|  |  |  |
| --- | --- | --- |
| **Правительство Российской Федерации**  **Федеральное государственное автономное образовательное**  **учреждение высшего профессионального образования**  **"Национальный исследовательский университет**  **"Высшая школа экономики"**  ***Отделение программной инженерии***  ***Кафедра Управления разработкой программного обеспечения*** | | |
|  |  | УТВЕРЖДАЮ  Зав. Кафедрой УРПО  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ С.М. Авдошин  «\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2014 г. |

***ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА***

***по направлению 231000.62 Программная инженерия***

***подготовки бакалавра***

На тему: Программа стеганографического сокрытия информации в видеофайлах формата MPEG

Студента группы № \_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_   
 Подпись (Ф.И.О.)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
(Дата)

Научный руководитель

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_   
(должность, звание) подпись (Ф.И.О.)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
(Дата)

Москва, 2014 г.

# Реферат

Отчет 38 с., 3 главы, 22 рис., 2 табл., 16 источников, 4 прил.

видео стеганография, стеганография mpeg, сокрытие информации в видео, встраивание и извлечение информации, дискретное косинусное преобразование, помехоустойчивое кодирование, циклические коды, двоичные циклические коды

Объектом разработки является программа стеганографического сокрытия информации в видеофайлах формата MPEG.

Цель работы – разработка программы, которая позволила бы встраивать информацию в видеофайл в формате MPEG и извлекать информацию из видеофайла в формате MPEG, скрыв сам факт встраивания.

В ходе работы проводились анализы существующих подходов к видео стеганографии и используемых ими алгоритмов: дискретного косинусного преобразования и помехоустойчивого кодирования. Также была изучена модель сжатия MPEG.

В результате работы была разработана программа, позволяющая встраивать текстовую информацию и изображения в видеофайлы формата MPEG и извлекать встроенную информацию из видеофайлов формата MPEG. В основе метода встраивания лежит метод дискретного косинусного преобразования. Для увеличения устойчивости использовалось помехоустойчивое кодирование двоичными циклическими кодами.

Программа может использоваться для передачи секретных данных на электронных носителях и по каналам связи.

В дальнейшем планируется увеличить скорость работы программы, изучить другие алгоритмы встраивания информации в видеофайлы и добавить их к текущему функционалу программы.

# Определения, обозначения и сокращения

MPEG – (англ. Motion Picture Experts Group) группа экспертов, которая разрабатывает, обновляет и поддерживает стандарты сжатия цифровой аудио и видео информации.

– (англ. Red Green Blue) цветовое пространство, состоящее из трёх цветовых компонент: красной, зелёной и синей.

– цветовое пространство, состоящее из трёх компонент: компоненты яркости и двух компонент цветности и .

ДКП – дискретное косинусное преобразование. Преобразует исходную матрицу данных в матрицу с частотными коэффициентами.

Квантование – преобразование матрицы в матрицу с уменьшенной областью значений.

Синдром ошибки – остаток от деления кодового слова или кодового слова с ошибкой на порождающий многочлен кода.

Сэплирование – сохранение цветовых компонент кадра с исходным или пониженным разрешением кадра.

ЦВЗ – цифровой водяной знак.

**Содержание**

[Реферат 2](#_Toc389210039)

[Определения, обозначения и сокращения 3](#_Toc389210040)

[Введение 6](#_Toc389210041)

[1. Основные понятия и обзор стеганографических методов 8](#_Toc389210042)

[1.1 Цветовые пространства 8](#_Toc389210043)

[1.1.1 RGB 8](#_Toc389210044)

[1.1.2 YCbCr 9](#_Toc389210045)

[1.2 Помехоустойчивое кодирование 10](#_Toc389210046)

[1.2.1 Двоичные циклические коды 11](#_Toc389210047)

[1.2.1.1 Кодирование 12](#_Toc389210048)

[1.2.1.2 Построение таблицы синдромов ошибок 13](#_Toc389210049)

[1.2.1.3 Декодирование и исправление ошибок 14](#_Toc389210050)

[1.3 Модель сжатия MPEG видео 15](#_Toc389210051)

[1.3.1 Сэмплирование 15](#_Toc389210052)

[1.3.2 Дискретное косинусное преобразование 16](#_Toc389210053)

[1.3.3 Квантование 18](#_Toc389210054)

[1.4 Обзор стеганографических методов 19](#_Toc389210055)

[1.4.1 Методы, работающие с исходным изображением 20](#_Toc389210056)

[1.4.2 Методы, работающие на этапе преобразований 21](#_Toc389210057)

[1.4.3 Методы, работающие на этапе сжатия 21](#_Toc389210058)

[2. Алгоритмы встраивания и извлечения информации 22](#_Toc389210059)

[2.1 Встраивание информации 22](#_Toc389210060)

[2.1.1 Преобразование исходного видео контейнера 22](#_Toc389210061)

[2.1.2 Преобразование встраиваемой информации 23](#_Toc389210062)

[2.1.3 Встраивание данных 23](#_Toc389210063)

[2.1.4 Обратное преобразование в стего видео контейнер 24](#_Toc389210064)

[2.2 Извлечение информации 24](#_Toc389210065)

[2.2.1 Извлечение данных 24](#_Toc389210066)

[2.2.2 Преобразование извлечённых данных 25](#_Toc389210067)

[3. Программная реализация 26](#_Toc389210068)

[3.1 Выбор средств реализации 26](#_Toc389210069)

[3.2 Описание библиотеки AForge.Video.FFMPEG 26](#_Toc389210070)

[3.3 Описание классов программы 26](#_Toc389210071)

[3.4 Окно программы 29](#_Toc389210072)

[3.5 Реализация дискретного косинусного преобразования 29](#_Toc389210073)

[3.6 Реализация помехоустойчивого кодирования 31](#_Toc389210074)

[3.7 Реализация встраивания 32](#_Toc389210075)

[3.8 Реализация извлечения 34](#_Toc389210076)

[Заключение 36](#_Toc389210077)

[Список использованных источников 37](#_Toc389210078)

Приложение А. Техническое задание 39

Приложение Б. Руководство оператора 51

Приложение В. Программа и методика испытаний 66

Приложение Г. Текст программы 79

# Введение

В настоящее время в сети Интернет находится большое количество различного мультимедийного контента такого, как цифровые изображения, звуковые файлы, например, музыка, а также видеозаписи. Весь этот контент может быть использован в качестве контейнеров для сокрытия информации от посторонних глаз. Стеганография – специальная наука, изучающая различные методы сокрытия информации в мультимедийных объектах [11], которая должна оставаться необнаружимой как статистически, так и для человеческого восприятия [12]. Другими словами, стеганография скрывает сам факт того, что какая-либо информация скрыта.

У каждого стеганографического метода можно выделить набор качественных характеристик. Выделяют три основные характеристики: невидимость, устойчивость и объём [8]. Невидимость показывает, насколько хорошо спрятана информация, что она остаётся незаметной для человеческого восприятия и статистических методов [5]. Устойчивость означает стойкость контейнера к ошибкам, а именно как модификация контейнера влияет на сокрытые в нём данные [9]. Под модификацией подразумевается применение различных фильтров, обрезки, изменения размеров контейнера [4]. Объём определяет количество информации, которое можно встроить в контейнер. Видеофайлы занимают большой размер с точки зрения хранения информации, поэтому они позволяют внедрить наибольший объём информации.

Актуальность данной темы напрямую связана с информационной безопасностью и защитой информации. Стеганографические методы очень схожи с технологиями, применяемыми для внедрения цифровых водяных знаков (ЦВЗ). ЦВЗ представляет собой некую цифровую подпись, которая встраивается в мультимедийный объект с целью защиты авторских прав [8]. Отличие ЦВЗ от стеганографии заключается в том, что стеганография больше нацелена на невидимость внедряемой информации для человеческого восприятия, а методы встраивания ЦВЗ делают упор на устойчивость внедряемой информации. Внедрение цифровых подписей позволяет определить владельца информации и отслеживать её незаконное распространение [10].

