БЪЛГАРСКА АКАДЕМИЯ НА НАУКИТЕ
ИНСТИТУТ ПО ИНФОРМАЦИОННИ И КОМУНИКАЦИОННИ ТЕХНОЛОГИИ

Светозар Валериев Илчев

Модулни методи за вграждане на цифрова информация в изображения за подобряване сигурността на Интернет-базирани комуникационни платформи

ДИСЕРТАЦИЯ

за придобиване на образователна и научна степен „доктор“

по научна специалност
02.21.04 „Компютърни системи, комплекси и мрежи“

в професионално направление
5.3 „Комуникационна и компютърна техника“

Научен ръководител:
доц. д-р. Румен Андреев

София, 2013 г.

Съдържание

Списък на фигурите viii
Списък на таблиците xi
Списък на съкращенията xii
Списък на някои важни термини и понятия xiv
Глава 1 Увод 1
1.1 Актуалност на проблема 1
1.1.1 Предимства на криенето на данни пред криптографските подходи 2
1.1.2 Криене на данни за целите на Интернет-базирани сценарии 4
1.2 Области на приложение 5
1.2.1 Тайна комуникация 7
1.2.2 Доказване на авторство 8
1.2.3 Проверка на автентичността на мултимедия 9
1.2.4 Маркиране на мултимедийни копия 9
1.3 Основни научни области от значение за развитието на криене на данни в мултимедия 10
1.4 Цел и задачи на дисертацията 11
1.5 Структура на дисертацията 12
Глава 2 Анализ на съществуващите методи и програмни продукти 14
2.1 Важни свойства на криенето на данни в мултимедия 14
2.1.1 Адаптивност към променливи изисквания на потребителите 14
2.1.2 Устойчивост срещу JPEG-базирани трансформации 15
2.1.3 Работа с произволна мултимедия и вградени данни 16
2.2 Съществуващи методи и програмни продукти за криене на данни в мултимедия 17
2.2.1 Съществуващи научноизследователски разработки и методи 18

2.2.2 Съществуващи програмни продукти и услуги....21
2.3 Заключение....27
Глава 3 Модулен подход за проектиране на методи за криене на данни....29
3.1 Модулно проектиране – общ поглед и архитектура....29
3.2 Базов модул, устойчив срещу JPEG трансформации....33
3.2.1 Основни цели при разработването на модула....33
3.2.2 Общ поглед върху стандарта JPEG....34
3.2.3 Базов модул: кодиране....39
3.2.4 Гарантиране на устойчивост срещу JPEG компресия, декомпресия и рекомпресия....42
3.2.5 Базов модул: декодиране....45
3.3 Приложно-специфичен модул за целите на стеганографията. Модулен стеганографски метод....46
3.3.1 Направляващо файлово описание....47
3.3.2 Кодове за корекция на грешки....48
3.3.3 Разбъркване на данните....50
3.3.4 Кодиране на данните....52
3.3.5 Декодиране на данните....53
3.4 Приложно-специфичен модул за целите на цифровото маркиране. Модулен метод за цифрово маркиране....54
3.4.1 Направляващо описание и кодове за корекция на грешки....56
3.4.2 Макроблокове....57
3.4.3 Откриване на промени в изображението и възстановяване на водния знак....58
3.4.4 Кодиране на водния знак....60
3.4.5 Декодиране на водния знак....62
3.5 Базов модул за изображения с беззагубна компресия....63
3.5.1 Основни цели при разработването на модула....64
3.5.2 Базов модул: кодиране....64
3.5.3 Базов модул: декодиране....68
3.6 Заключение....69
Глава 4 Програмна реализация на модулните методи....71

4.1 Архитектура на програмната система ... 72
4.2 Пространство на имената на спомагателните функции ... 74
4.3 Слой на данните ... 76
4.4 Слой на базовите модули ... 78
4.5 Слой на приложно-специфичните модули ... 81
4.6 Интерфейсен слой ... 83
4.6.1 Графичен потребителски интерфейс ... 84
4.6.2 Графичен потребителски интерфейс за групови обработки ... 86
4.6.3 Филтри и хистограмен анализ ... 90
4.6.4 Приложен програмен интерфейс за мрежови услуги ... 91
4.7 Заключение ... 94
Глава 5 Оценка и стеганализ на разработените методи ... 96
5.1 Експериментални множества от носещи изображения и двоични данни за вграждане ... 97
5.1.1 Експериментално множество от изображения ... 97
5.1.2 Експериментални множества от двоични данни ... 98
5.2 Оценка на модулните методи ... 98
5.2.1 Проверка на устойчивостта срещу JPEG трансформации ... 99
5.2.2 Проверка на качеството на изображенията ... 102
5.2.3 Експериментални резултати на модулния стеганографски метод ... 105
5.2.4 Експериментални резултати на модулния метод за цифрово маркиране ... 106
5.2.5 Оценка на модулните методи ... 107
5.3 Оценка на ефективността на съществуващите подходи и методи за статистически стеганализ ... 112
5.3.1 Целесъобразност на формалното изследване и проверка на системите за стеганография и цифрово маркиране ... 114
5.3.2 Методи за стеганализ, работещи директно с пикселите на изображението ... 116
5.3.3 Методи за стеганализ, работещи в DCT спектралната област ... 118

5.3.4 Ефективност на методите за стеганализ по отношение на модулните методи за криене на данни 121
5.4 Реализация и оценка на ефективността на два принципни подхода за стеганализ 123
5.4.1 Изследване на корелации в изображението 124
5.4.2 Прилагане на филтри върху анализираното изображение 128
5.5 Заключение 130
Глава 6 Приложение на методите в Интернет-базирани сценарии 132
6.1 Предотвратяване на фишинг на банкови портали 132
6.1.1 Фишинг – описание на проблема 133
6.1.2 Недостатъци на традиционните технологии 134
6.1.3 Предотвратяване на фишинг чрез криене на данни 135
6.1.4 Мрежова услуга за сертифициране 136
6.1.5 Мрежов интерфейс на услугата за сертифициране 139
6.1.6 Вграждана информация за проверка на автентичността 142
6.1.7 Интеграция на услугата с банковия Интернет портал 145
6.1.8 Интеграция на услугата с клиентски Интернет браузъри 148
6.2 Защита на мултимедийната интелектуална собственост на информационни агенции 155
6.2.1 Недостатъци на традиционните подходи 155
6.2.2 Подобряване на защитата чрез криене на данни 157
6.2.3 Откриване на нарушения на авторските права 157
6.3 Подобряване на законосъобразното използване на мултимедийно съдържание в Интернет-базирани общества 159
6.3.1 Използване на методи за цифрово маркиране 160
6.3.2 Приложни сценарии 161
6.4 Заключение 164
Глава 7 Заключение – основни резултати 166
7.1 Списък с авторски публикации по дисертацията 166
7.2 Участие в проекти 167
7.3 Постигнати резултати 167
7.3.1 Модулност и адаптивност 167

7.3.2 Подобрена устойчивост срещу JPEG трансформации и стеганализ 168
7.3.3 Услуга за сертифициране 169
7.4 Основни приноси 170
7.4.1 Научно-приложни приноси 170
7.4.2 Приложни приноси 170
7.5 Насоки за бъдеща работа 171
Благодарности 173
Декларация за оригиналност на резултатите 174
Библиография 175
Приложение 1 186
Приложение 2 195

Списък на фигурите

Фиг. 1: Криене на криптирани данни в мултимедия 4
Фиг. 2: Области на приложение на криенето на данни в мултимедия 6
Фиг. 3: Класификация на методите за криене на данни 17
Фиг. 4: Steganos Privacy Suite 22
Фиг. 5: Invisible Secrets 23
Фиг. 6: Вграден модул (англ. plug-in) за Photoshop на Digimarc 24
Фиг. 7: Потребителски интерфейс на Photopatrol (в браузър Internet Explorer) 25
Фиг. 8: Графичен потребителски интерфейс на SignMyImage 26
Фиг. 9: Модулен подход за криене на данни в мултимедия за целите на мрежови приложения 29
Фиг. 10: Базов модул 30
Фиг. 11: Приложно-специфичен модул 31
Фиг. 12: Вграждане на скрити данни 32
Фиг. 13: Декодиране на данни: 33
Фиг. 14: Кодиране на изображение в JPEG - общ поглед 35
Фиг. 15: Декодиране на JPEG изображение - общ поглед 38
Фиг. 16: Базов модул – етап на подготовка (*PrepareToEncode*) 40
Фиг. 17: DCT коефициенти - подбор 40
Фиг. 18: Базов модул – етап на завършване на кодирането (*FinishEncode*) 41
Фиг. 19: Базов модул – декодиране (*Decode*) 45
Фиг. 20: Модулен стеганографски метод – направляващо файлово описание 47
Фиг. 21: Кодове за корекция на грешки 48
Фиг. 22: Генератор на псевдослучайни числа CMWC_4096, предложен от Марсалия 51
Фиг. 23: Псевдослучаен ред на блоковете от изображението 51
Фиг. 24: Стеганографски метод - кодиране 52
Фиг. 25: Стеганографски метод - декодиране 53
Фиг. 26: Модулен метод за цифрово маркиране – нови свойства 54
Фиг. 27: Проверка на изображение, съдържащо неправомерно променени JPEG блокове 55
Фиг. 28: Метод за цифрово маркиране – направляващо описание на цифровия воден знак 56
Фиг. 29: Макроблокове – структура, размер и капацитет 57
Фиг. 30: Макроблокове в изображение 59
Фиг. 31: Откриване на промени в изображението 60
Фиг. 32: Метод за цифрово маркиране - кодиране 61
Фиг. 33: Метод за цифрово маркиране - декодиране 62

