком случае имеется возможность строгого разделения аппаратных подпрограмм от других подзадач, что положительно влияет на мобильность разрабатываемых подпрограмм.

Определение. «Модуль – функционально законченный фрагмент программы.»

Во многих языках оформляется в виде отдельного файла с исходным кодом. Некоторые языки предусматривают объединение модулей в пакеты.

Определение. «Пакет прикладных программ – набор взаимосвязанных модулей, предназначенных для решения задач определённого класса некоторой предметной области» [125].

К системе был применён каскадный подход программирования. Данный подход обосновывается следующим образом. Со временем накопления опыта при разработке программного обеспечения вырабатывается представление о сравнительно

рационалистичной модульной организации программного обеспечения. Когда программные модули сформированы, они не претерпевают значительных изменений при переходе от одной версии программы к другой, так же определяются модули, которые являются сменными и их местоположение относительно разрабатываемой программной системы. Таким образом используя каскадный подход может быть выработан каркас приложения и её архитектура. Каркас должен содержать гнезда для размещения сменных модулей [127]. Следует заметить, что при таком подходе работоспособность системы обеспечивается согласованием спецификаций сменных модулей и гнезд.

Определение специальной конфигурации таким образом упрощается для пользователя. В каркасе гнезда отражают характеристики решаемой задачи, а заменяемые модули представляют собой значения характеристик [127].

Модульное программирование значительным образом облегчает задачи, связанные с проектированием ПС. Принцип учитывает, что при разбиении ПС на отдельные модули для каждого из них заранее задаётся реализуемая им функциональность и вместе с этим определяются возможные взаимодействия с другими модулями. Среди основных преимуществ разработки ПС с помощью модульной архитектуры является возможность замены или обновления модулей, без переработки иных участков ПС.

В качестве модулей могут быть представлены такие программные элементы такие как библиотеки, структуры данных, сервисы, классы или другие подобные программные элементы, обладающие некоторой функциональностью и имеющие интерфейсы их задания.

Модуль отличается от библиотеки тем, что там не ставится цель полностью покрыть нужды предметной области, и приложение, вообще говоря, может использовать модули нескольких библиотек.

Как правило в каскадном подходе код программы разбивается на множество файлов, которые могут быть скомпилированы сепаративно. Подобный подход так же позволяет сэкономить время разработки программного кода и перекомпиляции при внесении изменений, в небольшое количество файлов исходников ПС. Также при данном подходе легче выполняется замена некоторых компонентов системы (например,

библиотеки dll для windows или jar файлы для среды Java в Android) скомпилированного программного продукта, без перекомпиляции всех элементов проекта программной системы. Благодаря данному подходу и обеспечена расширяемость разработанной в диссертации программной системы.

Так же следует отметить что среди методов разработки модульных ПС принципиальную роль играет ООП (объектное ориентированное программирование) обеспечивающее портативность тех или иных программных структур, что позволяет их внедрение в модульную структуру системы.

В разработанной программной системе так же присутствует панель управления системой, основанной на технологии Adobe AIR. Использование данной технологии в дальнейшем упрощает процессы дальнейшей портативности разработанной программной системы.



Рис.18. Демонстрация панели управления программной системы с использованием технологии Adobe AIR.

Первым шагом необходимо убедиться в работоспособности модулей приложения. Следующий рисунок схематично демонстрирует работу системы. На рисунке изображена топология модульной структуры системы.

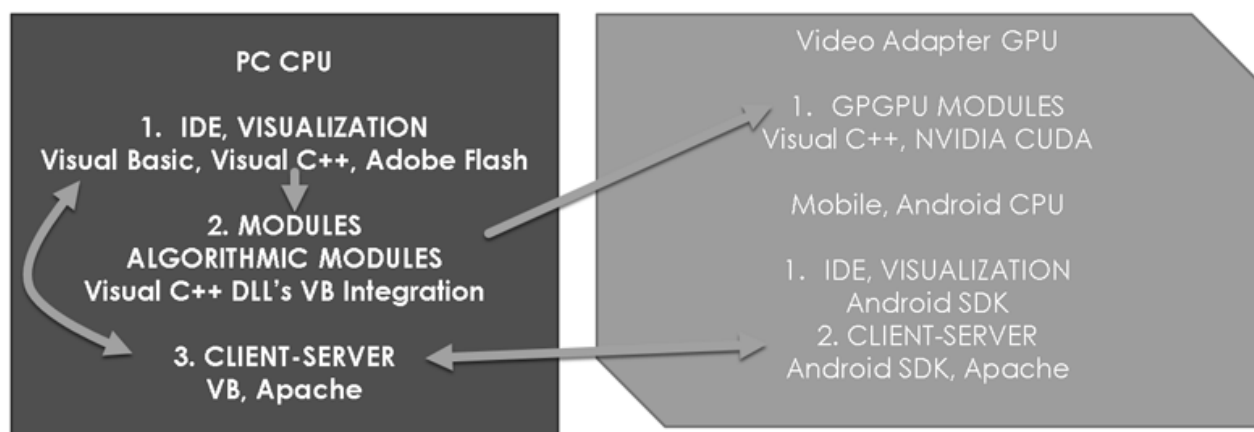


Рис.19. Основная топология программной системы

Интерфейс программы реализован на ИСР (интегрированной среде разработки) Microsoft Visual Basic.

Из интерфейса подгружаются библиотеки, реализованные на интегрированной среде разработки Microsoft Visual C++. Производительность модулей, реализованных на языке C++ значительно выше чем на Visual Basic поэтому и было принято решение реализации вычислений на языке C++. Модули могут быть написаны на любом языке программирования высшего уровня и использованы из интерфейса приложения на Visual Basic. Модули представляются в виде библиотек DLL для операционной системы Windows. Могут быть внесены небольшие модификации для портирования данных библиотек под операционную систему Linux или Mac OS. Что нельзя сказать об интерфейсе программного обеспечения. Интерфейс строго ориентирован для операционной системы Windows. Для обхода данной проблемы был разработан сервис для работы с модулями через сетевые возможности вычислительных машин. На момент написания данной работы сервис работает на операционной системе Windows и использует возможности визуальной подсистемы, разработанной на ИСР Visual Basic. Далее приведён рисунок такого взаимодействия с программным обеспечением.

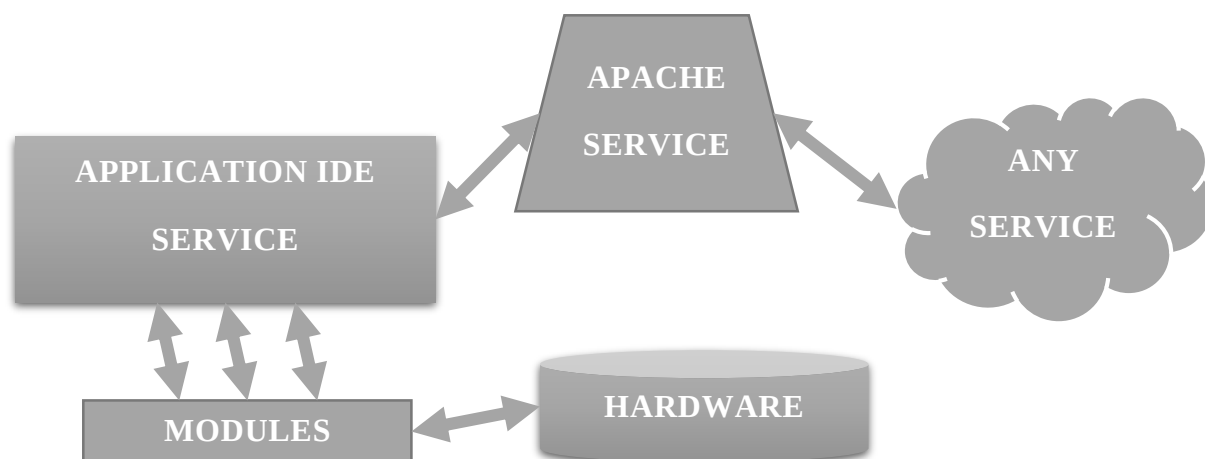


Рис.20. Взаимодействие внешних программ с алгоритмическими модулями приложения

Также была реализована программа для платформы Андроид, которая даёт возможность пользователю мобильного устройства воспользоваться некоторым спектром возможностей разработанных алгоритмов методов самовнедрения изображения. Допустим пользователь имеет изображение, которое должно быть защищено методом самовнедрения. Процесс обработки изображения происходит следующим образом: пользователь выделяет через интерфейс данное изображение или производит фотосъёмку (на усмотрение пользователя). Программа посылает данное изображение по сети на сервер, обрабатывающий данное изображение. После чего изображение, обработанное таким образом возвращается на мобильное устройство.