Целью данной работы является разработка программы, которая позволила бы встраивать информацию в видеофайл в формате MPEG и извлекать информацию из видеофайла в формате MPEG, скрыв сам факт встраивания.

Для достижения цели работы необходимо решить следующие задачи:

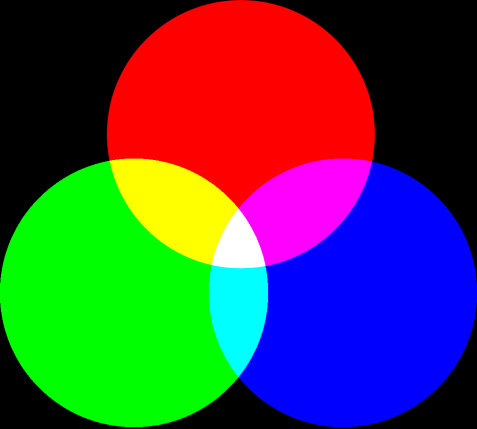
1. Изучить материалы по кодированию MPEG видео: структуре формата MPEG и модели сжатия;
2. Изучить стеганографические подходы к сокрытию информации в видео контейнерах и используемые ими алгоритмы и модели;
3. Выбрать стеганографический подход для реализации;
4. Выбрать технологии и средства реализации;
5. Разработать программу для встраивания информации в видеофайл в формате MPEG и извлечения информации из видеофайла в формате MPEG c использованием выбранного метода;
6. Разработать техническую документацию согласно техническому заданию;
7. Протестировать программу на соответствие требованиям.

# Основные понятия и обзор стеганографических методов

# Цветовые пространства

# RGB

Цветовое пространство состоит из трёх цветовых каналов: красного (red), зелёного (green), синего (blue). Данная модель является аддитивной, то есть итоговый цвет образуется в результате слияния трёх компонент (рис. 1.1). С помощью различных комбинаций этих трёх каналов можно получить любой цвет [16]. Количество цветов зависит от количества бит, которое отводится для хранения численного значения компоненты цвета. Наиболее распространён формат , в котором на каждую цветовую компоненту отводится по восемь бит, соответственно, числовое значение компоненты лежит в интервале [0; 255].



*Рисунок 1.1. Аддитивность каналов RGB*

Данная цветовая модель находит широкое применение в современной технике. Например, жидкокристаллические дисплеи состоят из ячеек, каждая из которых содержит три пикселя красного, зеленого и синего цвета. В зависимости от изображения на экране каждый пиксель подсвечивается с определённой интенсивностью. Благодаря аддитивности модели, со стороны зрителя три пикселя разных цветов сливаются в один цвет [16].



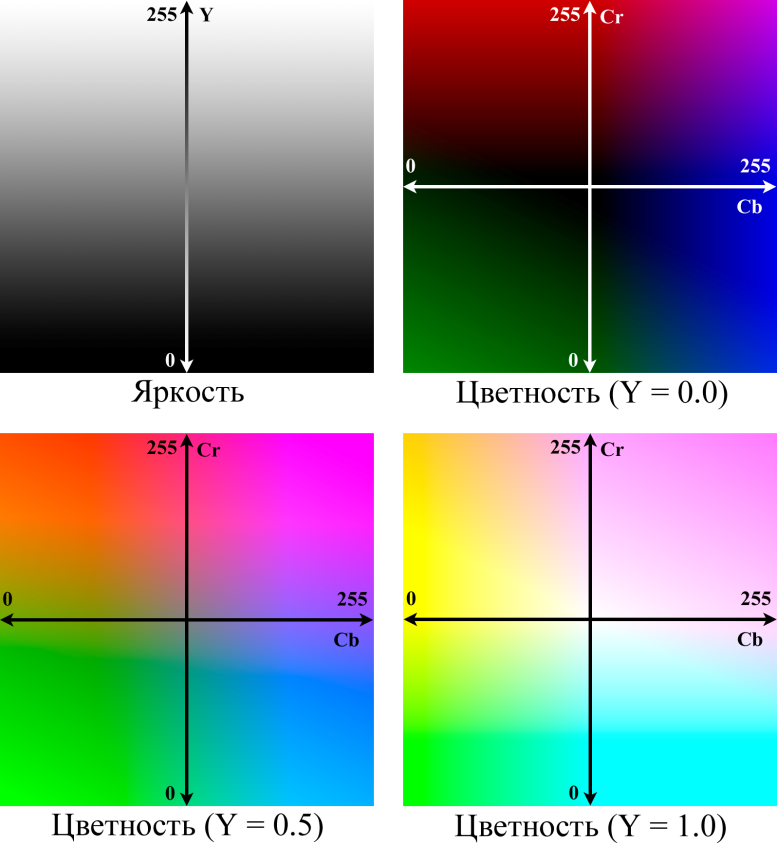
*Рисунок 1.2. Три цветовых канала RGB модели*

**

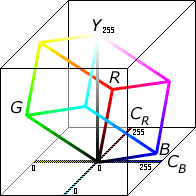
*Рисунок 1.3. Слияние трёх каналов в один общий цвет*

# YCbCr

Цветовое пространство состоит из трёх компонент: компоненты яркости и двух компонент цветности и (рис. 1.4-1.5). Такое разделение вызвано тем, что человеческое зрение имеет большую чувствительность к яркости, чем к цвету предмета [16]. В связи с этим компоненты цветности и можно хранить с меньшим разрешением, что позволяет уменьшить объём сохраняемых или передаваемых данных [16]. Поэтому данное цветовое пространство широко используется в цифровых изображениях и видео. Также цветовое пространство очень часто используется в стеганографических алгоритмах, для того чтобы работать с чёрно-белым изображением, используя канал яркости.



*Рисунок 1.4. Цветовое пространство*



*Рисунок 1.5. 3D представление цветового пространства*

Перевести изображение из цветового пространства в пространство можно с помощью формулы (1.1), используемой форматом JPEG [3]. В данной формуле компоненты пространства и пространства лежат в интервале .

Обратное преобразование из цветового пространства в пространство можно выполнить с помощью формулы (1.2), в которой компоненты обоих пространств также должны находиться в интервале .

# Помехоустойчивое кодирование

Некоторые авторы [13, 6] используют в алгоритмах встраивания информации помехоустойчивые коды для увеличения устойчивости сокрытых данных. Помехоустойчивое кодирование позволяет обнаруживать и исправлять определённое количество ошибок в данных, закодированных с помощью помехоустойчивых кодов. Оно добавляет к исходному сообщению долю избыточной информации, благодаря которой возможно обнаружение и исправление ошибок. Одним из типов помехоустойчивых кодов являются двоичные циклические коды.

# Двоичные циклические коды

Двоичные систематические циклические коды входят во множества различных типов кодов, благодаря которым процессы кодирования и декодирования легко описать и не сложно реализовать:

* Линейные коды

Коды данного множества позволяют с помощью линейной алгебры описать процессы кодирования и декодирования [15].

* Блоковые коды

Коды данного множества являются кодами фиксированной длины. При кодировании информационное слово длины преобразуется в кодовое слово длины .

* Двоичные коды

Коды данного множества используют модулярную арифметику по модулю 2, которая оперирует символами 0 и 1.

* Систематические коды

Кодовые слова данного множества можно разделить на две части: информационное (исходное) слово и проверочное (добавочное) слово.

* Циклические коды

Циклический сдвиг каждого кодового слова данного подмножества также является кодовым словом.

Определяя конкретный двоичный циклический код, используют три параметра:

* – длина кодового слова;
* – длина информационного (исходного) слова;
* – порождающий многочлен кода.

где – порождающий многочлен -кода,

,

,

– обозначение места соответствующей компоненты в кодовом слове (векторе),

Многочлен можно представить в виде двоичного слова . Порождающим многочленом циклического -кода может являться любой многочлен делящий многочлен без остатка [15]. Он назван порождающим, потому что он в буквальном смысле порождает циклический код, то есть преобразует информационное слово в кодовое.