Фиг. 34: Базов модул – етап на подготовка (*PrepareToEncode*) 64
Фиг. 35: Разпределение на битовете в даден цветови канал 65
Фиг. 36: Базов модул – етап на завършване на кодирането (*FinishEncode*) 66
Фиг. 37: Кодиране на данните в най-младшите битове на даден цветови канал 67
Фиг. 38: Кодиране на RGB-цветови канали в 24-битово или 32-битово цяло число 68
Фиг. 39: Базов модул – декодиране (*Decode*) 68
Фиг. 40: Програмна архитектура – общ поглед 73
Фиг. 41: Примерна част от кода на класа „LSBHiderEngine“, реализиран чрез VB.NET 78
Фиг. 42: Примерна част от кода на класа „DCTHiderEngine”, реализиран чрез C++ 79
Фиг. 43: Примерна част от кода на класа „StegoHider”, реализиран чрез VB.NET 81
Фиг. 44: Примерна част от кода на координатора, реализиран чрез VB.NET 82
Фиг. 45: Графичен потребителски интерфейс 85
Фиг. 46: Графичен потребителски интерфейс – програмни менюта 85
Фиг. 47: Диалогов прозорец *Options* 86
Фиг. 48: Графичен потребителски интерфейс – групови обработки 87
Фиг. 49: Графичен потребителски интерфейс – инструменти за статистически анализ 88
Фиг. 50: Графичен потребителски интерфейс – инструменти за сравнение 88
Фиг. 51: Резултат от прилагането на усредняващ филтър с маска 9×9 89
Фиг. 52: Резултат от хистограмен анализ в цветовото пространство RGB 90
Фиг. 53: Примерна SOAP заявка 91
Фиг. 54: Примерен отговор на SOAP заявка 92
Фиг. 55: Примерна HTTP POST заявка 93
Фиг. 56: Примерен отговор на HTTP POST заявка 93
Фиг. 57: Критерии за оценка на модулните методи 96
Фиг. 58: Примерни изображения, принадлежащи на експерименталното множество 98
Фиг. 59: Проверка на устойчивостта срещу JPEG трансформации 100
Фиг. 60: Проверка на качеството на изображенията 104
Фиг. 61: Модулни методи за криене на данни – резултати от оценката 111
Фиг. 62: Информация за криптографски сертификат на Интернет портал във Firefox 134
Фиг. 63: Приложение на модулните методи като мрежова услуга за сертифициране 137
Фиг. 64: Примерна SOAP заявка за вграждане на сигнатура в изображение 140
Фиг. 65: Примерен SOAP отговор на заявката от Фиг. 64 141
Фиг. 66: Примерна SOAP заявка за проверка на сигнатура, вградена в изображение 141
Фиг. 67: Примерен SOAP отговор на заявката от Фиг. 66 142
Фиг. 68: Формат на вградената информация за проверка на автентичността - XSD схема 143
Фиг. 69: Информация за проверка на автентичността - примерен XML документ 144
Фиг. 70: Ръчно вграждане на сигнатури в изображения - GUI 146
Фиг. 71: PHP скрипт – стартиране на метода *hideCopyrightInformationByImage* 147

Фиг. 72: Графичен интерфейс на потребителски скрипт преди проверка на сигнатурите 149
Фиг. 73: Графичен интерфейс на потребителски скрипт след проверка на сигнатурите 151
Фиг. 74: Потребителски скрипт – стартиране на мрежов метод 152
Фиг. 75: Идентификация на притежателя на авторските права чрез видим текст 156
Фиг. 76: Откриване на нарушения на авторските права 158
Фиг. 77: Микроплащане на мултимедийни творби 162
Фиг. 78: Откриване на нарушения на авторското право 163

Списък на таблиците

Табл. 1: DCT-базирани методи за криене на данни в мултимедия – обща оценка 21
Табл. 2: Продукти и услуги за криене на данни в мултимедия – обща оценка 27
Табл. 3: Модулен стеганографски метод – постигнати резултати 105
Табл. 4: Модулен метод за цифрово маркиране – постигнати резултати 106
Табл. 5: Характеристики на съществуващите методи за криене на данни 108
Табл. 6: Характеристики на съществуващите продукти и услуги за криене на данни 109
Табл. 7: Резултати от анализа, проведен със *Stegdetect* 122

Списък на съкращенията

.NET	.NET Framework (програмна платформа, създадена от Майкрософт)
API	Application Programming Interface (приложен програмен интерфейс)
ASP	Active Server Pages (технология на Майкрософт)
BMP	Bitmap (формат на изображение)
CA	Certification Authority (сертифицираща инстанция)
CMWC	Complimentary-multiply-with-carry (вид генератор на псевдослучайни числа)
dB	Decibel (мерна единица)
DCT	Discrete Cosine Transform (дискретна косинусова трансформация)
DFT	Discrete Fourier Transform (дискретна Фурие трансформация)
DNS	Domain Name System (система за имената на домейните)
DOM	Document Object Model (обектен модел на документа)
DRM	Digital Rights Management (управление на правата за ползване на цифрово съдържание)
DWT	Discrete Wavelet Transform (дискретна уейвлитна трансформация)
DW	Digital Watermarking (цифрово маркиране)
ECC	Error-Correcting Codes (кодове за корекция на грешки)
EXIF	Exchangeable Image File Format (вид формат за съхраняване на мета-данни за изображения)
FTP	File Transfer Protocol (вид протокол за обмен на файлове в Интернет)
GIF	Graphics Interchange Format (формат за изображения)
GUI	Graphical User Interface (графичен потребителски интерфейс)
HTML	Hypertext Markup Language (основен декларативен програмен език за създаване на Интернет страници)
HTTP	Hypertext Transfer Protocol (основен протокол за обмен на данни в Интернет)
ID	Identification (идентификационен номер или низ от знаци)
IDCT	Inverse Discrete Cosine Transform (обратна дискретна косинусова трансформация)
I/O	Input / Output (вход/изход)
IIS	Internet Information Server (широко разпространен сървър за мрежови приложения, създаден от Майкрософт)
IP	Internet Protocol (Интернет протокол)
JFIF	JPEG File Interchange Format (файлов формат за изображения, широко разпространен в Интернет)
JPEG	Joint Photographic Experts Group (групата, отговорна за стандартизацията на едноименния формат за изображения)
LSB	Least-significant Bit (най-младши бит)

MPEG	Moving Picture Experts Group (групата, отговорна за стандартизацията на едноименния формат за филми)
MSE	Mean Squared Error (средна квадратична грешка)
OOP	Object-Oriented Programming (обектно-ориентирано програмиране)
PIN	Personal Identification Number (персонален идентификационен номер)
PNG	Portable Network Graphics (формат за изображения)
PSNR	Peak Signal-to-Noise Ratio (върхово съотношение сигнал-шум)
QIM	Quantization Index Modulation (модулация на индекса на квантизация – вид техника за кодиране на информация)
RAD	Rapid Application Development (бърза разработка на приложения)
RGB	Red-green-blue (цветово пространство)
RSA	Rivest-Shamir-Adleman (криптографски алгоритъм)
SaaS	Software-as-a-Service (програмно осигуряване, предлагано под формата на мрежова услуга)
SOAP	Simple Object Access Protocol (вид протокол за обмен на данни в Интернет)
SSL	Secure Socket Layer (технология за криптиране на комуникацията в Интернет)
TAN	Transaction Authentication Number (потвърдителен/оторизационен номер на транзакция в банковото дело)
TCP	Transmission Control Protocol (основен протокол за обмен на данни в Интернет)
TIFF	Tagged Image File Format (формат за изображения)
UDP	User Datagram Protocol (протокол за обмен на данни в Интернет)
UML	Unified Modelling Language (единен език за моделиране)
URL	Uniform Resource Locator (вид стандартизиран идентификатор на информационни ресурси в Интернет)
XSD	XML Schema Definition (стандартизиран формат за задаване и проверка на структурата на XML документ)
XML	Extensible Markup Language (декларативен програмен език за съхраняване на данни)
YCbCr	Luminance-Chrominance-Blue-Chrominance-Red (цветово пространство)
ИТ	Информационни Технологии

Списък на някои важни термини и понятия

Воден знак	В контекста на цифровото маркиране – скрита в цифрова мултимедия информация, която се отнася до тази мултимедия и идентифицира нейния автор, купувач и законен потребител, гарантира нейната автентичност и т.н. (англ. watermark)
Възприемано качество	Субективно определеното качество на дадена мултимедия, оценено от човек и зависимо от начина на действие на човешките сетива (англ. perceptual quality)
Крехък воден знак	Воден знак, който престава да съществува при промяна на цифровата мултимедия, в която е вграден (англ. fragile watermark)
Направляващо описание на файл	Характерна спомагателна информация, съхранявана най-често в началото на двоичен файл и предшестваща основното съдържание на файла (англ. file header)
Носеща среда / Носещо изображение	Цифрова мултимедия / цифрово изображение, в които методите за криене на данни могат да вграждат информация (англ. host medium / host image)
Сляп метод	Метод за криене на данни в мултимедия, при който декодерът на данните не се нуждае от оригиналното изображение или статистическа информация, която го описва, за да декодира вградената информация (англ. blind method)
Сигнатура	Сравнително къса поредица от двоични данни (8-4096 байта), чието съдържание се използва за проверка на автентичността, целостта и др. свойства на по-голям блок цифрова информация (англ. signature)
Статистически стеганализ	Изследване на статистическите свойства на цифровите изображения, за да се установи дали в тях се съдържат вградени данни (англ. statistical steganalysis).
Цифрова мултимедия	Комбинация от текст, цифрови изображения, цифрово аудио, цифрово видео или анимация (англ. digital multimedia)

Глава 1

Увод

Криене на данни (англ. data hiding) е модерното име на класическа наука, водеща началото си от древна Гърция [1]. Първоначално тя е известна под името стеганография – съставено от гръцките думи “steganos“ (“покрит, таен”) и “graphia” (“писане”), което за пръв път е документирано от немския учен Йохан Тритемий (нем. Johannes Trithemius) в неговия научен труд “Steganographie” [2]. Стеганографията по това време е наука, фокусирана върху теоретичните методи и практическите приложения на скриването на тайна информация в различни видове носещи среди. Целта е скритата информация да бъде невидима за неинформирания потребител и да не се влияе от нормалните обработки на носещата среда, в която е вградена. Прочитането на информацията е възможно само посредством прилагането на специализирани трансформации. Класически примери за приложението на стеганографията са водните знаци и защитните метални нишки, скрити в банкнотите [3], невидимите мастила [4] и микро-печатът [5], който използва шрифтове с изключително малък размер (< 0.5 мм.), възприемани като тънка линия от невъоръженото човешко око.

1.1 Актуалност на проблема

Прогресът на модерните комуникационни технологии и цифровата мултимедия са причината за възобновяването на интереса към стеганографията и формирането ѝ като модерна наука, носеща вече по-общото наименование „криене на данни в мултимедия“ (англ. multimedia data hiding) [6]. С течение на времето се оформят две основни научноизследователски направления: първото направление наследява древното име на тази наука – стеганография[1], а второто направление е наречено цифрово маркиране (англ. digital watermarking) [6], [7].