Схема аналогична схеме работы с иными обрабатывающими сервисами.

Идеология разработанной модели подразумевает дальнейшее использование специального формата для защищённых изображений IPP (image processor protected) и дальнейшее распространение изображений в данном формате. Будет подразумеваться, что отображение данного формата будет осуществляется только с помощью специального приложения «отобразителя».

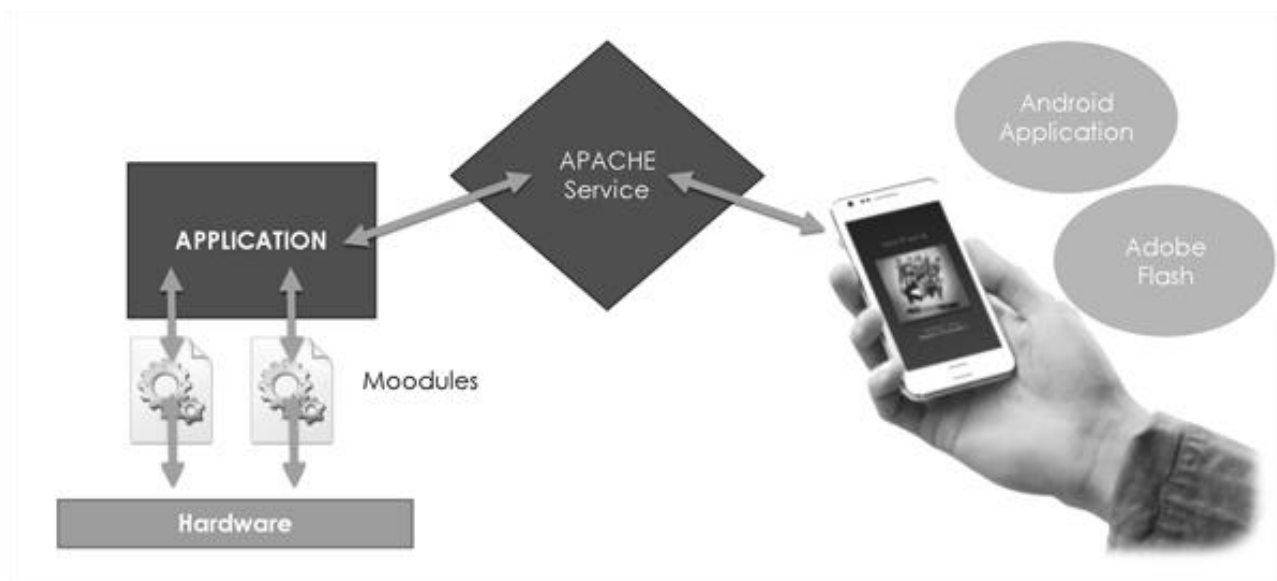


Рис.21. Схема интеракции мобильных устройств с модулями обработки

4.2 Методы обработки изображений в контексте разработанной программной системы

Разработанная программная система в основном нацелена на реализацию модели самовнедрения изображения, основанной на пространственных частотных алгоритмах [10], [70], [82]–[85] самовнедрения, описанных во второй и третьей главе.

Во многом эффективность системы зависит от правильно выбранных параметров при осуществлении самовнедрения. Для этого необходимо иметь достаточную экспериментальную базу знаний. В системе предоставлены возможности для проведения подобных экспериментов.

На каждом этапе проведения экспериментов предусматривается визуализация полученных результатов, что позволяет устанавливать качество (степень искажения) методами и алгоритмами оценки качества, представленными во второй главе, в том числе методом попиксельного сравнения или методами, учитывающими специфику человеческой визуальной периферии. Механизм сравнения, основанный на человеческом

восприятию визуальной информации, представляет крайнюю важность для разработанных методов, так как в данных системах самовнедрения обнаружение искажений и восприятие встроенного оригинала остаётся за человеком.

Интерфейсом, разработанным на ИСР Visual Basic, предусмотрено две возможности подачи на вход изображений для дальнейшей обработки: консольные и визуальные.

Далее изображение конвертируются в матрицу интенсивности пикселей. В интерфейсе сразу производится деление на два потока изображений, хотя в начале они идентичны. В одно изображение производится самовнедрение, второе изображение – ЦВЗ, модифицируется с учётом алгоритмов, представленных в второй и третьей главе.

В системе предполагается, что пользователем регулируются также параметры работы алгоритмов. Например, может быть задан идентификатор псевдослучайного правила смешивания пикселей. Предусмотрена возможность задания значения коэффициента α , определяющего мощность встраивания ЦВЗ методом самовнедрения. Также могут быть заданы параметры атаки на изображения, в случае их использования средствами программной системы. Для каждого алгоритма существует несколько специфических параметров. Перечислим только самые необходимые.

Для работы метода разложения изображения в спектральное представление, необходимо задать параметры сжатия изображения. Параметры также отображаются в визуальном интерфейсе приложения. Для этого предусмотрено специальное окно настроек. Названия параметров можно задать с помощью специального настроенного файла (в случае такой необходимости) (данная возможность предусмотрена, в основном, для разработчиков программного обеспечения).

После введения параметров и осуществления алгоритмов система запоминает их, тем самым обеспечивая дальнейшую возможность отладки и улучшения программного обеспечения. После применения алгоритма пользователь может оценить состояние самовнедренного изображения и в случае неудовлетворительного результата повторить встраивание с иными параметрами алгоритма. Удовлетворительные параметры так же

запоминаются в системе. Система продолжает работу в соответствии с последним удовлетворительным заданием параметров.

Для декодирования изображения самовнедрённое с помощью спектрального комбинированного алгоритма необходимо учитывать строгое соответствие параметрам, заданным при кодировании данного изображения. Так же для декодирования необходимо обладать следующей информацией: численное значение определяющее псевдослучайный алгоритм смешивания пикселей, численное значение коэффициента отвечающего за мощность процесса самовнедрения, матрица средних значений блоков изображения-контейнера. В результате получаем образ исходного изображения.

В случаях возникновения вопроса авторского права на изображение для упрощения процессов декодирования может быть задано изображение-источник. По извлечённой информации можно сделать вывод, является ли изображение искажённым источником или нет, на основе визуального восприятия или с помощью оценочных алгоритмов, применённых для исходного и выделенного изображения. В случаях, когда система применяется для скрытого переноса оригинала, сравнение с изображением-источником не представляется возможным. В таких случаях эффективность работы системы оценивается по качеству извлечённой информации. Для этого существуют универсальные алгоритмы, которые могут быть в дальнейшем добавлены в качестве модулей.

Так же следует отметить, что оценочные алгоритмы работают в системе в независимости от поставленной задачи. Тем самым пользователь может проверить сходство двух независимых изображений или обрабатывать изображения на основе добавленных модулей к системе.