# Кодирование

Кодирование с помощью двоичного циклического -кода с порождающим многочленом происходит по следующим шагам [15, гл. 3.3]:

1. Берётся информационное слово длины (), которое можно записать в виде многочлена .
2. Информационный многочлен умножается , где . В представлении двоичного слова данная операция является сдвигом влево на разрядов (.
3. Вычисляется остаток от деления многочлена , полученного на шаге 2, на порождающий многочлен . Этот остаток записывается в нулевые разряды, полученные после сдвига, формируя кодовый многочлен .

Ниже представлен пример кодирования информационного слова (1010) с помощью двоичного циклического (7,4)-кода с порождающим многочленом .

1. Записываем слово в виде многочлена
2. Умножаем информационный многочлен на .
3. Находим остаток от деления с помощью алгоритма деления Евклида

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | = |
| 1 | 0 | 1 | 1 | 0 | 0 | 0 | = |
|  |  |  | 1 | 0 | 0 | 0 | = |
|  |  |  | 1 | 0 | 1 | 1 | = |
|  |  |  |  | 0 | 1 | 1 | = |
|  |  |  |  |  |  |  |  |

1. Записываем остаток в нулевые разряды после сдвига и получаем кодовое слово (1010011) и соответствующий кодовый многочлен.

# Построение таблицы синдромов ошибок

Перед тем как начать декодирование, необходимо построить таблицу синдромов ошибок, с помощью которой будет происходить исправление ошибок. Для этого нужно знать количество ошибок, которое двоичный циклический -код может исправлять. Количество исправляемых ошибок для двоичного циклического -кода рассчитывается по следующей формуле.

где - количество исправляемых ошибок,

– минимальное кодовое расстояние.

Минимальное кодовое расстояние для двоичного циклического -кода рассчитывается как минимальный вес среди его всех ненулевых кодовых слов. Весом кодового слова является количество единичных бит в этом слове. Для двоичного циклического (7,4)-кода существует всего кодовых слов (табл. 1). Для этого кода минимальное кодовое расстояние . Подставляя это значение в формулу (1.3), получаем что, двоичный циклический (7,4)-код исправляет одну ошибку.

*Таблица 1.1*

Кодовые слова двоичного циклического (7,4)-кода ()

|  |  |  |  |
| --- | --- | --- | --- |
| 0000000 | 0100111 | 1000101 | 1100010 |
| 0001011 | 0101100 | 1001110 | 1101001 |
| 0010110 | 0110001 | 1010011 | 1110100 |
| 0011101 | 0111010 | 1011000 | 1111111 |
|  |  |  |  |

Так как двоичный циклический (7,4)-код исправляет всего одну ошибку, то общее количество многочленов ошибок равно длине кодового слова . Для каждого многочлена ошибки находится его синдром, остаток от деления на порождающий многочлен (табл. 1.2).

*Таблица 1.2*

Синдромы однократных ошибок двоичного циклического (7,4)-кода с порождающим многочленом .

|  |  |
| --- | --- |
| Ошибка | Синдром |
| 0000001 | 001 |
| 0000010 | 010 |
| 0000100 | 100 |
| 0001000 | 011 |
| 0010000 | 110 |
| 0100000 | 111 |
| 1000000 | 101 |
|  |  |

# Декодирование и исправление ошибок

Декодирование двоичного циклического -кода с порождающим многочленом происходит по следующим шагам [15, гл. 3.8]:

1. Вычисляется синдром ошибки с помощью алгоритма деления Евклида, описанном в 1.2.1.1.
2. Если синдром нулевой, то кодовое слово не содержит ошибок. Если синдром ненулевой, то определятся ошибочный бит с помощью таблицы синдромов и исправляется.
3. Так как кодирование систематическое, то можно просто отсечь проверочную часть длины и получить декодированное информационное слово.

Ниже представлен пример декодирования кодового слова (0010011) с помощью двоичного циклического (7,4)-кода с порождающим многочленом .

1. Находим остаток от деления (синдром ошибки) с помощью алгоритма деления Евклида

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
| 0 | 0 | 1 | 0 | 0 | 1 | 1 | = |
| 0 | 0 | 1 | 0 | 1 | 1 | 0 | = |
|  |  |  |  | 1 | 0 | 1 | = |
|  |  |  |  |  |  |  |  |

1. Синдром не равен нулю, поэтому находим синдром в таблице синдромов (табл. 1.2) и соответствующий ему ошибочный бит (1000000).
2. Исправляем ошибочный бит и получаем правильное кодовое слово (1010011). Отсекаем проверочную часть и получаем информационное слово (1010).

# Модель сжатия MPEG видео

MPEG (Motion Picture Experts Group) – группа экспертов, которая разрабатывает, обновляет и поддерживает стандарты сжатия цифровой аудио и видео информации [16]. Группа MPEG разработала множество стандартов, наиболее важными из которых являются MPEG-1, MPEG-2 и MPEG-4. MPEG-1 – самый первый стандарт, который был разработан для сжатия аудио и видео объектов с последующей записью на компакт-диск. Стандарт MPEG-2 используется в телевизионном вещании и в DVD видео. Стандарт MPEG-4 обладает более высоким коэффициентом сжатия по сравнению с MPEG-2 и позволяет оперировать объектами (изображения, трёхмерные модели, текстовые данные).

Высокая степень сжатия в стандартах MPEG достигается с помощью методов сжатия информации с потерями. Эти методы устраняют значительное количество межкадровой и внутрикадровой избыточности. Краткая модель сжатия MPEG приведена на рисунке 1.6.

*Рисунок 1.6. Модель сжатия MPEG*

Выбор сэмплирования (цветовой субдискретизации)