Модерната стеганография изучава кодирането и откриването на тайни съобщения, предавани през цифрови комуникационни канали и платформи. Стеганографските методи скриват наличието на произволни цифрови съобщения посредством тяхното кодиране като обичайна част от съдържанието на друго цифрово мултимедийно съдържание. По този начин те правят откриването на тези съобщения от потенциални неоторизирани трети лица много трудно [8]. В последните няколко години значението на

[1] От сега нататък, когато се използва понятието стеганография, винаги ще имаме предвид неговото съвременно значение като научноизследователско направление на криенето на данни в мултимедия.

стеганографията се преосмисли от редица правителства с оглед на рисковете от обмен на информация между терористични организации с цел терористични атаки [9], [10].

Цифровото маркиране, за разлика от стеганографията, има за основна цел да подобри защитата на интелектуалната собственост и да позволи проверка на автентичността на цифровите медии [11], [12], [13]. По подобие на стеганографските методи, методите за цифрово маркиране крият информация като неразделна част от съдържанието на съответната цифрова мултимедия. Разликата се състои в целта, постигана от скритата информация – тя не е произволно съобщение, а характеризира цифровата мултимедия, в която е скрита, като идентифицира нейния автор, купувач или потвърждава автентичността на мултимедийното съдържание. Методите за цифрово маркиране помагат при проследяване на бързото и евтино разпространение на цифровата мултимедия в глобалната мрежа. Те предоставят нови начини за адекватна защита на притежателите на авторски права в процеса на разпространение на интелектуалната собственост [14].

Методите за криене на данни в мултимедия се разработват специално за вграждане на информация като неразделима част от самото мултимедийно съдържание. Това ги прави изключително подходящи за подобряване на защитата на мултимедийните файлове в глобалната мрежа и по-успешното съблюдаване на авторските права. Тези методи се прилагат върху различни видове мултимедия – текст, цифрови изображения, цифрово аудио и видео, анимация и др., като могат да бъдат свързани с различни области на приложение [15], [16]. Научноизследователската дейност, описана в тази дисертация, е насочена основно към цифровите изображения и цифровото видео като носещи среди, към които има особен интерес. Основната причина за този акцент е огромната популярност на изображенията, а в последните години и на кратките видео клипове, в глобалната мрежа. Те се използват от почти всеки модерен Интернет портал, за да подобрят представянето на съдържанието му пред крайните потребители. Трябва да се подчертае, че резултатите, постигнати от методите за криене на данни в изображения, образуват основата, на която се базират методите за криене на данни във видео, тъй като повечето видео формати кодират своите т. нар. „ключови кадри” (англ. key frames) във формати за кодиране на цифрови изображения [17], [18], [19].

В следващите раздели се описват предимствата на технологията за криене на данни в мултимедия като средство за допълване и подобряване на съвременните широко използвани криптографски подходи. Дискутира се значението и се мотивира употребата на технологията за подобряване на сигурността в Интернет-базирани сценарии. Описват се някои популярни области на приложение, касаещи глобалната мрежа.

1.1.1 Предимства на криенето на данни пред криптографските подходи

Основните цели на двете научноизследователски направления на криенето на данни в мултимедия – стеганография (тайна комуникация) и цифрово маркиране (защита на

интелектуалната собственост и проверка на автентичността на цифровата мултимедия) – попадат традиционно в полето на приложение на криптографията. Криптографските подходи могат да направят информацията, предавана по комуникационни канали, разбираема само за избрания получател и недешифрируема за неоторизирани трети лица. С тяхна помощ може също така да се гарантира целостта и да се идентифицира произхода на цифровата мултимедия (за кратък обзор на криптографията вж. [20]). Тези методи имат солидна математическа основа (дискретна математика, линейна алгебра, теория на числата и други клонове от математиката) [21]. Като примери могат да се посочат т.нар. Secure Socket Layer (SSL) комуникация с редица Интернет портали или т.нар. Digital Rights Management (DRM) технологии за защита на интелектуалната собственост [22], [23].

По отношение на мултимедийното съдържание криенето на данни може да предложи специализирани решения, които имат следните ясно дефинирани технологични и правни предимства пред общоприложимите криптографски подходи:

- **Гъвкавост**: методите за криене на данни могат да бъдат разработени така, че скритите данни да са устойчиви на често прилагани трансформации на цифровата мултимедия. Такива трансформации са напр. загубна компресия, изрязване, скалиране, завъртане, смяна на цветовите пространства, прилагане на филтри, определяне на контури, корекция на яркост и контраст и др. Такива трансформации се извършват често в зависимост от областта на приложение на изображенията (класификация, разпознаване на обекти, цифрово предпечатно оформление, архивиране, и др. Това е особено важно за методите на цифрово маркиране, където защитената интелектуална собственост (под формата на цифрова мултимедия) често претърпява такива умишлени или неумишлени промени на съдържанието. За разлика от методите за криене на данни, DRM-схемите за защита на авторските права не позволяват каквито и да било трансформации на защитеното съдържание и по този начин ненужно ограничават легалната употреба на цифровата мултимедия.
- **Прозрачност**: приложението на технологии за криене на данни в мултимедия обикновено е незабележимо за потребителите. Използването на такива технологии може да бъде открито само с помощта на специализирано програмно осигуряване, създадено специално за тази цел. За разлика от тях, използването на криптографски подходи е лесно забележимо от потребителите.
- **Независимост**: методите за криене на данни вграждат информация директно в мултимедийното съдържание. При тях не се налага промяна във формата за запомняне и предаване на мултимедията поради наличие на допълнителни сигнатури, криптографски хеш-стойности и др., което често се налага, ако се използват подходи от криптографията.
- **Надеждност**: стеганографската информация, както и вградените цифрови маркери са интегрална част от самата мултимедия и не могат да бъдат надеждно премахнати

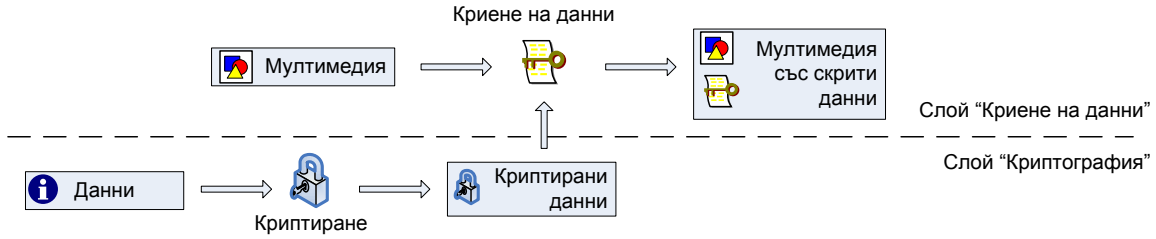
(без информация относно използваните методи) от неоторизирани трети лица, без да се загуби значителна част от мултимедийното съдържание, което води съответно до рязко влошаване на неговото качество. От друга страна, криптографската информация може лесно да се премахне, след като цифровата мултимедия се декриптира (напр. за да бъде показана на краен потребител).

- **Липса на правна регулация**: на този етап стеганографията и цифровото маркиране не подлежат на правно (законово) регулиране, характерно за традиционната криптография в някои страни [24].

За да бъдат използвани тези предимства, авторът на мултимедийното съдържание трябва да пожертва контрола на достъп до съдържанието, реализиран от повечето DRM-схеми. Стеганографията и цифровото маркиране не могат надеждно да предотвратят достъпа на крайния потребител до мултимедийната носеща среда и допускат, че всички потребители имат право на четене (гледане, слушане) на мултимедийното съдържание.

1.1.2 Криене на данни за целите на Интернет-базирани сценарии

Една от важните цели при разработката на Интернет е предоставянето на универсално достъпно средство за комуникация между хората, търговия и споделяне на знание [25]. Броят на потребителите на глобалната мрежа възлиза на почти 1.6 милиарда през 2008г. [26], което подчертава нейното значение като среда за обмен на информация. Тим О'Райли обобщава: "Мрежовите ефекти от приносите на потребителите са ключът към доминирането на пазара в ерата на „Уеб 2.0" [27], [28], [29]. Създаването на такива глобални мрежови ефекти зависи до голяма степен от свободния поток на информацията и свободния достъп до знанието.



Фиг. 1: Криене на криптирани данни в мултимедия

Въпреки тази важна тенденция към свобода на разпространението на информация, технологии за сигурност, осигуряващи поверителна комуникация [30], [31] или защита на интелектуалната собственост [32], [33], [34] са необходими за редица приложения. Традиционните криптографски подходи са базирани на цифрови сигнатури и публични/частни криптографски ключове, които се прилагат напр. за криптиране на комуникационни канали, цифров подпис на мултимедийни файлове и DRM-технологии за

предотвратяване на нежелан достъп и копиране на мултимедийно съдържание. Те са разпространен метод за защита на интелектуалната собственост, но имат редица слаби места:

1. Криптират цялото цифрово съдържание, което води до няколко важни недостатъка:
   - Ограничава свободния поток на информация;
   - Обявява наличието на защитени данни пред потенциални неоторизирани трети лица;
   - Не позволява разграничение между извършването на операции върху съдържанието, които имат семантично значение и са важни за приложението, и извършването на такива операции, които са необходими за протичането комуникационния процес (напр. прилагане на загубна компресия).
2. Прикрепват криптирана сигнатура към цифровата мултимедия, което се характеризира със следния важен недостатък – сигнатурата не е естествена част от цифровото мултимедийно съдържание и може да бъде лесно открита, загубена (напр. при промяна на файловия формат) или умишлено премахната.

Криенето на данни в мултимедия може да допълни и подобри съществуващите криптографски решения [35], [23], [16], [36] и да смекчи гореизброените недостатъци посредством вграждането на криптираните данни и/или сигнатури като интегрална част от самото цифрово мултимедийно съдържание (Фиг. 1). По този начин криенето на данни в мултимедия запазва възможността за лесен достъп до мултимедийното съдържание и осигурява труден за откриване и отстраняване механизъм за предаване на тайни съобщения, за съхраняване на поверителна информация, както и за проследяване на авторите, цифровите копия и законните притежатели на мултимедийно съдържание.