В системе также предусмотрена возможность извлечения самовнедрённого изображения в случае, если изображение было повёрнуто или обрезано. В случае обрезки возможно восстановить оригинал, но в случае применения алгоритма смешивания пикселей это негативно сказывается на качестве изображения. Для этого случая подобного искажения изображения разработчик предусмотрительно перед встраиванием должен отключить модуль смешивания пикселей. Также в системе предусмотрен

алгоритм эвристического заполнения недостающих значений. Пользователь может производить алгоритм несколько раз для получения наиболее реалистичного результата.

В визуальном интерфейсе системы так же предусмотрены следующие возможности обработки изображений:

1. Сжатие изображения спектральными методами, как прямым методом, так и обратным
2. Построение графиков гладкости изображения
3. Смешивание пикселей изображения с помощью преобразования Арнольда (об смешивании методом псевдослучайного правила было сказано ранее в данной главе)
4. Присутствует возможность наложения Гауссовского шума на изображения (тем самым представляя собой генератор атаки на изображение)
5. Возможность размытия изображения.
6. Для разработчиков даётся возможность использования специальных микропрограмм воздействия на изображения. Данные микропрограммы называются «Генераторами». Приведено несколько примеров таких микропрограмм. Предполагается написание их по аналогии. Для реализации метода внедрения микропрограмм был использован компонент операционной системы Windows – MSScriptControl.

Рассмотрим главное окно приложения Image Processor Engine Manager. На рисунке, представленном ниже, изображено окно программы.

В верхней части программы содержится системное меню приложения, пункты которого представляют собой выпадающее подменю.

Описание пунктов подменю.

File. Подменю содержит пункты для работы с файлами изображений. В подменю содержатся такие пункты, как открытие, сохранение графических файлов. Так же присутствует возможность сохранения изображения в текстовом формате.

Edit. Подменю содержит в себе пункты для внешней работы с графическими файлами. Для этого в системе предусмотрено два основных носителя визуальной информации с названиями – «Main» и «Second» соответственно. «Main» в основном используется для основной обработки. Все модули обрабатывают изображение под меткой «Main». Вторичное же изображение «Second», как правило, используется в основном в двух следующих целях: 1. для хранения оригинала изображения, 2. для временного буфера на время, пока обрабатывается изображение под меткой «Main». Так же следует заметить, что алгоритмы оценки качества так же работают с обоими изображениями. «Main» используется в качестве оригинала, а «Second» в качестве искажённого изображения. В представленном подменю также имеются функции ротационной замены данных изображений.

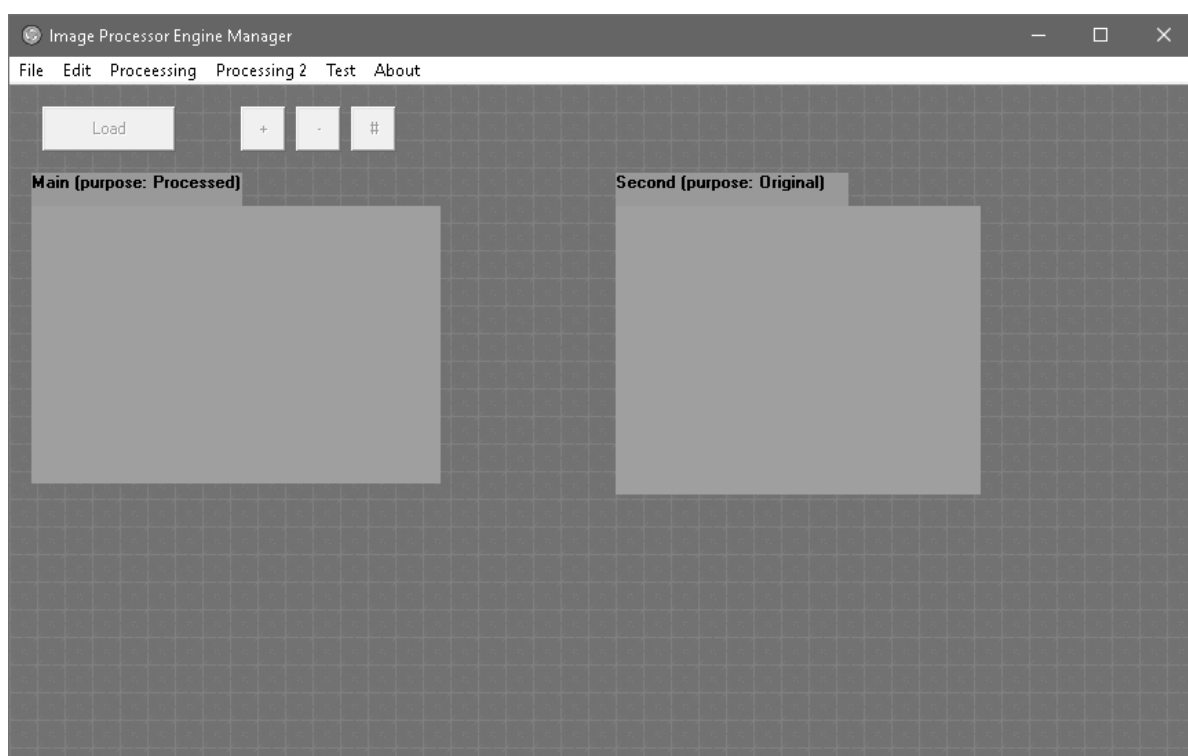


Рис. 22. Основное окно управления программной системой Image Processing Engine Manager

Processing. Подменю содержит алгоритмы, преимущественно представленные во второй главе. Следует заметить, что данное подменю использует вычислительные

возможности модулей, установка которых требуется ранее из панели управления системы. В данном подменю также содержатся функции обработки изображения, такие как конвертация изображения в серотонные представление. Также в данном подменю содержатся алгоритмы сжатия изображения DCT и FFT.

Смешивание пикселей так же производится из данного подменю. Для задания псевдослучайного правила нужно ввести характеризующее его числовое значение. Число следует задавать в шестнадцатеричном его представлении.

Также доступны и иные алгоритмы смешивания пикселей, в том числе и преобразование Арнольда. Его применение задаётся методом шага вперёд, но также реализована возможность обратного шага. При применении в достаточном количестве шагов в одну из сторон мы вернём исходное представление изображения.

Processing2. Подменю содержит алгоритмы, преимущественно представленные в третьей главе. Следует заметить, что данное подменю использует вычислительные возможности модулей, установка которых требуется ранее из панели управления системы. В данном подменю также содержатся функции обработки изображения, такие как демонстрация наибольших скачков в изображении. Так же в подменю содержатся функции конвертации изображения. Далее следует рисунок, демонстрирующий работу данного алгоритма. В области «Second» представлен оригинал изображения, а в области «Main» уже обработанный вариант данного изображения (применена инверсия). Также в данном подменю представляется возможным визуальное отображение алгоритма фрагментации изображения.

Test. Подменю содержит алгоритмы сравнения изображений. Сравнение происходит между двумя источниками визуальных данных «Main» и «Second». Описания алгоритмов сравнения и оценки качества изображения относительно оригинала приведены во второй главе.

About. Содержит информацию о программном обеспечении.