Несжатое видео

Прогнозирование и компенсация движения

ДКП

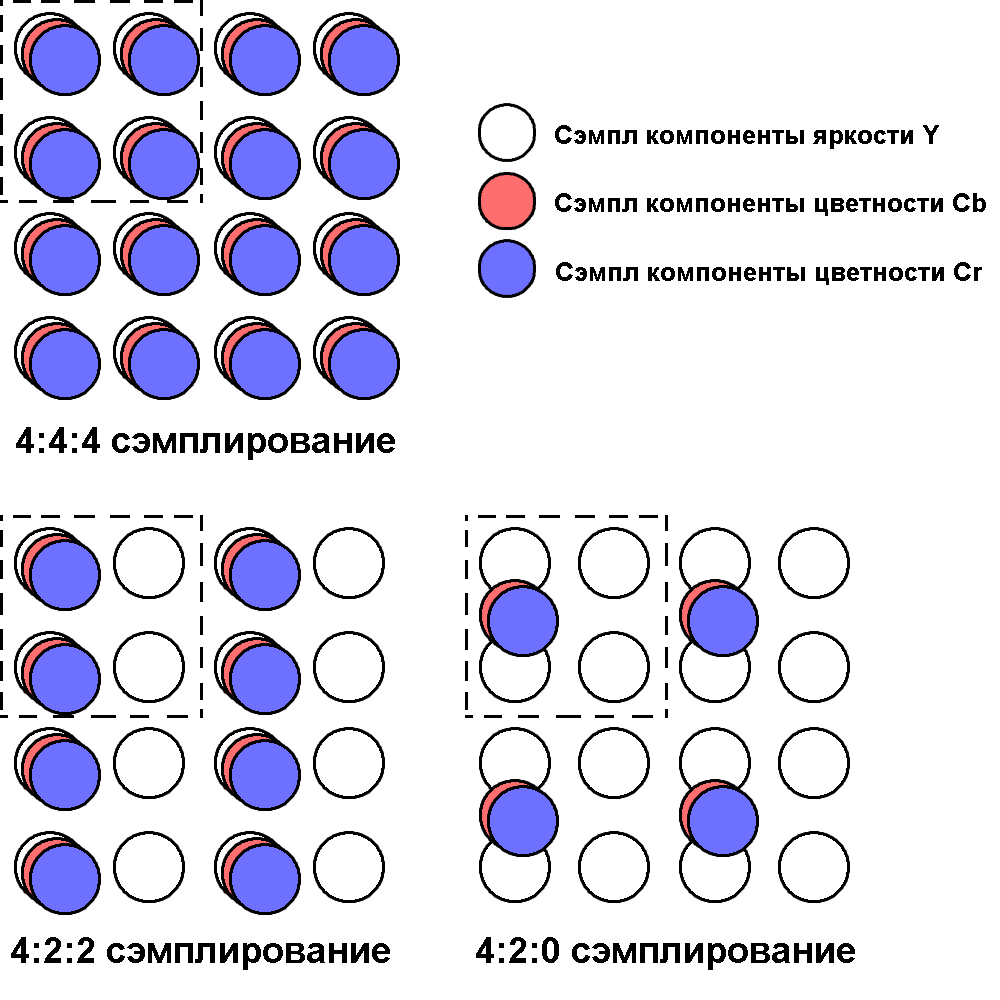
Квантование

Энтропийное кодирование

Для последующего понимания применимости стеганографических методов следует понимать этапы сжатия, на которых происходит потеря информации. Это этапы сэмплирования и квантования. Описание остальных этапов можно найти в [16].

# Сэмплирование

Цветовое сэмплирование (субдискретизация) применяется к кадрам в цветовом пространстве , которое используется форматом MPEG. Сэмплирование подразумевает выбор разрешения, в котором будут сохранены компоненты , и . Наиболее популярными форматами сэмплирования являются 4:2:0, 4:2:2 и 4:4:4 (рис. 7).



*Рисунок 1.7. Сэмплирование 4:4:4, 4:2:2 и 4:2:0*

Сэмплирование 4:4:4 означает, что компоненты цветности и сохраняют исходное разрешение, на каждые четыре сэмпла яркости приходится по четыре сэмпла компонент цветности и . Сэмплирование 4:2:2 означает, что на четыре сэмпла яркости приходится по два сэмпла компонент цветности и , то есть они имеют лишь половину от горизонтального разрешения яркости. В сэмплировании 4:2:0 на четыре сэмпла яркости приходится лишь по одному сэмплу компонент цветности и , они имеют половину от горизонтального и вертикального разрешения. Сохраняя лишь одну компоненту цветности вместо четырёх, отсекаются значительные объемы информации, что в итоге даёт хороший уровень сжатия. Поэтому сэмплирование 4:2:0 так популярно, что используется в цифровом телевидении и DVD [16].

# Дискретное косинусное преобразование

Известно, что пиксели на изображении коррелируют со своими соседями, потому что значение конкретного пикселя можно предсказать по его соседям. Дискретное косинусное преобразование (ДКП) уменьшает эту избыточность между пикселями [10]. Оно преобразует исходную матрицу данных в матрицу некоррелированных величин (1.4), используя суммы косинусов на разных частотах. ДКП имеет обратное преобразование (1.5).

где – исходная матрица,

– матрица коэффициентов ДКП,

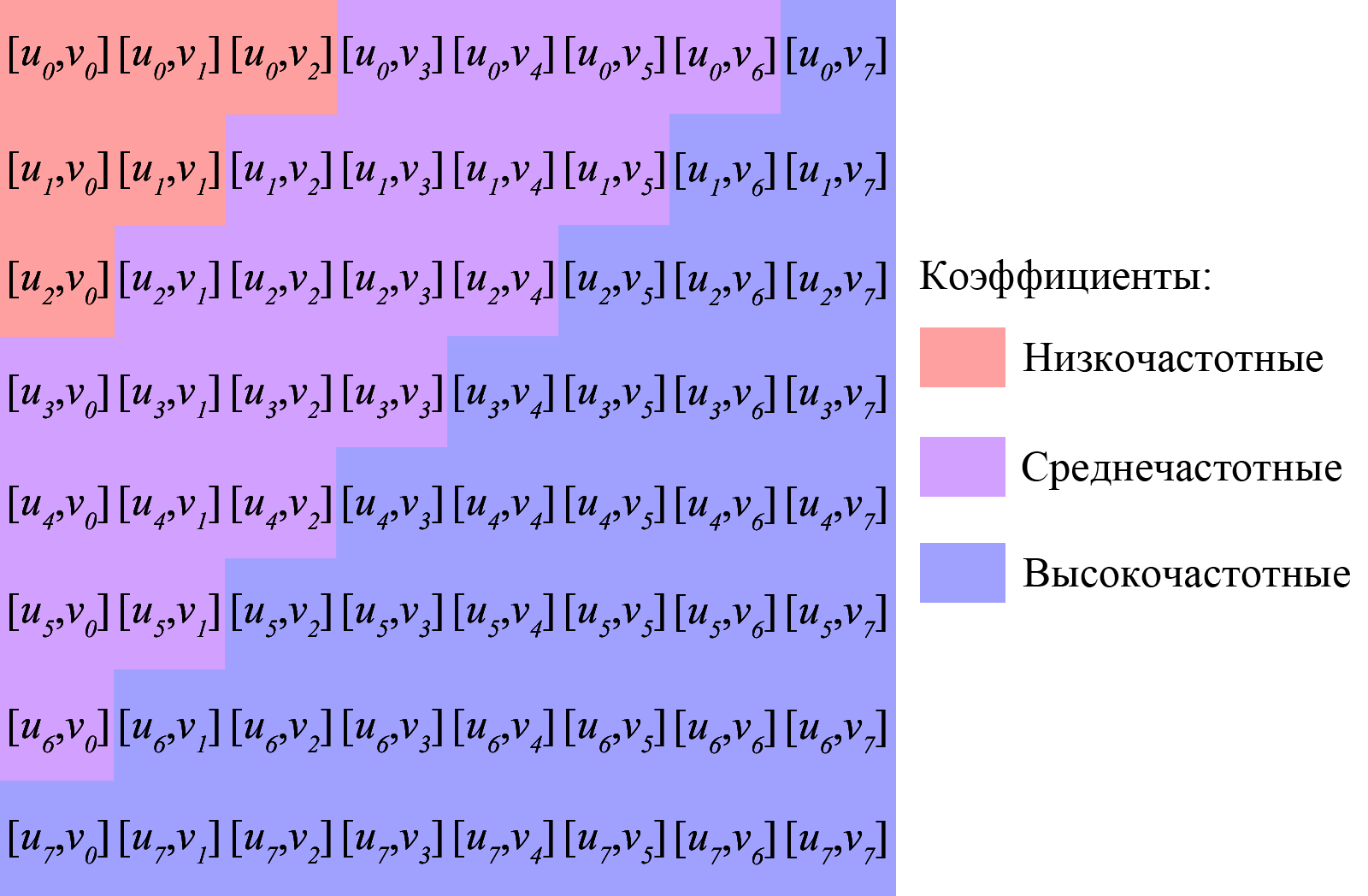
– размер матриц и ,

,

*=* , или  *=*

.

Преобразование происходит таким образом, что коэффициенты матрицы ДКП получаются упорядоченными по частоте. Сначала следуют низкочастотные коэффициенты, затем среднечастотные и высокочастотные (рис. 1.8). Низкочастотные коэффициенты содержат самую важную информацию для восстановления исходных данных, и их изменение приведёт к сильному искажению данных после применения обратного преобразования. Высокочастотные коэффициенты можно отрезать (занулить) без сильного воздействия на данные после применения обратного преобразования, что и происходит на этапе квантования.



*Рисунок 1.8. Матрица коэффициентов ДКП размером 8х8*

В модели сжатия MPEG в качестве данных для матрицы берутся числовые значения компонент цветового пространства . Для этого компоненты , и разбивается на блоки размером 8х8 или 4х4, и затем к каждому из блоков применяется ДКП.