Мрежови услуги за криене на данни в мултимедия могат да бъдат предоставени и използвани в допълнение и като надграждане на криптографските услуги, използвани в дадена компания. Като входна оперативна информация тези услуги получават цифровата мултимедия, избрания алгоритъм за криене на данни и поверителните криптирани данни, които трябва да се вградят в мултимедията (в случая на кодиране). Като изходна информация тези услуги предоставят цифрова мултимедия, почти идентична с оригинала, в която са скрити поверителните криптирани данни. Тъй като входното и изходното мултимедийно съдържание са подобни, то интегрирането на новите услуги във вече съществуващи ИТ-инфраструктури и бизнес процеси не е трудно. За разлика от криптографските услуги, те осигуряват съвместимост със системи, които не вземат предвид проблеми, свързани със сигурността.

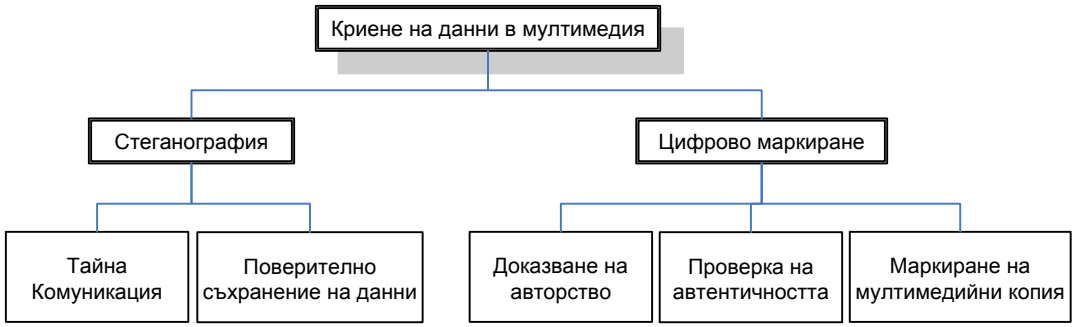
1.2 Области на приложение

Представените в този раздел области на приложение разглеждат някои типични приложни сценарии на технологиите за криене на данни в мултимедия и показват как

тези технологии могат да работят в близост до крайните потребители и да се използват за тяхна защита в Интернет.

По отношение на Интернет-базираните сценарии и приложения, съществуват пет основни приложни области на технологиите за криене на данни в мултимедия [6], [9], [37], [38], [14], които са показани на Фиг. 2. Първите две приложни области попадат в рамките на научноизследователското направление стеганография, а останалите три са предмет на изследване на направлението цифрово маркиране.

Първата област на приложение е т.нар. *тайна комуникация*. Тя разглежда предаването на поверителни съобщения през глобалната мрежа с помощта на цифрова мултимедия. Целта на тайната комуникация е подобна на предаването на криптирани данни – гарантиране на поверителността на личната комуникация между двама участници. Допълнителното предимство на стеганографските техники в тази област е, че те скриват самото наличие на комуникационен процес чрез вграждането на съобщенията в произволно мултимедийно съдържание. Двамата участници в комуникацията използват мултимедийни данни, чието съдържание е видимо за произволни трети лица (например снимки в Интернет блог) и не е поверително. Наличието и смисълът на тайните съобщения, вградени в мултимедията остават неоткриваеми от неупълномощени трети лица.



Фиг. 2: Области на приложение на криенето на данни в мултимедия

Втората приложна област засяга *невидимото съхранение* на поверителни данни като номера на кредитни карти, банкови сметки, пароли и др. [39]. Сигурността на съхранението на тези данни може да бъде повишена с помощта на технологиите за вграждане на данни в мултимедия. Те могат да направят криптираните лични данни невидими, като ги вградят в мултимедия. По този начин се повишава нивото на защита - добавя се допълнителен защитен слой (допълващ криптографската защита) между конфиденциалните данни и потенциалните хакери.

Третата област на приложение засяга защитата на интелектуалната собственост и позволява *доказване на авторски права* върху дадена мултимедия. Технологиите за криене на данни могат да вградят информация за истинския автор директно в мултимедийното съдържание. По този начин, когато възникнат спорове, касаещи авторските права върху произведението, законният автор може да докаже и защити своите права.

Четвъртата приложна област – *проверка на автентичността* – се отнася до защитата на мултимедийно съдържание срещу неправомерни промени. Посредством технологиите за криене на данни отделните региони от мултимедийното съдържание, които са били неправомерно променени, могат да бъдат разпознати и в зависимост от използвания метод, оригиналното съдържание може даже да се възстанови до определена степен [40].

Петата област на приложение – *маркиране на мултимедийни копия* – също засяга защитата на авторски права. По подобие на приложенията за доказването на авторство, методите за криене на данни вграждат информация, която идентифицира купувача или законния ползвател на мултимедийното съдържание. По този начин, ако се предприемат действия, нарушаващи правата на автора – като например нелегалното разпространение на мултимедийното съдържание в Интернет – нарушителят може да се разпознае посредством скритата информация, вградена в нелегално разпространените копия.

Представените по-горе пет приложни области описват само най-значимите и най-разпространените приложения на криенето на данни в мултимедия, които имат отношение към Интернет. Глобалната мрежа е изключително динамична среда и този списък от приложения лесно може да се разшири с нови области на приложение при всеки по-важен етап на развитие и напредък на модерните информационни и комуникационни технологии.

За да бъдат илюстрирани някои от приложенията на криенето на данни в мултимедия (по-детайлна дискусия е дадена в [6]), в следващите раздели се разглеждат един пример, описващ приложение на стеганографията, и три примера, описващи приложения на цифровото маркиране. Тези примери показват практическата полза от двете технологии в сравнение с традиционните криптографски подходи, както и възможностите за гъвкавата им интеграция в модерните Интернет-базирани бизнес процеси, включващи мрежова комуникация и разпространение на интелектуална собственост. Допълнителни примери за приложение са разгледани в Глава 6.

1.2.1 Тайна комуникация

Този пример дискутира сливането на две акционерни дружества, чиито акции се търгуват на фондовите борси. Подобно сливане обикновено има голямо въздействие върху финансовите пазари и двете компании се опитват да го запазят в тайна. Те трябва да комуникират една с друга, за да договорят условията на сливането, но дори самото наличие на повишена комуникация между тях може да послужи като индикатор за предстоящо сливане на компаниите, от което финансовите спекуланти да се възползват. Интернет е несигурна комуникационна среда и двете компании не са сигурни кой би могъл да наблюдава техните комуникации и с каква цел.

Традиционните криптографски подходи не са полезни в тази ситуация, тъй като те не могат да скрият факта, че значителна по обем информация се обменя между компа-

ниите. На базата на повишения обем на комуникацията и текущото икономическо положение, предстоящото сливане лесно може да бъде предвидено. Стеганографията може да скрие самото наличие на комуникационния процес, като вгражда и предава кореспонденцията като част от ненатрапващи се графични изображения или снимки. Двете компании могат да използват своите корпоративни и публични форуми, общодостъпни услуги за съхранение и споделяне на изображения като Фликр (англ. Flickr), платформи за обмен на знание като Уикипедия (англ. Wikipedia) или комуникационни и социални платформи от типа на Фейсбук (англ. Facebook) за обмен на мултимедийните файлове, съдържащи вградената поверителна кореспонденция.

Кодирането (вграждането) и декодирането (разчитането) на скритите съобщения може да се реализира като разширение на корпоративната информационна инфраструктура чрез добавяне на допълнителни мрежови услуги, чиято задача е следенето и обработването на вградените съобщения в мултимедийни файлове.

1.2.2 Доказване на авторство

Този пример се отнася до новинарската агенция *A*, която поддържа и обновява информационен портал в Интернет. Конкурентната новинарска агенция *B* сваля снимките, придружаващи новинарските статии от Интернет портала на *A*, и ги използва в изданията на своя собствен Интернет вестник. *A* би желала да докаже, че тя е законният собственик на снимките и да изобличи действията на *B* като нарушения на авторското право. Тъй като цифровите данни, описващи снимките могат лесно да бъдат променени, шансът за успех на *A* не е голям.

Традиционните криптографски DRM подходи не могат да предотвратят незаконното присвояване на авторско съдържание или да помогнат при доказването на авторски права върху снимките, ако те вече са направени публично достъпни в Интернет. Цифровото маркиране, от друга страна, може да вгради информация, идентифицираща законния автор директно в съдържанието на снимките. Новинарската агенция *A* може да се абонира за такава услуга за цифрово маркиране, предоставена от компанията *C*, която е специалист в областта на цифровото маркиране. *A* трябва да направи малки промени в конфигурацията на своя Интернет сървър. Когато някой от авторите на статии качи снимка на Интернет сървъра, която е част от новинарска статия, тя автоматично се изпраща към услугата, предоставена от *C*, за да се подложи на цифрово маркиране, преди да се направи публично достъпна в новинарския информационен портал.

Поради предимствата *гъвкавост*, *независимост* и *надеждност*, дискутирани в раздел 1.1.1, данните, скрити от методите за цифрово маркиране най-вероятно ще присъстват в копията на снимките, публикувани в Интернет вестника на новинарската агенция *B*. Поради предимството *прозрачност* новинарската агенция *B* най-вероятно не подозира за съществуването на скрити водни знаци в снимките. Когато въпросът за нарушените авторски права се повдигне, с помощта на услугите на *C* и вградените водни

знаци може да се потвърди, че новинарската агенция *A* е истинският носител на авторските права върху снимките.

1.2.3 Проверка на автентичността на мултимедия

Да разгледаме пример, в който водеща машиностроителна компания, е внедрила мрежа от камери за наблюдение в свое предприятие. Камерите са свързани към Интернет и автоматично архивират всички изображения на централен сървър. Един ден в една от производствените сгради е установена липсата на нов прототип. Записът от камерите показва как нарушители са влезли неправомерно в сградата. Заснето е също така лицето на един от работниците, който твърди, че е невинен.

За да се гарантира, че записът от камерите е надеждно доказателство, компанията се нуждае от средство за проверка, че записът е оригинален и не е бил променян от момента на неговото създаване. Това би могло да се направи с помощта на криптографски подходи [41], но цифровото маркиране предлага допълнително възможността за откриване на точните региони от записа, които са били неправомерно променени. Това се постига чрез вграждането (в записите от камерите) на специални сигнатури, наречени крехки водни знаци, които се разрушават в случай на промяна на изображението, в което са вградени. В допълнение към откриването на променени региони, съществуват специализирани методи за цифрово маркиране, които позволяват частичното възстановяване на оригиналното съдържание [40].