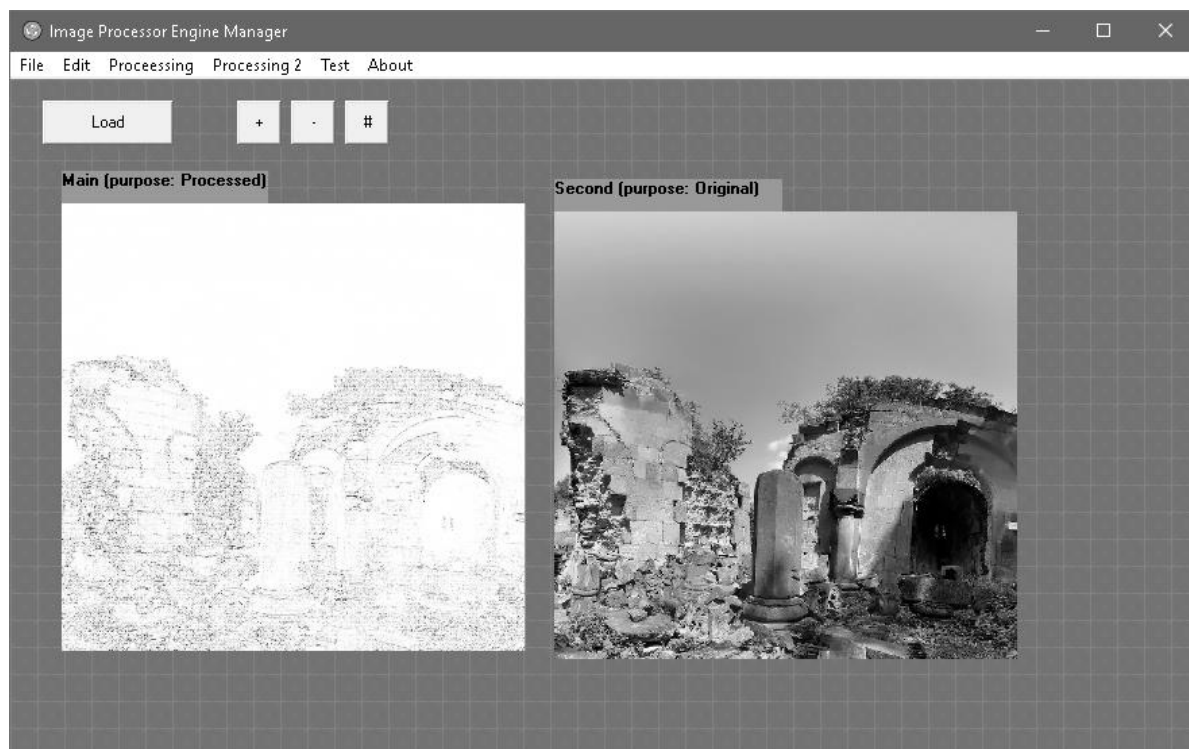


Рис.23. Демонстрация работы весового алгоритма, подменю Processing2

4.3 Применение алгоритмов самовнедрения изображения для обеспечения безопасности электронных подписей

Как было оговорено в первой главе, в последнем десятилетии возникла острая необходимость в использовании вычислительных технологий в самых разнообразных отраслях человеческой деятельности. Отметим, что электронные средства связи обеспечивают оперативное распространение информации. В связи с большим объёмом функциональных возможностей эти средства используют в различных областях человеческой деятельности, и с течением времени область их применения только увеличивается. В частности, большое распространение получило использование электронных документов в тех или иных правовых отношениях. Сейчас уже значительная часть передачи информации и документов осуществляется благодаря электронным устройствам, и информационно-коммуникационные технологии (далее – ИКТ) используются в целях совершения гражданско-правовых сделок.

Электронная подпись и электронный документооборот способствуют актуализации и росту экономических процессов, распространению ИКТ на все сферы общественной жизни и развивают возможности электронного бизнеса. Особенно это актуально в странах, где на законодательном уровне закреплено понятие электронной подписи и её применение. С точки зрения использования электронной подписи в современных электронных документах уже становится возможным с использованием электронных устройств безопасно и гарантированно совершать практически любые сделки, отстаивать свои права в органах государственной системы [126].

Необходимость использования электронных подписей возникла ещё в 1970х годах, хотя в то время не было для того достаточных предпосылок или условий, позволяющих реализацию подобных нужд, например, не существовало структур, благодаря которым возможно было полноценное установление личности для установления авторского права на электронный документ. На сегодняшний день ЭЦП (электронная цифровая подпись) – это (зашифрованное) крипто-программное средство, благодаря которому возможно установление факта подтверждения того, что электронная подпись поставлена именно её автором, а не иным лицом.

Таким образом, внедрение системы электронного оборота документов является не только экономически выгодным шагом, но и необходимым шагом в связи с точностью работы данных систем. Электронный оборот документов позволяет использовать эффективнее накопленные знания, увеличить рентабельность и качество процессов в экономике, а также увеличить управляемость происходящих процессов. В этом аспекте вопросы защиты и сохранения подлинной документации становится все более насущной проблемой. Представленные в работе алгоритмы позволяют внедрение электронных подписей в документы, представленные в виде изображений. Метод самовнедрения позволяет установить было ли изображение искажено по сравнению с оригиналом.

Произведём алгоритм самовнедрения для сформированного документа бухгалтерского учёта счёт-фактуры, на котором присутствуют все необходимые реквизиты и поставлена печать.

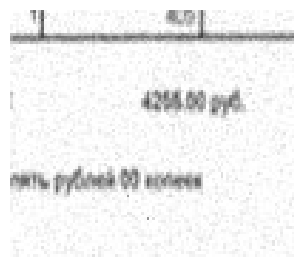


Рис.27. Декодированное изображение к которому применено обратное спектральное преобразование

На декодированном изображении можно отчётливо увидеть число (общую сумму) записанную в качестве реквизита в данной счёт-фактуре. Можно заметить, что в изображении общая сумма была заменена с 4255 руб. на 44255 руб.

Благодаря работе системы удалось установить истинное состояние документа.

4.4. Средства для работы с модулями параллельных вычислений

Разработаны так же аналогичные модули использующие средства GPGPU NVIDIA CUDA API для ускорения вычислений.

Как было указано в первой главе при произведении расчётов на графических процессорах типа GPGPU наиболее уязвимым местом является шлюз передачи информации между шиной памяти графического адаптера и шиной передачи информации между центральным процессором и оперативной памятью. Иными словами, считывание информации из памяти графического процессора представляет собой существенную помеху производительности и при нерациональном использовании ресурсов может привести к бессмысленности использования ресурсов графического процессора.

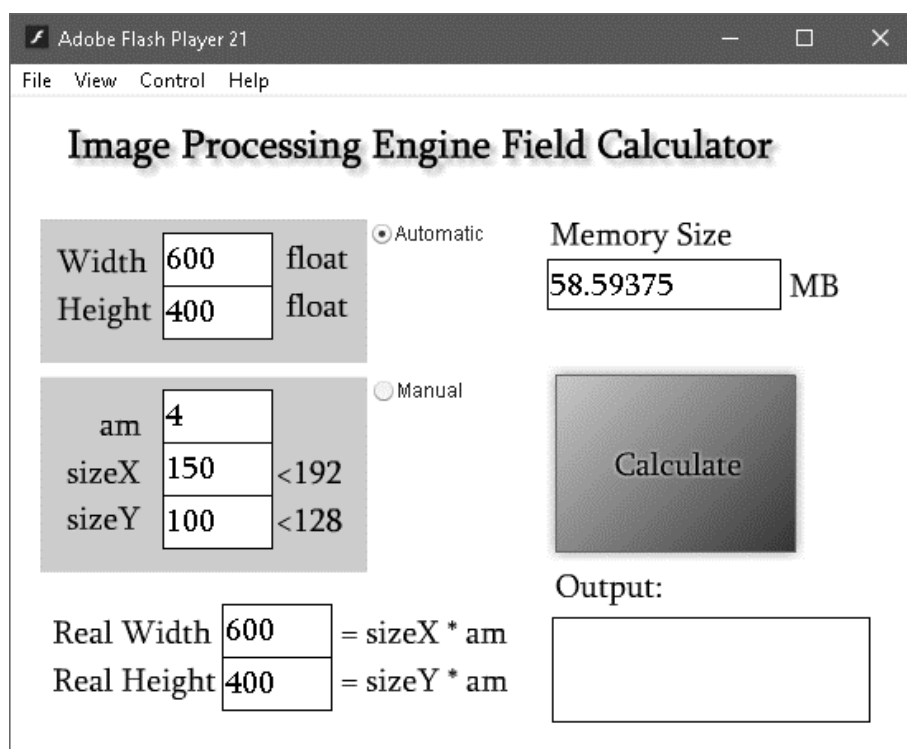


Рис.28. Интерфейс утилиты задания параметров для параллельных вычислений

Более того, может даже уменьшится общая производительность системы. Для избегания подобных осложнений была разработана небольшая утилита, написанная с использованием технологии Adobe Flash для задания наиболее оптимальных параметров алгоритма передачи и распараллеливания вычислений.