# Квантование

На этапе квантования матрица коэффициентов ДКП преобразуется в новую матрицу с уменьшенной областью значений [16]. Преобразование выполняется последовательным делением каждого коэффициента матрицы ДКП на значение шага квантования и округлением полученного значения (1.6).

где – квантованная матрица коэффициентов ДКП,

– исходная матрица коэффициентов ДКП,

– шаг квантования,

,

– размер матриц и ,

– операция округления.

Ниже представлен пример квантования матрицы коэффициентов ДКП с шагом 5

Из-за операции округления часть данных теряется, и при применении обратного квантования (1.7) при декодировании полученная матрица будет в некоторой степени отличаться от исходной .

где – матрица коэффициентов ДКП, прошедшая квантование и обратное квантование,

– квантованная матрица коэффициентов ДКП,

– шаг квантования,

,

– размер матриц и .

Ниже представлен пример обратного квантования матрицы коэффициентов ДКП с шагом 5

# Обзор стеганографических методов

В настоящее время существует достаточно много подходов для сокрытия данных в видеофайлах [14], и новые подходы всё продолжают появляться. На рисунке 1.9 изображена краткая модель сжатия MPEG, и выделены этапы, на которых возможно встраивание информации.

Выбор сэмплирования (цветовой субдискретизации)

Несжатое видео

Прогнозирование и компенсация движения

ДКП

Квантование

Энтропийное кодирование

*Рисунок 1.9. Этапы модели сжатия MPEG на которых возможно встраивание информации*

# Методы, работающие с исходным изображением

Методы данной категории работают напрямую с изображением. Наиболее популярным является метод наименее значимых бит (*от англ. Least Significant Bit (LSB)*). Данный метод подразумевает изменение наименее значимых бит цветовых компонент изображения (кадра), чтобы минимизировать искажения картинки в целом [7]. Допустим, используется изображение в формате RGB, в котором отводится по 1 байту на каждую цветовую компоненту. Тогда изменение одного иди двух наименее значимых бит в каждой из компонент (рис. 1.10) не окажет влияния на визуальное восприятие картинки в целом. Под изменением подразумевается замена исходных бит на биты внедряемого сообщения.



*Рисунок 1.10. Наименее значимые биты (белые) в формате RGB*

Однако невидимость информации, внедрённой с помощью этого способа, может быть достаточно слабой. Шум, внедрённый в участки изображения с плавным переходом, может быть визуально замечен [7]. Также данный метод применим только к несжатому видео, так как при сжатии малозначимая информация попросту отсекается. Поэтому у данного метода отсутствует какая-либо устойчивость, зато он позволяет встраивать значительный объём информации.

# Методы, работающие на этапе преобразований

Такие методы, которые предполагают частичное или полное преобразование исходных данных называют методами, работающими на этапе преобразований. Примером такого подхода является метод ДКП.

Перед использованием метода ДКП кадры видео делятся на блоки равного размера, которые затем преобразуются с помощью ДКП. Деление на блоки происходит только по каналу яркости. Затем информационные биты внедряются в полученные коэффициенты [7]. Процесс внедрения осуществляется с помощью определённых алгоритмов. Например, возможна замена заранее определённых средне- или высокочастотных коэффициентов на информационные биты. Далее применяется обратное преобразование, и формируются исходные данные с внедренной информацией.

Такой подход позволяет уменьшить шум, произведённый внедрением, а также повысить устойчивость внедрённых данных.

# Методы, работающие на этапе сжатия

Методы данной категории направлены исключительно на работу со сжатыми форматами файлов. Такие методы встраивают информацию в контейнер и извлекают информацию из контейнера в середине процессов компрессии и декомпрессии, оперируя промежуточными данными, что даёт значительную скорость встраивания и извлечения [9].

Примерами такого подхода могут служить методы, использующие ДКП для встраивания информации, так как ДКП является одним из шагов в процессе сжатия JPEG и MPEG файлов.

**Выводы по главе**

Из рассмотренных подходов для реализации выбран метод ДКП, работающий на этапе преобразований. В дополнение к методу выбрано помехоустойчивое кодирование двоичными циклическими кодами для увеличения устойчивости встроенных данных.

# Алгоритмы встраивания и извлечения информации

Используемый алгоритм встраивания и извлечения информации основан на алгоритме, описанном Blackledge и Iakovenko в [1].

# Встраивание информации

Полный алгоритм встраивания информации можно разделить на четыре этапа (рис. 2.1):

1. Преобразование исходного видео контейнера
2. Преобразование встраиваемой информации
3. Встраивание данных
4. Обратное преобразование в стего видео контейнер.

Секретная информация

Видео контейнер

Преобразованные данные

Преобразованное видео

Преобразование

Преобразование

Встраивание

Стего видео контейнер

*Рисунок 2.1. Этапы встраивания информации*

# Преобразование исходного видео контейнера

Перед началом встраивания необходимо подготовить исходные видео данные для встраивания. Подготовка происходит в пять этапов:

1. Видео декодируется из формата MPEG-4.
2. Полученное видео разбивается на кадры в цветовом пространстве .
3. Каждый кадр преобразовывается из цветового пространства в пространство (1.1).
4. Каждый кадр разбивается на блоки размером , где . Меньшие значения позволяют встроить большее количество информации, так как количество блоков возрастает, но и увеличивает шум.
5. Выбираются блоки компоненты яркости и преобразуются с помощью ДКП (1.4) в матрицы размером .

# Преобразование встраиваемой информации

Перед началом встраивания необходимо закодировать внедряемую информацию с помощью помехоустойчивых кодов. Кодирование происходит в пять этапов:

1. Выполняется преобразование внедряемой информации в битовую последовательность.
2. Выбирается двоичный циклический -код для кодирования информации.
3. Битовая последовательность делится на отрезки (информационные слова) длины .
4. Каждое информационное слово кодируется с помощью выбранного двоичного циклического -кода (гл. 1.2.1).
5. Все полученные кодовые слова соединяются в одну битовую последовательность.

# Встраивание данных

После того как исходные видео данные и внедряемые данные преобразованы, можно приступать к встраиванию. Для каждой матрицы коэффициентов ДКП, полученной после преобразования исходного видеоконтейнера, выполняются следующие шаги:

1. Берётся самый первый бит из битовой последовательности, полученной на этапе преобразования встраиваемой информации.
2. Выбирается значение уровня шума (стойкости) , от которого зависит устойчивость встроенных данных. И чем больше это значение, тем сильнее будут искажены кадры видео после встраивания.
3. Наиболее высокочастотный коэффициент матрицы ДКП заменяется значением стойкости (2.1).

где – исходная матрица коэффициентов ДКП,

– размер матрицы ДКП,

*–* коэффициент стойкости,

– встраиваемый бит.

1. Самый первый бит удаляется из битовой последовательности.

Встраивание длится до тех пор, пока битовая последовательность не станет пуста.

# Обратное преобразование в стего видео контейнер

После встраивания информации все блоки должны быть собраны в кадры и снова закодированы в формат MPEG-4.

1. Для каждого блока применяется обратное ДКП (1.5);
2. Блоки соединяются в кадры;
3. Кадры преобразовываются из цветового пространства в пространство (1.2);
4. Видео кодируется в формат MPEG-4.

# Извлечение информации

Алгоритм извлечения можно разбить на три этапа (рис. 2.2):

1. Преобразование стего видео контейнера;
2. Извлечение данных;
3. Преобразование извлечённых данных.

Стего видео контейнер

Преобразование

Преобразованное видео

Извлечение

Извлеченные данные

Секретные данные

Преобразование

*Рисунок 2.2. Этапы извлечения встроенной информации*

Первый шаг, преобразование стего видео контейнера, выполняется так же, как и при встраивании. Только в данном случае необходимо знать размер матрицы ДКП , который использовался на этапе встраивания.