В този пример методите за цифрово маркиране могат да потвърдят, че регионът от записа на камерите, съдържащ лицето на работника, е бил неправомерно променен. Възможно е дори да се възстанови лицето на реалния извършител и това да доведе до заключението, че работникът е невинен. Останалата информация относно начина, по който е била извършена кражбата, е автентична и може да се използва в хода на разследването.

1.2.4 Маркиране на мултимедийни копия

Този пример е вдъхновен от истински случай, описан в Интернет [42].

Художник създава цифрови картини (интелектуална собственост). Той има агент, който се грижи за показването на картините в (Интернет) галерии и тяхната продажба на индивидуални клиенти. Художникът записва своите нови творби на сървъра на агента си и получава от него известия за хода на продажбите. Един ден художникът намира част от своите произведения свободно достъпни в Интернет, въпреки клаузи в договора с галериите и крайните клиенти, които забраняват разпространението на творбите. Художникът би искал да предотврати подобни инциденти в бъдеще, както и да съди нарушителя за щети, изисквайки материална компенсация.

Традиционните криптографски DRM подходи не могат да помогнат в такава ситуация. Те не могат да предотвратят надеждно разпространението на творбите или да идентифицират нарушителя. Всяка оторизирана галерия или краен купувач могат да разгледат картина, криптирана чрез DRM-техники, което означава, че те могат да репродуцират творбата и премахнат криптографската защита. Това им дава свободата да разпространяват картината в Интернет, без да оставят следи, водещи до разкриването на тяхната самоличност.

Технологиите за цифрово маркиране предлагат ефективно решение. Те не могат да предотвратят разпространението на картините, но предлагат надеждно средство за идентификация на нарушителя на договора за неразпространение чрез маркирането на всяко едно продадено/предоставено копие от творбата с информация, идентифицираща първоначалния купувач/ползвател.

За да използва защита на базата на цифрово маркиране за продаваните творби, агентът на художника се абонира за услуга за маркиране на мултимедийни копия, предоставяна от компанията *C*, специализирана в областта на цифровото маркиране. Услугата приема като входни данни оригиналното изображение и низ от знаци, идентифициращ купувача. Като изходни данни, услугата предоставя маркираното копие на оригиналната творба. За да използва услугата, агентът на художника трябва да направи лека промяна на своите информационни системи. Преди да бъдат изпратени на галерии и крайни купувачи, творбите първо се препращат към услугата на *C*, която ги маркира с данни, идентифициращи съответния получател. Маркираните картини, върнати като резултат от услугата на *C*, се изпращат от системите на агента на художника на крайните ползватели.

Ако нелегално копие на някоя от творбите бъде открито в Интернет, поради предимствата *гъвкавост*, *независимост* и *надеждност* (вж. раздел 1.1.1) данните, скрити от методите за цифрово маркиране и идентифициращи първоначалния купувач, най-вероятно ще присъстват в мултимедийното съдържание. Тези данни могат да бъдат прочетени от допълнителна услуга, предоставена от *C*, и нарушителят на договора за неразпространение може да бъде съден за нанесени щети.

1.3 Основни научни области от значение за развитието на криене на данни в мултимедия

За реализирането на ефективно криене на данни в мултимедия е необходимо доброто познаване на редица научни и приложни области като обработка на сигнали в честотната област, компресия на мултимедийно съдържание, обработка на изображения, аудио и видео, криптография, теория на числата, теория на кодирането, софтуерно инженерство и др.

В тази дисертация фокусът лежи върху криене на данни в компресирани (JPEG) и некомпресирани (PNG, BMP) изображения. Неделима част от криенето на данни в

JPEG изображения е прилагането в частност на дискретната косинусова трансформация (англ. discrete cosine transform, DCT), характерна за формата на компресия JPEG. Както изображенията, така и вгражданите данни се подлагат на компресия чрез аритметично кодиране, кодиране с дължини на сериите (англ. run-length coding), Хъфман кодиране и др. Обработката на изображения включва прилагането на различни видове филтри и построяването на хистограми, използвани често при алгоритмите за анализ/стеганализ на изображенията.

Вгражданите данни често се криптират с различни криптографски методи или представляват криптографски сигнатури, използвани за защита на съдържанието. Потребителски дефинирани и криптографски преобразувани пароли в комбинация с генератори на псевдослучайни числа се използват за определяне на реда на криене на данни в изображенията. При подготовката на данните за вграждане се използват кодове за корекция на грешки и техники на кодиране като модулацията на индекса на квантизация.

Софтуерното инженерство намира приложение при изготвянето на архитектурата и реализацията на програмната система, както и при осигуряването на надеждна Интернет-базирана комуникация с други системи. В дисертацията се използва обектно-ориентирания подход за създаване на програмни системи [43], както и редица протоколи и формати за обмен и съхранение на данни, характерни за Интернет.

Интегрирането на резултати от тези научни и приложни области при проектирането и разработката на системи за криене на данни в мултимедия не е тривиална задача.

1.4 Цел и задачи на дисертацията

Целта на дисертацията е разработването на модулен подход и модулни методи за вграждане на цифрова информация в изображения за подобряване сигурността на Интернет-базирани комуникационни платформи. За постигането на тази цел са формулирани следните задачи:

1. Определяне на основните недостатъци на съществуващите методи за криене на данни в мултимедия по отношение на тяхното приложение в Интернет-базирани сценарии.
2. Разработване на модулен подход за криене на данни в мултимедия за целите на мрежовите приложения. Съгласно този подход всеки метод за криене на данни се състои от най-малко два модула – базов модул и приложно-специфичен модул.
3. Разработване на два базови и два приложно-специфични модула, подходящи за приложение в Интернет-базирани сценарии. Използването на модулите в различни комбинации води до създаването на общо четири нови модулни метода за криене на данни в мултимедия.

4. Създаване на архитектура на програмна система и програмно реализиране на новите методи с помощта на т.нар. .NET платформа (.NET Framework) на Майкрософт.
5. Оценяване на реализацията на методите и техните свойства и на качеството на получените в резултат от тяхната работа изображения. Оценката се използва от крайния потребител при избор на подходящ метод в даден приложен сценарий.
6. Анализ на сигурността на модулните методи от гледна точка на статистическия стеганализ.
7. Осигуряване на подходящ достъп до модулните методи за криене на данни чрез предоставянето на подходящи мрежови услуги и формати за обмен на данни. Цели се лесно включване на модулните методи към цялостни програмни решения и инфраструктури за реализиране на практически насочени Интернет-базирани сценарии.

Решаването на горните задачи дава възможност за постигането на ново ниво на сигурност в различни Интернет-базирани сценарии.

1.5 Структура на дисертацията

Настоящата дисертация се състои от общо седем глави, включващи увод и заключение, благодарности, декларация за оригиналност на резултатите, списък на цитираната литература и две приложения.

Глава 1 представлява въведение в областта на криене на данни в мултимедия. Аргументира се използването на методи за криене на данни в приложни сценарии, свързани с Интернет и се дават някои конкретни примери. Описват се основните научни и приложни области, които са от значение при разработката на нови методи и се формулират целта и задачите на дисертацията.

Глава 2 представя някои важни свойства на методите за криене на данни, които са от първостепенно значение при тяхното приложение в глобалната мрежа. Съществуващите научноизследователски методи и програмни продукти и услуги се анализират и оценяват по отношение на всяко от тези свойства.

Глава 3 описва общата концепция и архитектурата на нов модулен подход за проектиране на методи за криене на данни. Представя се накратко графичния формат със загубна компресия JPEG и се предлага нов базов модул, осигуряващ устойчивост на вградените данни срещу JPEG трансформации. Също така се предлагат два нови приложно-специфични модула – един за стеганографски цели и един за целите на цифровото маркиране. Представен е и базов модул, проектиран за използване с некомпресирани изображения и изображения с беззагубна компресия. Комбинирането на разработените модули позволява създаването на общо четири различни модулни метода за криене на данни.

Глава 4 представя архитектурата на реализираната многослойна програмна система. Реализацията се базира на обектно-ориентираната платформа за програмиране Майкрософт .NET. Програмната система позволява изследване на представянето на всеки от модулните методи, както и провеждането на стеганализ. Достъпът на потребителите и външните програмни системи до функционалността на системата се осигурява от графичен потребителски интерфейс и приложен програмен интерфейс за мрежови услуги.

Глава 5 извършва многокритериална оценка на разработените методи и изследва податливостта им на статистически стеганализ с помощта на експериментални множества от изображения и данни за вграждане. Представят се някои от съществуващите методи за стеганализ, показват се техните недостатъци по отношение на модулните методи и се изследва ефективността на два принципни подхода за стеганализ, реализирани като част от новата програмна система.

Глава 6 представя три примерни Интернет-базирани сценария за приложение на новите методи, базирани на нуждите на потребители от различни сектори на Интернет-ориентирания бизнес и Интернет-базираните общества. Разглеждат се предимствата на модулния подход в среда, близка до реалната и ориентирана към проблемите на потребителите. За целите на примерните приложни сценарии се проектира и реализира мрежова услуга за сертифициране чрез криене на данни.

Глава 7 обобщава постигнатите резултати и представя най-важните приноси на дисертацията. Описват се предимствата на модулния подход и модулните методи за криене на данни пред традиционните методи за криене на данни с оглед на тяхното приложение в глобалната мрежа. Разглеждат се и някои насоки за бъдещо развитие на разработените модулни методи.

Глава 2

Анализ на съществуващите методи и програмни продукти

В тази глава се дефинират някои важни свойства на методите за криене на данни в мултимедия по отношение на приложението им в Интернет-базирани сценарии. Разглеждат се някои от съществуващите методи и програмни продукти в областта, като се оценява реализацията на свойствата във всеки един от тях.

2.1 Важни свойства на криенето на данни в мултимедия

Методите за криене на данни в мултимедия могат да имат различни свойства като устойчивост срещу геометрични трансформации, промени на формата, промяна на региони от мултимедийното съдържание и др. [44], [45]. Три свойства са особено важни по отношение на Интернет-базираните сценарии:

1. Адаптивност на методите към конкретните нужди на клиентите;
2. Устойчивост срещу JPEG-базирани трансформации;
3. Възможност за работа с произволна мултимедия и вграждани данни.