ОСНОВНЫЕ РЕЗУЛЬТАТЫ ДИССЕРТАЦИОННОЙ РАБОТЫ

1. Разработаны методы и алгоритмы защиты и восстановления изображений путём их самовнедрения, обеспечивающие сохранность данных и устойчивость к различным искажениям [1].
2. Разработан метод для кодирования цифровых изображений, основанный на обработке растровых данных изображения, обеспечивающий компактное представление большинства графических изображений и дающий возможность самовнедрения изображения [3].
3. Реализована модель обеспечения безопасности цифровых изображений, путём совмещения алгоритма сжатого кодирования некоторого класса изображений и разработанного модифицированного комбинированного (пространственно-частотного) алгоритма самовнедрения изображения [2, 4].
4. Разработана программная система, реализующая предложенную модель безопасности цифровых изображений, в которой использованы описанные алгоритмы [1-5].
5. Разработано программное обеспечение, позволяющее реализовывать описанные алгоритмы на графических процессорах, а также представлена реализация клиент-серверной технологии для мобильных устройств [1-5].

СПИСОК ПУБЛИКАЦИЙ

1. D. Asatryan, A. Petrosyan, N. Asatryan, N. Lanina. Method for Detection of Image Tampering and Partial Recovery. // Proc. of 9th Int. Conf. on Computer Science and Information Technologies - CSIT'2013, pp. 177-180, Yerevan, September 2013.
2. David Asatryan, Naira Asatryan, Natalya Lanina, Alexander Petrosyan // IEEE Conference Publications Method of detection of image tampering and partial recovery ISBN: 978-1-4799-2460-8 pp.1-6

3. А. Петросян. Метод кодирования дискретных изображений для сжатия и последующего встраивания ЦВЗ // Сборник статей ЦНС “Международные научные исследования” по материалам VII международной научно-практической конференции: «Проблемы и перспективы современной науки», N, Москва, 2016. ISSN 4684-1407 pp. 61-66
4. А. Петросян. Некоторые Методы Фрагментного Кодирования Дискретных Изображений. //Институт Проблем Информатики и Автоматизации НАН РА pp.1-15
5. А. Петросян. Кодирование Дискретных Изображений Методом Относительных Отклонений // «Евразийское Научное Содружество», N, Москва, 2016. ПИ № ФС77 – 63258

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ:

- [1] Афонский А. А., Дьяконов В. П. Цифровые Анализаторы Спектра, Сигналов И Логики / Под Ред. Проф. В. П. Дьяконова. — М.: Солон-Пресс, 2009. — С. 248. — Isbn 978-5-913-59049-7.
- [2] Сергиенко А. Б. Цифровая обработка сигналов. — 2-е изд. — СПб.: Питер, 2006. — С. 751. — ISBN 5-469-00816-9.
- [3] Condon, E. U. (1937), ‘Immersion of the Fourier transform in a continuous group of functional transformations’, Proc. Natl. Acad. Sci. USA 23: 158–164.
- [4] A Brief History of Computing, by Gerard O’Regan, 2008, p. 87,92.
- [5] Fung, et al., ‘Mediated Reality Using Computer Graphics Hardware for Computer Vision’, Proceedings of the International Symposium on Wearable Computing 2002 (ISWC2002), Seattle, Washington, USA, 7–10 October 2002, pp. 83–89.
- [6] Giorgos Vasiliadis; Spiros Antonatos; Michalis Polychronakis; Evangelos P. Markatos; Sotiris Ioannidis (September 2008). ‘Gnort: High Performance Network Intrusion Detection Using Graphics Processors’. Proceedings of the 11th International Symposium on Re.
- [7] Arie Kaufman (1993). Rendering, Visualization and Rasterization Hardware. Springer Science & Business Media. pp. 86–87. ISBN 978-3-540-56787-5.
- [8] Цымбал В. П. Теория информации и кодирование. — Киев: Выща Школа, 1977. — 288 с.
- [9] Моргунова Е. А. Авторское право: / Е. А. Моргунова; отв. ред. В. П. Мозолин. — М.: Норма, 2008. — 288 с. — ISBN 978-5-468-00205-6.
- [10] Бабкин С.А. Интеллектуальная собственность в Интернет. — М.: АО ‘Центр ЮрИнфоР’, 2006. — 512 с. — ISBN 5-89158-070-5.

- [11] Липчик Д. Авторское право и смежные права / Droit d'auteur et droit voisins. — Пер. с фр.; Предисл. М. А. Федотова. — М.: Ладомир; Изд-во ЮНЕСКО, 2002. — 788 с. — 2000 экз. — ISBN 5-86218-395-7 UNESCO 92-3-202837-9.
- [12] Hartung F., Kutter M. Multimedia Watermarking Techniques. Proceedings of the IEEE. Vol. 87, №. 7. 1999.
- [13] A.Z.Tirkel, G.A. Rankin, R.M. Van Schyndel, W.J.Ho, N.R.A.Mee, C.F.Osborne. 'Electronic Water Mark'. DICTA 93, Macquarie University. p.666-673, 1993.
- [14] Osborne C., van Schyndel R., Tirkel A. A Digital Watermark // IEEE In-tern. Conf. on Image Processing, 1994. P. 86-90.
- [15] Hembrooke E. F. Identification of sound and like signals. United States Patent, 3,004,104, 1961.
- [16] Cox J., Miller M.L. The First 50 Years of Electronic Watermarking. EURASIP Journal of Applied Signal Processing. Vol. 2, pp. 126-132, 2002.
- [17] G. Coatrieux, H. Main, B. Sankur, Y. Rolland, and R. Collorec. Relevance of watermarking in medical imaging. In IEEE-embs Information Technology Applications in Biomedicine, Arlington, USA, pp. 250–255, Nov. 2000.
- [18] Osborne D., Abbott D., Sorell M., Rogers D. Multiple embedding using robust watermarks for wireless medical images. In IEEE Symposium on Electronics and Telecommunications. Timisoara, Romania. Section 13(34), pp. 120-125, 2004.
- [19] Jana M. Zane, Abdul R.M. Faze, Asian A. Aziz. Clinical evaluation of watermarked medical images. In Proceedings of the 28th Annual International Conference of the IEEE EMBS, New York City, pp. 5459-5462, 2006.
- [20] Fotopoulos V., Stavropol M.L, Skouras A.N. Medical image authentication and self-correction through an adaptive reversible watermarking technique. In Proceedings of the 8th IEEE International Conference on Bioinformatics and Bioengineering, pp.1-5, 2008.