# Извлечение данных

После того преобразования стего видео данных, можно приступать к извлечению. Для извлечения данных необходимо знать длину встроенного сообщения. Для каждой матрицы коэффициентов ДКП выполняются следующие шаги:

1. Проверяется наиболее высокочастотный коэффициент матрицы ДКП. Если он положителен, то извлекаемый бит ‘0’, если отрицательный – ‘1’ (2.2).

где – извлекаемый бит,

– матрица коэффициентов ДКП со встроенной информацией,

– размер матрицы ДКП.

1. Извлечённый бит записывается в новую битовую последовательность.

# Преобразование извлечённых данных

По завершению процесса извлечения необходимо декодировать данные и исправить возможные ошибки:

1. Для двоичного циклического -кода, использованного при встраивании информации, строится таблица символов (гл. 1.2.1.2).
2. Битовая последовательность, полученная на предыдущем шаге извлечения данных, делится на отрезки (кодовые слова) длины .
3. Для каждого кодового слова проверяется синдром ошибки, и если ошибка присутствует, то она исправляется (гл. 1.2.1.3).
4. Декодированные информационные слова длины соединяются в битовую последовательность.
5. Битовая последовательность преобразуется в формат исходных данных.

# Программная реализация

# Выбор средств реализации

Для реализации программы стеганографического сокрытия данных в видеофайлах формата MPEG был выбран язык программирования C#, среда разработки Visual Studio 2010, графического интерфейс Windows Forms и платформа .NET Framework 3.5. Для кодирования и декодирования видео в формате MPEG-4 используется библиотека «AForge.Video.FFMPEG» [2].

# Описание библиотеки AForge.Video.FFMPEG

Библиотека AForge.Video.FFMPEG предоставляет доступ к классам для покадрового чтения и записи видеофайлов. При реализации программы использовались два класса из данной библиотеки: VideoFileReader и VideoFileWriter.

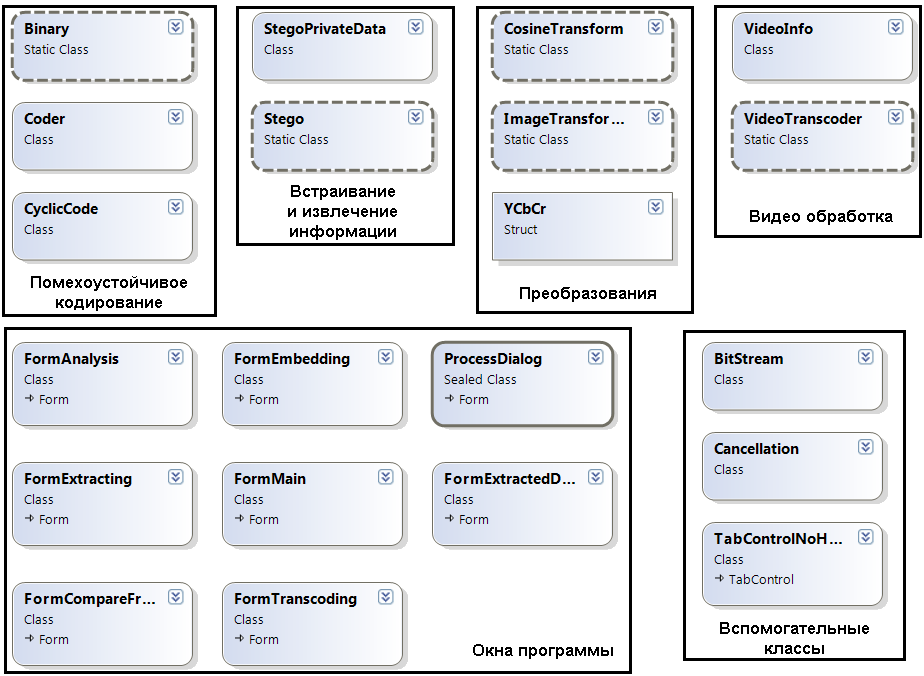
Класс VideoFileReader позволяет считывать основную информацию о видео (используемый кодек, количество кадров, размеры видео) и в режиме потокового чтения декодировать видео, получая видеокадры.

Класс VideoFileWriter позволяет покадрово записывать видеофайлы, используя различные кодеки, в том числе и MPEG-4.

Основное преимущество данной библиотеки состоит в том, что она основана на библиотеке FFMPEG, которая содержит реализацию большого количества кодеков. Поэтому для использования данной библиотеки не требуется установка кодеков.

# Описание классов программы

В программе присутствуют шесть групп классов (рис. 3.1).



*Рисунок 3.1. Группы классов программы*

Классы, связанные с обработкой видео:

1. VideoTranscoder.cs

Содержит методы для открытия и транскодирования видео, а также для встраивания и извлечения информации из видеофайлов.

1. VideoInfo.cs

Используется для хранения и передачи основных видео параметров между методами.

Классы, связанные со встраиванием и извлечением информации:

1. Stego.cs

Содержит методы встраивания и извлечения информации из видеокадров.

1. StegoPrivateData.cs

Используется для хранения и передачи информации о встраивании и извлечении между методами.

Классы, связанные с помехоустойчивым кодированием:

1. CyclicCode.cs

Содержит информацию о параметрах двоичного циклического кода.

1. Coder.cs

Используется для кодирования и декодирования с помощью двоичных циклических кодов.

1. Binary.cs

Cодержит вспомогательные методы для работы с бинарными числами.

Классы основных окон программы:

1. FormMain.cs

Главное окно программы.

1. FormEmbedding.cs

Окно встраивания информации.

1. FormAnalysis.cs

Окно сравнения информации (пост анализа).

1. FormCompareFrames.cs

Окно сравнения кадров.

1. FormTranscoding.cs

Окно транскодирования видео.

1. FormExtarcting.cs

Окно извлечения информации.

1. FormExtractedData.cs

Окно для отображения извлечённой информации.

1. ProcessDialog.cs

Окно обработки информации.

Вспомогательные классы:

1. BitStream.cs

Класс битовой последовательности.

1. Cancellation.cs

Содержит токен отмены, который передаётся в другой поток.

1. TabControlNoHeaders.cs

Класс, унаследованный от класса *TabControl*. Содержит отключённые заголовки страниц.

Классы преобразований:

1. CosineTransform.cs

Содержит методы ДКП и обратного ДКП.

1. ImageTrasnformer.cs

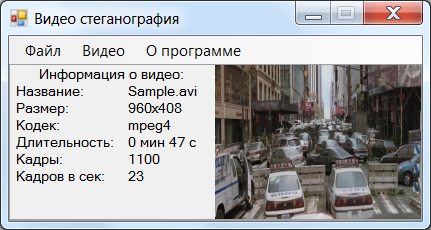
Содержит методы преобразований из цветового пространства в и из в , а также методы преобразований изображений в байтовый массив и обратно.

1. YCbCr.cs

Структура, содержащая значения цветового пространства .

# Окно программы

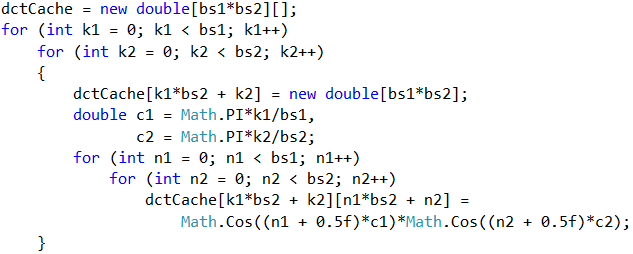
В главном окне программы (рис. 3.2) пользователю отображаются основные данные об открытом видеофайле и его превью. В разделе меню «Файл» предоставляются возможность открыть видеофайл и выйти из программы. В разделе меню «Видео» пользователь может транскодировать видеофайл в формат MPEG-4, встроить информацию в видеофайл и извлечь информацию из видеофайла. В разделе меню «О программе» отображается основная информация о программе.