Тези свойства се разглеждат по-подробно в следващите раздели.

2.1.1 Адаптивност към променливи изисквания на потребителите

Присъщата отвореност и непредсказуемост на глобалната мрежа в комбинация с бързите и резки промени в съвременните социални, бизнес и технологични среди водят до постоянни изменения на потребностите и изискванията на клиентите. Някои от тези изменения касаят и областите на приложение на криенето на данни в мултимедия. Поради тази причина методите за криене на данни трябва да могат да бъдат лесно адаптирани към промени в изискванията на клиентите – т.е. те трябва да могат да променят част от свойствата си (в зависимост от конкретното приложение и цели на клиента), като в същото време запазват една основна функционалност, която потребителите очакват от всички такива методи.

Изискването за адаптивност на методите налага да се вземат под внимание следните фактори за неговото постигане:

1. Прилагане на нов модулен подход при разработката на методите. Съществуващите монолитни решения не са адаптируеми и не могат да бъдат лесно напасвани в съответствие с честите промени в модерната глобална мрежа.
2. Прилагане на утвърдена и стандартизирана технология за свързване на методите с вече съществуващи системи. Тя ще позволи вграждането на модулните методи за криене на данни във вече изградена корпоративна инфраструктура и ще предостави възможност за гъвкава интеграция с други технологии - криптография, компресия и т.н.

И двата фактора показват необходимостта от глобален, гъвкав, и в същото време стандартизиран подход към криенето на данни в мултимедия, за разлика от съществуващите в момента нестандартизирани монолитни решения.

2.1.2 Устойчивост срещу JPEG-базирани трансформации

Важно изискване към методите за криене на данни в мултимедия, особено ако се цели тяхното приложение в Интернет, е запазването на вградената информация след компресия на мултимедийното съдържание [6], [14]. Това изискване е свързано с предимството гъвкавост (вж. раздел 1.1.1).

Компресията се прилага, за да намали размера на предаваната мултимедия. Най-често се използва т.нар. загубна компресия [46], при която се губи част от мултимедийното съдържание, но се постига с пъти по-високо ниво на компресия от стандартните беззагубни методи за компресия. Един от най-важните формати за загубна компресия на изображения, намиращ широко приложение и при кодиране на видео кадри, е JPEG, който е базиран на т.нар. дискретна косинусова трансформация (DCT) [47]. Неговата универсалност и добра степен на компресия са направили JPEG предпочитан избор за съхранение и предаване на цветни изображения. Поради тази причина е важно данните, вградени в изображения и видео кадри, да не се влияят от JPEG трансформации.

Съществуват три основни типа JPEG трансформации:

1. Компресия – кодиране на матрица от пиксели в JPEG формат. Всяко цифрово изображение трябва да се представи под формата на правоъгълна матрица от пиксели преди извършването JPEG компресия.
2. Декомпресия – декодиране на JPEG формат до матрица от пиксели.
3. Рекомпресия – промяна на степента на компресия или други параметри на компресията на изображение, което е вече кодирано в JPEG формат.

Устойчивостта на вградените данни срещу трите вида трансформации е важна предпоставка за гъвкавото приложение на методите за криене на данни в глобалната мрежа. Компресията се използва, за да намали размера на новосъздадени изображения и за да ги направи разпознаваеми за Интернет браузърите. Декомпресията се използва за прочитане на мултимедийното съдържание с цел показването му на екран, промяна

или запомняне в друг формат. Рекомпресията се използва най-често, за да се намали размерът на вече компресиран мултимедиен файл. Тя често се прилага към изображения преди тяхното разпространение през Интернет-базирани или мобилни канали (изпращане по e-mail, пренос чрез файлов сървър или мрежова услуга за споделяне на файлове, публикуване в Интернет портал, копиране и употреба в мобилни устройства).

Важно е да се постигне устойчивост срещу изпълнението на тези три вида трансформации от произволни програми за обработка на изображения/видео, като се цели минимална промяна на съдържанието от страна на методите за криене на данни. За тази цел е необходимо детайлно познаване на JPEG стандарта за компресия на изображения и неговите практически реализации в разпространените програмни системи и програмни библиотеки за обработка и компресия на изображения, които имат свободата да променят определени параметри на обработка на мултимедийното съдържание в рамките на стандарта [47], [48].

2.1.3 Работа с произволна мултимедия и вградени данни

Методите за криене на данни трябва да бъдат достатъчно гъвкави, така че да могат да обработват произволна мултимедия, предоставена от крайните потребители. Това изискване води до две важни последствия:

1. Методите за вграждане на данни трябва да могат да работят с черно-бели, полутонови (с нива на сиво) и цветни изображения и не трябва да зависят от никакви характеристики, специфични за даден клас изображения.
2. Оригиналното изображение или статистическа информация, която го описва не трябва да бъдат необходими за декодиране на вградените данни. Методите за вграждане на данни, които имат това свойство, се наричат слепи методи [6] и осигуряват максимална гъвкавост на приложение.

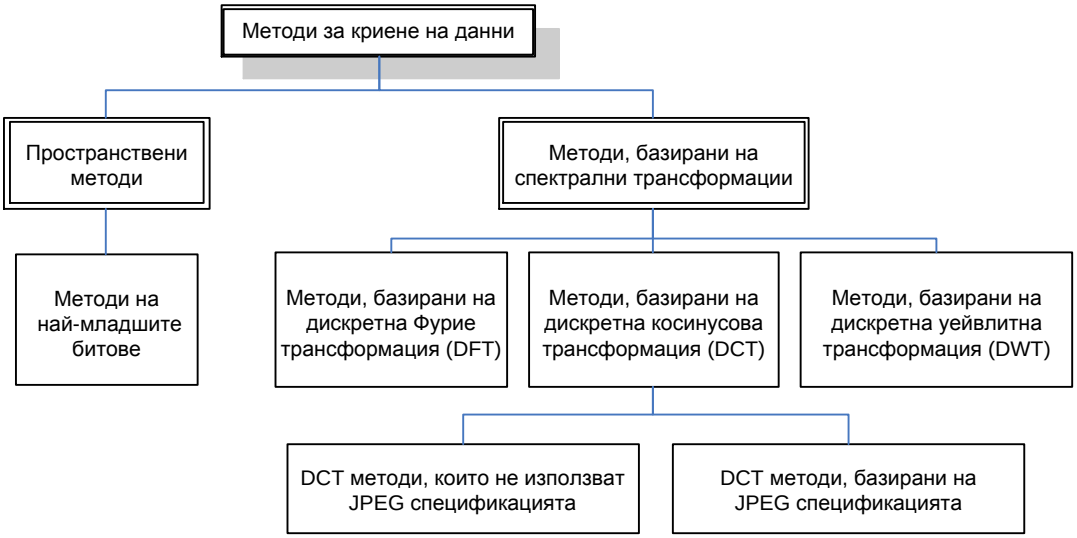
Тъй като методите за криене на данни за приложения в глобалната мрежа често работят с криптирани или компресирани данни, предоставени от крайния потребител, то тези методи не трябва да налагат ограничения върху формата или последващата употреба на вградените данни. Два основни аспекта трябва да бъдат взети под внимание:

1. Данните за вграждане трябва да бъдат разглеждани като произволен поток от двоични данни.
2. Възстановяването на вградените данни без наличие на грешки трябва да бъде възможно с оглед на тяхната последваща употреба от други произволни технологии.

Възможността за работа с произволна мултимедия и вграждани данни подобрява до голяма степен адаптивността на методите за криене на данни към променящите се нужди и изисквания на клиентите, което прави това свойство особено важно при разработката на модулния подход за криене на данни в мултимедия.

2.2 Съществуващи методи и програмни продукти за криене на данни в мултимедия

Настоящият раздел представя обзор на съществуващите научноизследователски методи за криене на данни в мултимедия и на програмните продукти, предлагани на пазара, като главен фокус е тяхната употреба в Интернет-базирани сценарии. Най-важните типове методи за криене на данни могат да се класифицират както е показано на Фиг. 3 ([7], [45], [49], [14], [38]).



Фиг. 3: Класификация на методите за криене на данни

Пространствените методи работят директно с пикселите на изображението или видео кадъра. Най-често те попадат в групата на т.нар. методи на най-младшите битове, които вграждат данни в изображенията посредством промяна на най-младшите битове на описанието на пикселите. Методите за криене на данни, които използват спектрални трансформации, оперират върху резултата от различни математически трансформации, приложени върху изображението. Тъй като загубната компресия на изображения често се базира върху такива трансформации, то тези методи за криене на данни работят добре, когато вградената информация трябва да бъде устойчива на компресия на изображението или видео кадъра.

Три широко използвани трансформации са дискретната Фурие трансформация (англ. DFT), дискретната косинусова трансформация (англ. DCT) и дискретната уейвлитна трансформация (англ. DWT). Първите две трансформации представят изображението като сума от косинусови и/или синусови функции с фиксирани честоти и амплитуди, определени от пикселите на изображението. Дискретната уейвлитна трансформация дава информация за честотите, присъстващи в изображението, като запазва и част от пространствената информация за положението на пикселите.

Методите за криене на данни, базирани върху DCT, използват същата математическа трансформация, която се прилага в JPEG стандарта за компресия [47]. Тези методи могат да бъдат допълнително разделени на две групи: DCT методи, които не използват спецификацията на JPEG, и DCT методи, базирани на детайлния анализ на JPEG спецификацията.

Поради големия брой съществуващи методи за криене на данни и важното значение на JPEG формата за обмен на изображения в Интернет, в този раздел се разглеждат само *методите, базирани на DCT трансформацията*, като се изследва тяхната устойчивост срещу различните видове JPEG трансформации. Както се дискутира в раздел 2.1.2, групата на DCT методите, базирани на спецификацията на JPEG, е от специално значение. Също така се разглежда до каква степен тези методи притежават свойствата, дискутирани в раздели 2.1.1 и 2.1.3.