- [21] Samoa M., Matsuura Y., Takashima Y. A scheme of digital watermarking for geographical map data. In Proc. Cryptography and Information Security, Okinawa, Japan, pp. 26-28, 2000.
- [22] Schulz G., Voigt M. A high capacity watermarking system for digital maps. In Proc. Multimedia and Security Workshop on Multimedia and Security. Magdeburg, Germany, pp. 180-186, 2004.
- [23] Jiao P.H., Chen Y. Z., Ma J. S., et al. Digital watermark-based security technology for geo-spatial graphics data. Chinese Geographical Science. Vol. 16, №. 3, pp. 276-281, 2006.
- [24] Nadir M.A., Ahmad I. Digital Text Watermarking: Secure Content Delivery and Data Hiding in Digital Documents. IEEE 39th Annual International Carnahan Conference on Security Technology (CCST), pp. 101-104, 2005.
- [25] Baron B., Gomez S., Beggaring V. Steganographic Watermarking for Documents. 34th Annual Hawaii International Conference on System Sciences (HICSS), Vol. 9, Outrigger Wailea Resort, Island of Maui, Hawaii, 2001.
- [26] Sarukhanyan H., Petrosyan G., Agaian S., Astola J. Robustness Testing of Some Watermarking Methods Using IDCT-II. Proc. of The 2005 International TICSP Workshop on Spectral Methods and Multirate Signal Processing. Riga, Latvia, pp. 89-94, 2005.
- [27] Асатрян Д.Г., Ланина Н.С. Адаптивный алгоритм встраивания цифровых водяных знаков в изображение. Годичная научная конференция РАУ. Сборник научных статей. Ереван, сс. 59-65, 2006.
- [28] Asatryan D., Lanina N., Shahverdyan H. Adaptive Robust Algorithm for Digital Watermarking of Medical Images. Proceedings of 6-th International Conference «Computer Science and Information Technologies – CSIT 2007», Armenia, Yerevan, 24-28 September, pp.16.
- [29] Asatryan D.G., Lanina N.S. Adaptive Robust Watermarking Algorithm for Image Protection. Vestnik RAU (Herald of the RAU), Armenia, Yerevan, pp. 50-56, 2009.

- [30] Margarov G.I. Covered remote user authentication in steganographic collaboration framework. Proceedings of 7-th International Conference «Computer Science and Information Technologies – CSIT 2009», Armenia, Yerevan, pp. 169-172, 2009.
- [31] Margarov G.I., Markarov V.G., Soghomonyan S. Distributed steganographic data storage with variable dynamic reconfiguration. Proceedings of 7-th International Conference «Computer Science and Information Technologies – CSIT 2009», Armenia, Yerevan, pp. 173-
.
- [32] Margarov V.I., Tolba M., Biometrics Based Secret Sharing Using Fuzzy Vault. Proceedings of 7-th International Conference «Computer Science and Information Technologies – CSIT 2009», Armenia, Yerevan, pp. 177-180, 2009.
- [33] Petrosyan G., Sarukhanyan H., Agaian S., Astola J. Robustness Testing of Some Watermarking Methods Using IDCT-II. Proc. of The 2005 International TICSP Workshop on Spectral Methods and Multirate Signal Processing. Riga, Latvia, pp. 89-94, 2005.
- [34] Agaian S., Caglayan O., Perez J.P., Sarukhanyan H., Astola J.. Golden Ratio-Haar Wavelet Based Steganography. International TICSP Workshop on Spectral Methods and MultiMate Signal Processing. Italy, pp. 89-94, 2006.
- [35] Грибунин В. Г., Оков И. Н., Туринцев И. В. Цифровая стеганография.
- [36] Li C. T.. Digital fragile watermarking schime for authentication of JPEG image. IEE Proc. on Vision, Image and Signal Processing. Vol. 151, №. 6, pp. 460-466, 2004.
- [37] Gang W, Ni-Ni R. A fragile watermarking scheme for medical image. Conf Proc IEEE Eng Med Biol Soc. Vol 4, pp. 3406-3409, 2005.
- [38] Chang Yu-Jie, Wang Ran-Zan, Lin Ja-Chen. A Sharing-Based Fragile Watermarking Method for Authentication and Self-Recovery of Image Tampering. EURASIP Journal on Advances in Signal Processing. Vol. 2008. Article ID 846967, 17 pages. 2008.
- [39] Maeno K., Sun Q., Chang Shih-Fu, Suto M. New Semi-Fragile Image Authentication Watermarking Techniques, Using Random Bias and Non-Uniform Quantization. IEEE Transactions on Multimedia, Vol. 8, № 1, 2006.

- [40] Jiang X., Liu Q. Semi-fragile watermarking algorithm for image tampers localization and recovery., China, pp. 343-351, 2008. Voloshynovskiy S., Pereira S., Pun T. Attacks on Digital Watermarks: Classification, Estimation-Based Attacks and Benchmarks. IEEE.
- [41] Yang Y., Sun X., Yang H., Li C.-T. Removable visible image watermarking algorithm in the discrete cosine transform domain. J.Electron. Imaging. Vol. 17, № 3, pp. 033008-1–033008-11, 2008.
- [42] Hu Y.J., Kwong S., Huang J. An algorithm for removable visible watermarking. IEEE Trans. Circuits Syst. Video Technol. Vol. 16, № 1, pp. 129–133, 2006.
- [43] Kankanhalli M.S., Rajmohan, Ramakrishnan K.R. Adaptive visible watermarking of images. In Proc. IEEE Int. Conf. Multimedia Comput. Syst. Vol. 441. Florence, SC, pp. 568–573, 1999.
- [44] Anderson R., editor. // Proc. Int. Workshop on Information Hiding: Lecture Notes in Computer Science. Springer-Verlag, Cambridge. 1996.
- [45] Ali Al-Haj. Combined DWT-DCT Digital Image Watermarking. Journal of Computer Science 3 (9): pp. 740-746, 2007.
- [46] Frank Y.Shih, Scott Y.Y.Wu. Combinational Image watermarking in the Spatial and Frequency domain. Pattern Recognition, Vol. 36, pp. 969-975, 2003.
- [47] J. Fridrich; Center for Intelligent Syst., State Univ. of New York Image Processing, 1999. ICIP Images with self-correcting capabilities ISBN: 0-7803-5467-2 pp.792 - 796 vol.3.
- [48] Xinpeng Zhang; Shuozhong Wang; Zhenxing Qia; Reference Sharing Mechanism for Watermark Self-Embedding ISSN: 1057-7149, 2010, pp 485 - 495.
- [49] Chuan Qina, Chin-Chen Changb, Pei-Yu Chend, Self-embedding fragile watermarking with restoration capability based on adaptive bit allocation mechanism // Signal Processing, Volume 92, Issue 4, April 2012, Pages 1137–1150.

- [50] Jose Antonio Mendoza-Noriega, Mariko Nakano-Miyatake., 'Halftoning-based Self-embedding Watermarking for Image Authentication and Recovery', IEEE Transaction 2010.
- [51] Na Li, Xiaoshi Zheng, Yanling Zhao, Huimin Wu, Shifeng Li. Robust Algorithm of Digital Image Watermarking Based on Discrete Wavelet Transform. International Symposium on Electronic Commerce and Security, pp. 942-945 Cox Kilian Leighton Shamoon Secure spre.
- [52] Barni M., Bartolini F., Cappellini V., Lippi A., Piva A. A DWT-based technique for spatio-frequency masking of digital signatures. Proceedings of the 11th SPIE Annual Symposium, Electronic Imaging '99, Security and Watermarking of Multimedia Contents. Vol.
- [53] Corvi M., Nicchiotti G. Wavelet-based image watermarking for copyright protection. Scandinavian Conference on Image Analysis. 1997.
- [54] Dugad R., Ratakonda K., Ahuja N. A new wavelet-based scheme for watermarking images. Proceedings of the IEEE International Conference on Image Processing. 1998.
- [55] Jun Y., Qing L., Chao L. An adaptive semi-fragile watermarking algorithm based on wavelet low frequency coefficients. American Journal of Engineering and Technology Research. Vol. 11, N.9, 2011.
- [56] Lu C.-S., Liao H.-Y. M. Oblivious watermarking using generalized Gaussian. Proceedings of the 7th International Conference on Fuzzy Theory and Technology, pp. 260-263, 2000.
- [57] Lakshmi D., Ganesh R., Marni S., Prakash R., Arulmozhivarman P. SVM based effective watermarking scheme for embedding binary logo and audio signals in images. In Proc. of IEEE TENCON. Hyderabad, India, pp. 1-5, 2008.
- [58] Chae J.J., Mukherjee D., Manjunath B.S. A robust embedded data from wavelet coefficients. Proceedings of SPIE, Electronic Imaging, Storage and Retrieval for Image and Video Database. Vol. 3312, pp. 308-317, 1998.