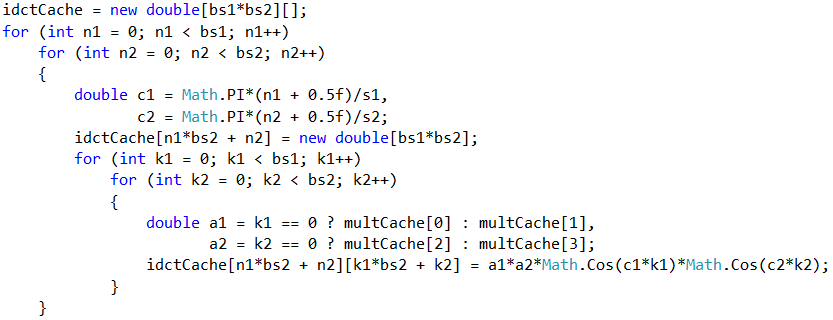
*Рисунок 3.2. Основное окно программы.*

# Реализация дискретного косинусного преобразования

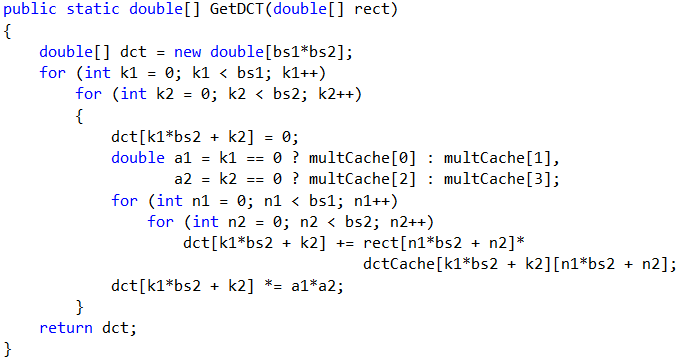
ДКП и обратное ДКП реализованы в классе CosineTransform с помощью формул (1.4) и (1.5). Эти формулы содержат большое количество умножений, и прямая их реализация сильно скажется на производительности работы программы. Однако можно заметить, что в данных формулах часть остаётся прежней при преобразовании матриц, меняются лишь данные преобразуемых матриц. Поэтому для увеличения производительности программы было решено просчитывать эти части формул заранее в кэш массив (рис. 3.3-3.4), чтобы впоследствии при преобразовании обращаться к этим значениям (рис. 3.5-3.6). Также для увеличения производительности используются одномерные массивы вместо двумерных, что значительно повышает скорость доступа к элементам массива.



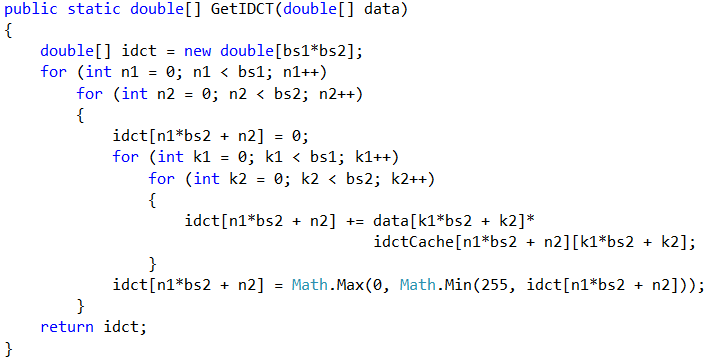
*Рисунок 3.3. Генерация кэш массива для ДКП*

**

*Рисунок 3.4. Генерация кэш массива для обратного ДКП*

**

*Рисунок 3.5. Вычисление ДКП*

**

*Рисунок 3.6. Вычисление обратного ДКП*

# Реализация помехоустойчивого кодирования

Для реализации помехоустойчивого кодирования использовались двоичные циклические коды. Их реализация состоит из трёх классов: CyclicCode, Binary и Coder.

Класс CyclicCode используется для обозначения определенного двоичного циклического -кода с порождающим многочленом . В данной работе используются три типа кода:

* (3,1)-код с порождающим многочленом (Код Хэмминга)

Код позволяет исправлять одну ошибку в кодовом слове.

* (7,4)-код с порождающим многочленом (Код Хэмминга)

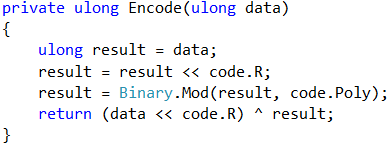
Код позволяет исправлять одну ошибку в кодовом слове. Однако избыточность данного кода меньше, слова из четырёх бит преобразуются в слова из семи бит.

* (23,12)-код с порождающим многочленом (Код Голея)

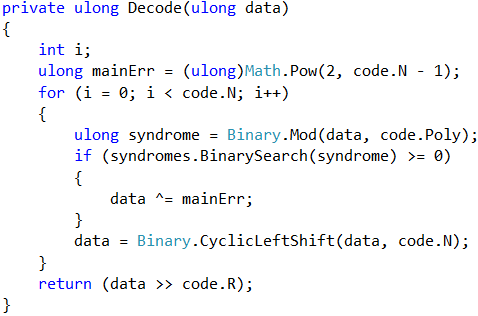
Код позволяет исправлять три ошибки в кодовом слове.

Класс Binary содержит вспомогательные методы, используемые классом Coder, такие как циклический сдвиг кодового слова и вычисление остатка по алгоритму деления Евклида, описанный в пункте 1.2.1.1.

Класс Coder предоставляет возможность кодирования и декодирования двоичными циклическими кодами. Он реализует метод генерации таблицы синдромов, описанной в пункте 1.2.1.2, методы кодирования и декодирования отдельного слова (рис. 3.7-3.8), и методы кодирования и декодирования байтовой последовательности.



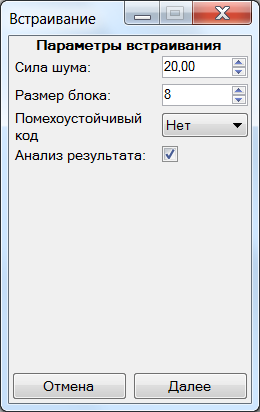
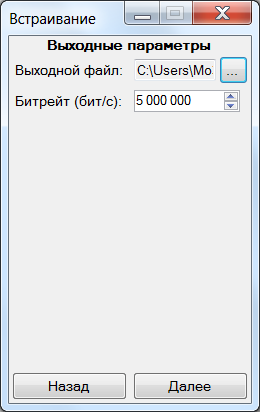
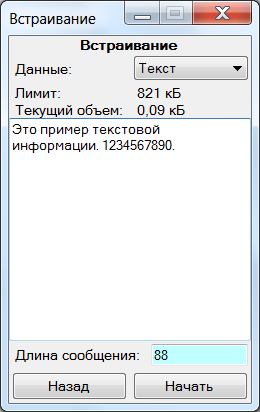
*Рисунок 3.7. Кодирование информационного слова в кодовое*

**

*Рисунок 3.8. Декодирование кодового слова в информационное*

# Реализация встраивания

Перед встраиванием пользователь вводит необходимую информацию и выбирает различные параметры встраивания в окне встраивания *FormEmbedding.cs* (рис. 3.9). Окно состоит из нескольких страниц, которые реализованы с помощью элемента TabControl. Однако у элемента TabControl нельзя отключить убрать заголовки страниц, поэтому был создан класс *TabControlNoHeaders.cs* с отключёнными заголовками, который наследуется от класса TabControl.