2.2.1 Съществуващи научноизследователски разработки и методи

Първият JPEG-базиран метод за криене на данни в мултимедия е разработен през 1993 от Дерек Уфам и носи името JSTEG [50], [51], [52], [53]. Той е стеганографски метод, който вгражда произволна двоична информация чрез промяна на най-младшите битове на DCT коефициентите на JPEG изображенията. Друг ранен метод за криене на данни в мултимедия, разработен за целите на цифровото маркиране, е предложен от Джао и Кох, Институт Фраунхофер, Дармщат през 1995 [54], [55]. Той крие един бит във всеки DCT блок посредством създаването на специална зависимост между елементите на множество, съставено от три DCT коефициента. О'Руанаид и сътрудниците му от Женевски Университет, предлага през 1996 метод за надеждно бидирекционално кодиране на цифрови водни знаци в DCT коефициенти [56]. Методът не е „сляп", но може да работи с произволно избрани данни, използвани като воден знак. Никой от тези класически методи не дискутира устойчивостта срещу JPEG декомпресия или рекомпресия.

Друг метод за цифрово маркиране е разработен през 1997 от Кокс и сътрудниците му от Университетски Колеж, Лондон [57]. Методът вгражда цифрови водни знаци, получени чрез Гаусово нормално разпределение, в DCT коефициенти посредством скалиращи функции. Алгоритъмът е устойчив, но не е сляп и не може да работи с произволни цифрови водни знаци. През 1998, Ву и Лиу, Университет Принстън, разработват друг метод за цифрово маркиране [58]. Той крие „двоични данни, имащи визуално значение" – на практика малко черно-бяло лого, комбинирано с някои характеристики на носещото изображение. Тези данни се вграждат в квантизираните DCT коефициенти посредством специални референтни таблици (англ. look-up tables). Устойчивостта срещу JPEG декомпресия и JPEG рекомпресия не са взети под внимание.

Лин и Чанг, Колумбийски Университет, представят през 2000 година метод за цифрово маркиране, който цели откриването и частичното възстановяване на променени

части от изображенията чрез използването на специални водни знаци, съставени от груби апроксимации на оригиналното изображение [59], [60], [61]. Методът е разработен така, че да бъде устойчив срещу JPEG рекомпресия, но авторите дискутират в [60] погрешни отчитания на несъществуващи промени в изображенията (англ. false alarms), дължащи се на допълнителния шум, предизвикан при провеждането на JPEG компресия и декомпресия.

Стеганографски метод за вграждане на данни, разработен с изричната цел данните да не могат да бъдат открити от методи за статистически стеганализ, е предложен от Нилс Провос, Мичигански Университет, през 2001 година [62], [63] и е имплементиран под програмното име *OutGuess*. Андреас Вестфелд, Технически Университет Дрезден, предлага през същата година метод, наречен *F5*, разработен за целите на стеганографията [64]. Този метод използва алгоритъм за т.нар. матрично кодиране (англ. matrix coding) [65] с цел да минимизира броя на необходимите промени на DCT коефициентите и следователно, подобно на *OutGuess*, да не може да бъде открит от методите за статистически стеганализ. В по-късни публикации е доказано, че горните два метода за криене на данни се поддават на специализирани статистически стеганализ атаки [66], [67]. Освен това те не вземат под внимание устойчивостта срещу JPEG декомпресия или рекомпресия. Своеобразно тяхно подобрение е предложено през 2004 година от Салее [68] с цел да се подобри устойчивостта срещу стеганализ. Това подобрение е базирано на статистически изследвания на изображенията и вградените данни, но не отчита устойчивостта срещу JPEG декомпресия или рекомпресия.

Друг стеганографски метод за криене на данни в JPEG изображения чрез промяна на квантизиращата JPEG таблица е предложен от Чанг и неговите колеги през 2002 година [69]. Методът заменя квантизиращите стойности, съответстващи на честотите от средния диапазон, със стойности равни на “1” и вгражда данни в DCT коефициентите, съответстващи на тези честоти. Устойчивост срещу JPEG декомпресия или JPEG рекомпресия не се разглежда.

Джесика Фридрих, Университет Бингамтън, и нейната научноизследователска група предлагат няколко метода за цифрово маркиране, базирани върху DCT трансформации. В метода, описан в [70] и [71], носещото изображение се разделя на блокове с големина 64x64 пиксела. Всеки блок се трансформира в DCT спектрално пространство. След това цифров воден знак, дефиниран от потребителя, се вгражда в DCT коефициентите. Методът, разгледан в [40] и [72], вгражда специализиран воден знак, който позволява частично реконструиране на блоковете от изображението, променени неправомерно от трети лица. Авторите също така предлагат метод за т.нар. беззагубно вграждане на данни (англ. lossless data embedding) [73], [74], [75]. Методът вгражда воден знак, дефиниран от потребителя, и позволява пълно реконструиране на оригиналното немаркирано изображение от получателя. Всички алгоритми, предложени от Фридрих и нейните колеги вземат под внимание JPEG компресията, но не и декомпресията или рекомпресията.

Друг метод, предложен от Джао и колегите му през 2008, използва т.нар. Котешка Трансформация на Арнолд (англ. Arnold's Cat Map transform) [76]. Методът използва разбъркано черно-бяло изображение като цифров воден знак, който се вгражда в DCT коефициентите на изображения с нива на сиво (англ. grey-level images) [77]. През същата година Джанг и колегите му предлагат метод, базиран на референтни таблици, който използва статистически модел, за да намали изкривяванията на носещото изображение [78]. Двата метода, посочени по-горе, са устойчиви срещу JPEG компресия, но не целят безпогрешно възстановяване на вградените данни. JPEG декомпресия и рекомпресия не се разглеждат.

Някои от съвременните методи за криене на данни в мултимедия използват техника, наречена Модулация на Индекса на Квантизация (англ. Quantization Index Modulation - QIM), разработена от Коща през далечната 1983 [79]. Тази техника е преоткрита и анализирана от Чен и Уорнел през 2001 за целите на цифровото маркиране и неговите приложения [80]. Ли и Кокс предлагат през 2007 метод за цифрово маркиране, базиран на QIM и перцептуалния модел за оценка на изображения, разработен от Уотсън [81], [82]. Подобрена версия на метода е разработена през 2008 година от Сун и колегите му [83]. Двата метода са създадени така, че да бъдат устойчиви на промени в яркостта на изображенията, но безпогрешното възстановяване на вградените цифрови водни знаци след JPEG компресия не е възможно.

Друг стеганографски метод, използващ QIM, е предложен от Изадиния и колегите му през 2009 [84]. Методът вгражда произволни съобщения чрез прилагането на алгоритъм за предсказващо кодиране (англ. predictive coding), предложен от Юу и колегите му [85], към квантизираните DCT коефициенти. Устойчивостта на метода срещу JPEG декомпресия и JPEG рекомпресия не е разгледана. Също през 2009, Равал предлага използването на специална таблица, получена от статистическите особености на изображението, при определянето на коефициентите на QIM [86]. Устойчивостта на метода срещу JPEG декомпресия и JPEG рекомпресия не е разгледана.

Кратка обща оценка на всички разгледани методи по отношение на свойствата, дискутирани в раздел 2.1 е представена в Табл. 1 („Н.Р." обозначава "Не се Разглежда").

Никой от разгледаните методи за криене на данни в мултимедия не дискутира свойството адаптивност (свойство 2.1.1). Тези методи са монолитни решения, разработени за целите на специализирани приложения със свои собствени конкретни изисквания относно свойствата на методите. Авторите не дискутират как тези методи биха могли да бъдат интегрирани като част от вече съществуващи решения и инфраструктури.

Устойчивостта срещу JPEG трансформации и възможността за работа с произволна мултимедия и вграждани данни са само частично взети под внимание. Повечето методи разглеждат само определени аспекти на тези свойства. Потенциалът за подобрение

се състои в едновременното реализиране на всички значими аспекти на свойствата, представени в раздел 2.1.

Табл. 1: DCT-базирани методи за криене на данни в мултимедия – обща оценка

Метод	Адаптивност	Устойчивост срещу JPEG трансформации			Възможност за работа с произволна мултимедия и вграждани данни			
		Компресия	Декомпресия	Рекомпресия	Произволна мултимедия	„Сляп метод”	Произволни данни	Безпогрешно възстановяване на данните
JSteg [50], [52]	не	да	Н.Р.	Н.Р.	да	да	да	да
Джао, Кох [54], [55]	не	да	Н.Р.	Н.Р.	да	да	да	да
О’Руанаид [56]	не	да	Н.Р.	Н.Р.	да	не	да	да
Кокс [57]	не	да	да	да	да	не	не	не
Ву, Лиу [58]	не	да	Н.Р.	Н.Р.	да	да	не	да
Лин, Чанг [59], [60], [61]	не	да	да	да	да	да	не	частично
Провос [62], [63]	не	да	Н.Р.	Н.Р.	да	да	да	да
Вестфелд [64]	не	да	Н.Р.	Н.Р.	да	да	да	да
Салее [68]	не	да	Н.Р.	Н.Р.	да	да	да	да
Чанг [69]	не	да	Н.Р.	Н.Р.	да	да	да	да
Фридрих [70], [71]	не	да	Н.Р.	Н.Р.	да	да	да	не
Фридрих [40], [72]	не	да	Н.Р.	Н.Р.	да	да	не	да
Фридрих [73], [74], [75]	не	да	Н.Р.	Н.Р.	да	да	да	да
Джао [77]	не	да	Н.Р.	Н.Р.	да	да	не	не
Джанг [78]	не	да	Н.Р.	Н.Р.	да	да	да	не
Ли, Кокс [81]	не	да	Н.Р.	Н.Р.	да	да	да	не
Сун [83]	не	да	Н.Р.	Н.Р.	да	да	да	не
Изадиния [84]	не	да	Н.Р.	Н.Р.	да	да	да	да
Равал [86]	не	да	Н.Р.	Н.Р.	да	да	да	да