- [59] Kundur D., Hatzinakos D. A robust digital image watermarking method using wavelet-based fusion. Proceedings of the IEEE International Conference on Image Processing. Vol. 1. pp. 544-547. 1997.
- [60] Sverdllov A., Dexter S., Eskicioglu A.M. Robust DCT-SVD domain image watermarking for copyright protection: Embedding data in all frequencies. Proceedings of the 13th European Signal Processing Conference, 2005.
- [61] Xiaohu Ma, Xiaofeng Shen. A novel blind grayscale watermark algorithm based on SVD. Proceedings of the International Conference on Audio, Language and Image Processing, pp. 1063–1068, 2008.
- [62] Rahmani H., Mortezaei R., Moghaddam M.E. A New Robust Watermarking Scheme to Increase Image Security. EURASIP Journal on Advances in Signal Processing. Vol. 2010, Article ID 428183, 30 pages.
- [63] Mohanty S.P., Saraju P., Ramakrishnan K.R., Kankanhalli M.S. A dual watermarking technique for images. Proceedings of the 7th ACM International Multimedia Conference. Vol. 2, pp. 49–51, 1999.
- [64] Anand D., Niranjana U.C. Watermarking Medical Images with Patient Information, Proc. IEEE/EMSB Conference, Hong Kong, China, pp. 703-706. 1998.
- [65] Kickoffs A. La cryptographie militaires, Journal des sciences militaires. Vol. IX, pp. 5-38, 1883, Ferrier 1883.
- [66] Fredric J. Image watermarking for tamper detection. In: Proceedings of the IEEE International Conference on Image Processing. Vol. 2, Chicago, IL, USA, pp. 404–408, 1998.
- [67] Wu A. Swami Nathan, K. J. Ray Liu. Image tampering identification using blind disconsolation. Proc. IEEE ICIP, 2006.
- [68] Watanabe J., Hasegawa M., Kato S. A study on a watermarking method for both copyright protection and tamper detection. Proc. ICIP, International Conference on Image Processing. Vol. 4, pp. 2155– 2158, 2004.

- [69] Kutter M.F, Petitcolas A.P. A fair benchmark for image watermarking systems. Proceedings of SPIE: Security and Watermarking of Multimedia Content. San Jose, CA. Vol. 3657, pp. 219-239, 1999.
- [70] Cox I., Kilian J., Leighton T., Shamoon T. Secure spread spectrum watermarking for multimedia // IEEE Transactions on Image Processing. Vol. 6. № 12, pp. 1673-1687, 1997.
- [71] Moskowitz I., Longdon G., Chang L. A New Paradigm Hidden in Steganography. Proc. 2000 Workshop on new security paradigms, Ballycotton, Country Cork, Ireland, ACM Press, New York, pp. 41–50, 2000.
- [72] Sharp T. An implementation of key-based digital signal steganography. Proc. 4th International Workshop on Information Hiding, Pittsburgh, USA, Springer LNCS. Vol. 2137, pp. 13–26, 2001.
- [73] Mao Jia-Fa, Zhang Ru, Niu Xin-Xin, Yang Yi-Xian, Zhou Lin-Na. Research of Spatial Domain Image Digital Watermarking Payload. EURASIP Journal on Information Security. Vol. 2011, Article ID 502748, 12 pages. doi:10.1155/2011/502748, 2011.
- [74] Zain J., Clarke M. Security In Telemedicine: Issues in Watermarking Medical Images. 3rd International Conference Sciences of Electronic. Technologies of Information and Telecommunications (SETIT 2005), Susa, Tunisia, 2005.
- [75] J. Samuel Manoharan, Kezi S.Vijila A. Sathesh D.Narain Ponraj. Data Hiding for Medical Images: Issues and Challenges. International Journal of Computational Intelligence and Information Security. Vol. 2, №. 5. pp. 17-26, 2011.
- [76] Woo C.-S., Du J., Pham B. «Multiple Watermark Method for Privacy Control and Tamper Detection in Medical Images». In Proceedings APRS Workshop on Digital Image Computing Pattern Recognition and Imaging for Medical Applications, pp. 43–48. 2005.
- [77] Trichili H., Boubel M., Derbel N., Kamoun L. A new medical image watermarking scheme for a better telediagnosis. Proceedings of the IEEE International Conference on Systems, Man and Cybernetics 2002, 6–9 Oct. 2002, Vol.1. pp. 556–559, 2002.
- [78] Ruanaidh O., Pun T. Rotation, Scale and translation invariant spread spectrum digital image watermarking, Signal Processing, Vol. 66, pp. 303- 317, 1998.

- [79] Zhang X.P., Wang S.Z. Statistical Fragile Watermarking Capable of Locating Individual Tampered Pixels, SPLetters(14), №. 10. pp. 727-730, 2007.
- [80] Devapriya M., Ramar K. Statistical Image Watermarking In DWT with Capacity Improvement. Global Journal of Computer Science and Technology. Vol. 10. Issue 2 (Ver 1.0), pp. 20-24. 2010.
- [81] Moller S., Pfithman A., Stirand I. Computer Based Steganography. How It Works And Why Therefore Any Restriction On Cryptography Are Nonsense, At Best. // information hiding: First International Workshop «InfoHiding'96», Springer as Lecture Notes in Comput.
- [82] Koch E., Zhao J. Towards Robust and Hidden Image Copyright Labeling // IEEE Workshop on Nonlinear Signal and Image Processing. pp. 123–132. 1995.
- [83] Hsu C.-T., Wu J.-L. Hidden digital watermarks in images // IEEE Transactions on Image Processing. Vol. 8. № 1, pp. 58-68. 1999.
- [84] Podilchuk C.I., Zeng W. Perceptual watermarking of still images. Electronic Proceedings of the IEEE Workshop on Multimedia Signal Processing. 1997.
- [85] J.Samuel Manoharan, Kezi C.Vijila, A.Sathesh. Performance Analysis of Spatial and Frequency Domain Multiple Data Embedding Techniques towards Geometric Attacks. International Journal of Security (IJS), Vol.4, Issue 3, pp. 27-37. 2010.
- [86] Ланина Н.С. Об устойчивости к атакам пространственного алгоритма встраивания в изображение цифровых водяных знаков. Тезисы докладов научно-практической конференции по вопросам надежности и безопасности информационных систем. Армянская технологическая акад.
- [87] Асатрян Д.Г., Ланина Н.С. Исследование ошибок в адаптивном методе цифровой защиты информации. Труды второй годичной научной конференции РАУ, Ереван, сс. 67-74. 2007.
- [88] Hyi Su, C.-C. Jay Kuo. Spatial-frequency composite watermarking for digital image copyright protection. San Jose, CA, pp. 295-305. 2000.