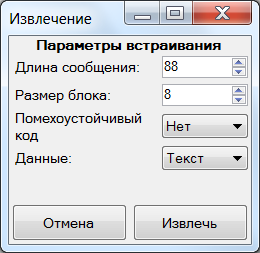
*Рисунок 3.9. Три шага встраивания информации.*

Встраивание происходит по следующим шагам:

1. Введённый пользователем секретный текст преобразуется в последовательность байт с помощью метода Encoding.Unicode.GetBytes.
2. Байтовой последовательность кодируется помехоустойчивым кодом с помощью метода Coder.EncodeFull.
3. Байтовая последовательность преобразуется в битовую с помощью класса BitStream и сохраняется в объекте класса StegoPrivateData вместе с выбранным пользователем уровнем шума и размером блока.
4. Объект класса StegoPrivateData передаётся в метод VideoTranscoder.EmbedData через класс ProcessDialog. В этом методе начинается покадровое чтение видео, где для каждого кадра вызывается метод Stego.EmbedData.
5. В методе Stego.EmbedData кадр преобразуется в  цветовое  пространство с помощью метода ImageTransformer.ConvertToRgb.
6. Из кадра в цветовом пространстве поочередно с помощью метода ImageTransformer.GetRectFromData выделяются блоки компоненты яркости одинакового размера, который был указан пользователем на шаге 1.
7. К каждому блоку применяется ДКП, которое реализовано в методе CosineTransform.GetDCT.
8. Из объекта битовой последовательности класса BitStream с помощью метода GetNext выбирается бит и встраивается в самый высокочастотный коэффициент матрицы ДКП (2.1).
9. Каждый блок преобразуется с помощью обратного ДКП, которое реализовано в методе CosineTransform.GetIDCT.
10. Видеокадр преобразуется из цветового пространства в с помощью класса ImageTransformer.ConvertToYcbcr.
11. Полученный кадр записывается в стего видео файл в методе VideoTranscoder.EmbedData.

# Реализация извлечения

В окне извлечения *FormExtracting.cs* (рис. 3.10) пользователь указывает параметры извлечения и длину сообщения, которая указывается в окне встраивания на третьем шаге.



*Рисунок 3.10. Окно извлечения информации*

Извлечение происходит по следующим шагам:

1. Параметры извлечения передаются в объекте класса StegoPrivateData в метод VideoTranscoder.ExtractData через класс ProcessDialog. В этом методе начинается покадровое чтение видео, где для каждого кадра вызывается метод Stego.ExtractData.
2. В методе Stego.ExtractData кадр преобразуется в  цветовое  пространство с помощью метода ImageTransformer.ConvertToRgb.
3. Из кадра в цветовом пространстве поочередно с помощью метода ImageTransformer.GetRectFromData выделяются блоки компоненты яркости одинакового размера, который был указан пользователем на шаге 1.
4. К каждому блоку применяется ДКП, которое реализовано в методе CosineTransform.GetDCT.
5. Встроенный бит извлекается из самого высокочастотного коэффициента матрицы ДКП (2.2) и заносится в объект битовой последовательности BitStream c помощью метода SetNext.
6. Битовая последовательность декодируется помехоустойчивым кодом с помощью метода Coder.DecodeFull.
7. Полученная байтовая последовательность преобразуется в текст с помощью метода Encoding.Unicode.GetBytes.

# Заключение

В результате данной работы, были выполнены следующие задачи:

1. Изучены материалы по кодированию MPEG видео: структуре формата MPEG и модели сжатия;
2. Изучены стеганографические подходы к сокрытию информации в видео контейнерах и используемые ими алгоритмы и модели;
3. Выбран стеганографический подход для реализации;
4. Выбраны технологии и средства реализации;
5. Разработана программа для встраивания информации в видеофайл в формате MPEG и извлечения информации из видеофайла в формате MPEG c использованием выбранного метода;
6. Разработана техническая документация согласно техническому заданию;
7. Программа протестирована на соответствие требованиям.

Результатом работы является программа стеганографического сокрытия информации в видеофайлах формата MPEG. В программе реализовано встраивание текстовой информации и изображений в видеофайлы формата MPEG и извлечение встроенной информации из видеофайлов в формате MPEG. В основе метода встраивания лежит метод дискретного косинусного преобразования. Для увеличения устойчивости встроенных данных использовалось помехоустойчивое кодирование двоичными циклическими кодами.

Программа может использоваться для передачи секретных данных на электронных носителях и по каналам связи.

В дальнейшем планируется увеличить скорость работы программы, изучить другие алгоритмы встраивания информации в видеофайлы и добавить их к текущему функционалу программы.

# Список использованных источников

1. Blackledge, J. Resilient Digital Image Watermarking for Document Authentication / J. Blackledge, O. Iakovenko // IAENG International Journal of Computer Science. – 2014. – № 41(1). – С. 1-17.
2. Codeplex. [Электронный ресурс]: AForge.Net Framework, Video.FFMPEG. – Электрон. дан. – Режим доступа: <http://aforgeffmpeg.codeplex.com>, свободный. (дата обращения: 26.05.14).
3. Hamilton, E. JPEG File Interchange Format [Электронный ресурс]: Version 1.02 / Hamilton E. – Электрон. текст. дан. – Milpitas, CA: C-Cube Microsystems, 1992 – Режим доступа: <http://www.jpeg.org/public/jfif.pdf>, свободный. (дата обращения: 28.04.14).
4. Hartung, F. Watermarking of Uncompressed and Compressed Video / F. Hartung, B. Girod // Signal Processing. – 1998. – № 66(3). – С. 283-301.
5. Ishtiaq, M. Robust and imperceptible watermarking of video streams for low power devices. / M. Ishtiaq, M. A. Jaffar, M. A. Khan, Z. Jan, A. M. Mirza. // Signal Processing, Image Processing and Pattern Recognition: International Conference, SIP 2009, Held as Part of the Future Generation Information Technology Conference, FGIT 2009, Jeju Island, Korea, December 10-12, 2009. Proceedings / D. Ślęzak, B. -H. Kang, T. Kim, H. Kuroda, S. K. Pal, J. Gu. – Berlin: Springer-Verlag Berlin Heidelberg. – C. 177-184.
6. Liu, Y. A robust data hiding algorithm for H.264/AVC video streams / Y. Liu, Z. Li, X. Ma, J. Liu // Journal of Systems and Software. – 2013. – № 86(8). – С. 2174-2183.
7. Lu, C. -S. Multimedia security: Steganography and digital watermarking techniques for protection of intellectual property / C. -S. Lu. – Hershey: Idea Group Publishing, 2005. – 255 c.
8. Masoumi, M. A blind scene-based watermarking for video copyright protection / M. Masoumi, S. Amiri // AEU - International Journal of Electronics and Communications. – 2013. – № 67(6). – С. 528-535.
9. Mobasseri, B. G. Lossless watermarking of compressed media using reversibly decodable packets / B. G. Mobasseri, D. Cinalli // Signal Processing. – 2006. – № 86(5). – С. 951-961.
10. Singh, H. V. Robust copyright marking using Weibull distribution / H. V. Singh, S. Rai, A. Mohan, S. P. Singh // Computers & Electrical Engineering. – 2011. – № 37(5). – С. 714-728.
11. Su, Y. A video steganalytic algorithm against motion-vector-based steganography / Y. Su, C. Zhang, C. Zhang // Signal Processing. – 2011. – № 91(8). – С. 1901-1909.
12. Swanson, M. D. Multimedia Data-Embedding and Watermarking Technologies / M. D. Swanson, M. Kobayashi, A. H. Tewfik // Proceedings of the IEEE. – 1998. – № 86(6). – С. 1064-1087.
13. Usman, I. BCH coding and intelligent watermark embedding: Employing both frequency and strength selection / I. Usman, A. Khan // Applied Soft Computing. – 2010. – № 10(1). – С. 332-343.
14. Xu, D. A novel watermarking scheme for H.264/AVC video authentication / D. Xu, R. Wang, J. Wang // Signal Processing: Image Communication. – 2011. – № 26(6). – С. 267-279.
15. Вернер, М. Основы кодирования: Учебник для ВУЗов / М. Вернер; пер. с нем. Д. К. Зигангирова. – Москва: Техносфера, 2004. – 288 с. – (Мир программирования).
16. Ричардсон, Я. Видеокодирование H.264 и MPEG-4 – стандарты нового поколения / Я. Ричардсон; пер. с англ. В. В. Чепыжова. – Москва: Техносфера, 2005. – 368 с. – (Мир цифровой обработки).