- [89] Ganesan K., Guptha T.K. Multiple Binary Images Watermarking in Spatial and Frequency Domains. *Signal & Image Processing. An International Journal (SIPIJ)*. Vol.1, №.2, pp. 148-159, 2010.
- [90] Piva A, Barni M., Bartolini F., Cappellini V. Threshold Selection for Correlation-Based Watermark Detection. *Proceedings of COST 254 Workshop on Intelligent Communications*, L'Aquila, Italy. 1998.
- [91] Piva A., Barni M., Bartolini F., Cappellini V. DCT-based watermark recovering without resorting to the uncorrupted original image. In *Proc. of International Conference on Image Processing*, Santa Barbara, California. 1997.
- [92] Barni M., Bartolini F., Cappellini V., Piva A. A DCT-domain system for robust image watermarking. *Signal Processing, Special Issue in Copyright Protection and Access Control for Multimedia Services*. Vol. (66) 3, pp. 357-372. 1998.
- [93] Veeraswamy S., Kumar S. Adaptive AC-coefficient prediction for image compression and blind watermarking. *Journal of Multimedia*. Vol. 3, №. 1, pp. 16–22, 2008.
- [94] Mohanty S.P., Bhargava B.K.. Invisible watermarking based on creation and robust insertion-extraction of image adaptive watermarks. *ACM Transactions on Multimedia Computing, Communications and Applications*. Vol. 5, № 2, article 12, 2008.
- [95] Igor Pavlov '7z Format' 7-zip.org/7z.html Retrieved 2013.
- [96] Biryukov, Alex and Khovratovich, Dmitry Related-key Cryptanalysis of the Full AES-192 and AES-256 // *Advances in Cryptology – ASIACRYPT 2009*. — Springer Berlin / Heidelberg, 2009. — Vol. 5912. — P. 1—18. — DOI:10.1007/978-3-642-10366-7_1.
- [97] Federal Information Processing Standards Publication 197. 'Announcing the ADVANCED ENCRYPTION STANDARD (AES)'. United States National Institute of Standards and Technology (NIST) November 26,2001.
- [98] 1321 RFC, section 3.4, 'Step 4. Process Message in 16-Word Blocks', page 5.
- [99] Intel Processors Specifications intel.com.

- [100] A. S. Gevorkyan, H. G. Abajyan, H. S. Sukiasyan, ‘A New Parallel Algorithm for Simulation of Spin-Glasses in External Fields’, Proceedings of 34th International Convention on Information and Communication Technology, Electronics and Microelectronics : Gri.
- [101] H. G. Abajyan, ‘Implementation and Benchmarking of Ideal Ensemble of Disordered 1D Spatial Spin-Chains on GPUs and Computer Clusters’, Proceedings of Eighth International Conference on Computer Science and Information Technologies, pp. 231-235, 2011.
- [102] Hubert Nguyen GPU Gems 3 Addison Wesley Professional Print ISBN-10: 0-321-51526-9 2007 p1008.
- [103] Nikolaidis N., Pitas I. Robust image watermarking in the spatial domain. Signal Processing, Special Issue on Copyright Protection and Control. Vol. 66. № 3, pp. 385-403. 1998.
- [104] Ming Tong, Jia Hu, Hongbing Ji. An adaptive digital image watermark algorithm based on gray scale morphology. Journal of Electronics. Vol. 26, , pp. 417-422. 2009.
- [105] Chang-Hsing Lee, Yeuan-Kuen Lee. An Adaptive Digital Image Watermarking Technique for Copyright Protection. IEEE Transactions on Consumer Electronics. Vol. 45, No 4, pp. 1005–1015.1999.
- [106] Keshari S., Modani S. G. Image Encryption Algorithm based on Chaotic Map Lattice and Arnold cat map for Secure Transmission. International Journal on Computer Science and Technology. Vol. 2, Issue 1, pp.132-135, 2011.
- [107] Liehuang Z., Wenzhuo L., Lejian L., Hong L. A Novel Image Scrambling Algorithm for Digital Watermarking Based on Chaotic Sequences. International Journal of Computer Science and Network Security. Vol. 6, No.8B, August 2006.
- [108] Wang Hui-qin, Hao Ji-chao, Cui Fu-ming. Colour Image Watermarking Algorithm Based on the Arnold Transform. International Conference on Communications and Mobile Computing (CMC). Location: Shenzhen. pp. 66, 2010.

- [109] Song J. H., Z. Yong. A Digital Watermarking Method by Double Encryption Based on Arnold and Chaos in DCT Domain. *Applied Mechanics and Materials*. Vol. 65, pp. 104-107, 2011.
- [110] Kurak C., McHugh J. A Cautionary Note on Image Downgrading. *Proc. IEEE 8th Annual Computer Security Applications Conference*. San Antonio, USA, pp. 153–155, 1992.
- [111] Aura T. Practical Invisibility in Digital Communication. In *Proceedings of Information Hiding*, pp. 265-278. 1996.
- [112] Zhenjun Tang and Xianquan Zhang. Secure Image Encryption without Size Limitation Using Arnold Transform and Random Strategies. *Journal of multimedia*, Vol. 6, № 2, pp. 202-206, 2011.
- [113] Peterson G.. Arnold's Cat Map , *Math 45. – Linear Algebra*, 1997.
- [114] D. H. Lehmer, Mathematical methods in large-scale computing units, *Proceedings of a Second Symposium on Large-Scale Digital Calculating Machinery*, 1949, Harvard University Press, Cambridge, Mass., 1951, pp. 141—146. MR 0044899.
- [115] Дональд Кнут. Том 2. Получисленные методы // Искусство программирования. Указ. соч. — С. 21—37 Вильямс 2001.
- [116] M. Greenberger, Method in randomness, *Comm. ACM* 8 (1965), 177—179.
- [117] T.E. Hull and A.R. Dobell «Random Number Generators», *SIAM Review* 4-3 , p.230-254 1962.
- [118] Бакнелл Д. М. Фундаментальные алгоритмы и структуры данных в Delphi. Библиотека программиста. 2002 год. журнал Delphi Informant Magazine. Глава 6.
- [119] Stephen K. Park and Keith W. Miller (1988). Random Number Generators: Good Ones Are Hard To Find. *Communications of the ACM* 31 (10): 1192—1201.
- [120] Кулаков И. А. Линейные конгруэнтные и рандомизационные генераторы. Москва. 2012.

[121] Asatryan D.G., Lanina N.S., Shahverdyan H.S. Adaptive Robust Algorithm for Digital Watermarking of Medical Images. Proc. of 6th Int. Conf. on Computer Science and Information Technologies - CSIT'2007, Yerevan, pp. 161-164, 2007.

[122] Xiangjun, L., Jianfei, C. Robust transmission of JPEG2000 encoded images over packet loss channels. ICME pp. 947-950. School of Computer Engineering, Nanyang Technological University 2007.

[123] Salomon, David (2007). Data Compression: The Complete Reference (4 ed.). Springer. pp. 281. ISBN 978-1846286025. Retrieved 26 July 2012.

[124] Федорченко В.Н. Карповец Г.Н. 'Технология разработки программных продуктов' Санкт-Петербург 2010г.

[125] Кочергин В. И. Введение // Англо-русский толковый научно-технический словарь по системному анализу, программированию, электронике и электроприводу. — Томск, 2008. — Т. 1. — С. 31. — 652 с. — ISBN 5751119371.

[126] Электронная подпись в сфере нотариата, приказ Министерства юстиции Украины и Администрации государственной службы специальной связи и защиты информации от 28.08.2012 г. №1236/5/453, зарегистрированный в Министерстве юстиции 20 августа 2012 №1398/21710

[127] Горбунов-Посадов М.М. Глава 3. Пакет программ // Расширяемые программы. — М.: Полиптих, 1999. — 336 с