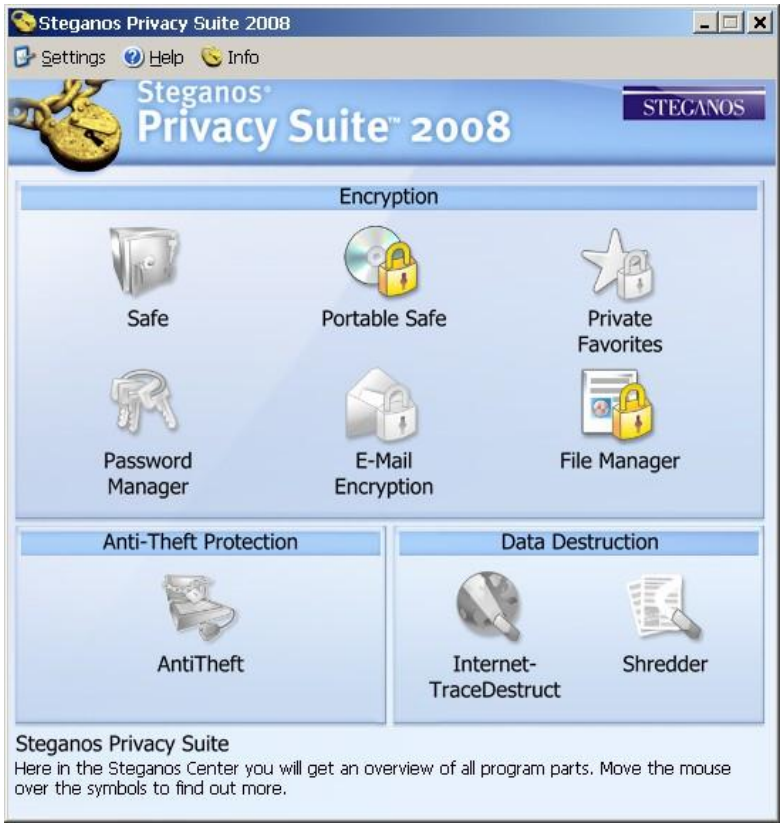
* Н.Р. = Не се Разглежда

2.2.2 Съществуващи програмни продукти и услуги

Вследствие на силния академичен и корпоративен интерес към криенето на данни в мултимедия, съществуват редица практически насочени програмни продукти и услуги в областта на стеганографията и цифровото маркиране. Те са добили популярност в последните години и се предлагат в Интернет или като част от по-големи програмни пакети.

Едно от най-популярните стеганографски решения на пазара е Steganos Privacy Suite [87] (Фиг. 4). Приложението *Файлов Мениджър* (англ. *File Manager*) може да вгражда данни в компресирани или некомпресирани носещи изображения. Скритата

информация е устойчива срещу JPEG компресия при ниски нива на компресия, но не е устойчива срещу JPEG декомпресия или рекомпресия. Използваният стеганографски метод е сляп и може да работи както с произволни файлове с данни, така и с произволни носещи изображения. Важно предимство на програмата и използвания от нея метод е отличното качество на изображенията след вграждането на данните.



Фиг. 4: Steganos Privacy Suite

Класическа стеганографска програма за вграждане на произволни файлове с данни в JPEG изображения е JPHide [88]. Използваният от нея стеганографски метод е устойчив срещу JPEG компресия при ниски нива на компресия, но не е устойчив срещу JPEG декомпресия или рекомпресия. Методът е сляп и може да работи с произволни носещи JPEG изображения и, подобно на Steganos Privacy Suite дава като резултат изображения с отлично качество.

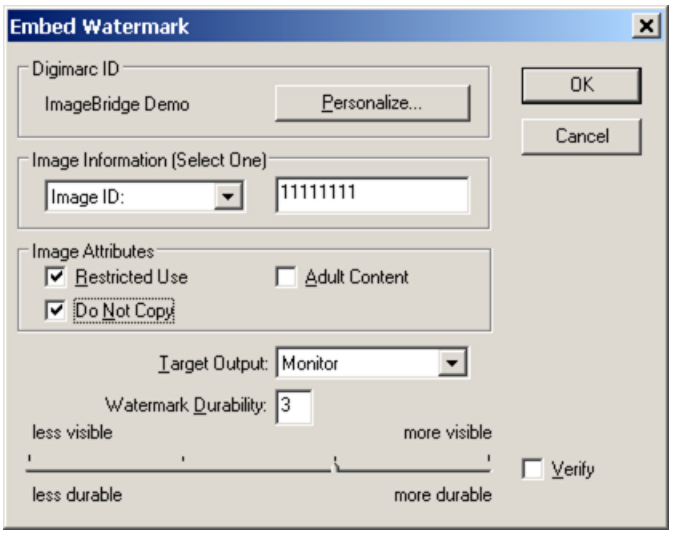
Нова стеганографска разработка е програмата InvisibleSecrets [89] (за момента във версия 4). Тя има удобен графичен потребителски интерфейс (Фиг. 5) и поддържа компресирани и некомпресирани графични формати. По отношение на JPEG, програмата крие информацията в сегментите за коментари на JPEG файла (вж. [47]). Този подход има предимството, че не налага лимити върху размера на вгражданите данни, но анулира много от предимствата на криенето на данни в мултимедия, разглеждани в раздел 1.1.1. Използваният метод може да работи с произволни данни и е устойчив срещу JPEG компресия и рекомпресия, но не е устойчив срещу JPEG декомпресия.

Digimarc [90] е една от водещите компании, специализирани в областта на цифровото маркиране. Най-известният продукт на компанията е програмен модул (англ. plug-in) за Фотошоп (англ. Photoshop) – Фиг. 6. Той вгражда в цифровото съдържание кратък идентификационен номер (англ. identification number - ID) заедно с три булеви атрибута, задаващи параметри на позволената употреба на изображението. Идентификационният номер се използва в други програмни решения, предлагани от Digimarc – напр. като ключ за търсене в услугата за намиране на изображения, която сканира Интернет за изображения, принадлежащи на даден клиент. Той също така служи като уникален идентификатор на изображенията при интеграцията с други системи за управление на цифрово съдържание, разработени от компанията за целите на корпоративни потребители.

Фиг. 5: Invisible Secrets

Услугата за търсене на изображения, предлагана от Digimarc, [91] сканира Интернет портали за цифрови изображения, чиито авторски права се държат от клиенти на Digimarc. Най-напред се идентифицират всички изображения, използвани от даден Интернет портал. След това услугата за търсене се опитва да прочете вече вграден идентификационен номер от всяко изображение. Ако идентификационният номер съществува и съответства на текущ потребител на услугата, то тогава местоположението на съответното изображение се съобщава на потребителя. По този начин услугата на Digimarc помага на потребителите да открият адресите в Интернет, на които техни цифрови изображения са публикувани онлайн.

Методът за вграждане на цифрови водни знаци, използван от Digimarc, е устойчив до голяма степен срещу JPEG компресия, декомпресия и рекомпресия. Крайният потребител може да регулира степента на устойчивост за сметка на качеството на крайното изображение след вграждане на цифровия воден знак (вж. Глава 5) посредством слайдера в долната част на прозореца на програмния модул (Фиг. 6). Методът е сляп и работи с произволни носещи изображения.

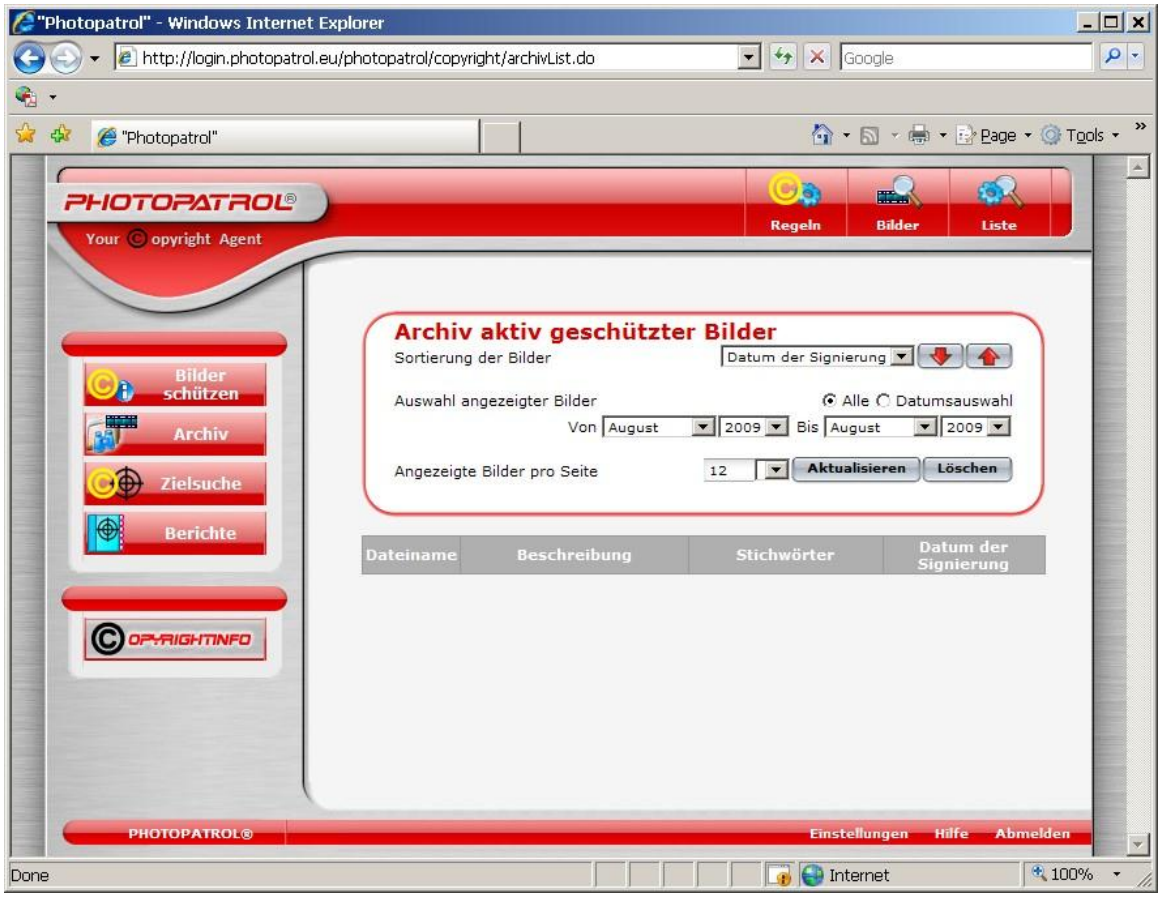


Фиг. 6: Вграден модул (англ. plug-in) за Photoshop на Digimarc

Друга компания, предоставяща услуги за цифрово маркиране, е Фотопатрул (англ. Photopatrol) [92], която използва технология за цифрово маркиране, разработена в Институт СИТ-Фраунхофер (англ. Fraunhofer Institute SIT), Дармщат [93]. Приложенията на компанията се предлагат под формата на Интернет базирани услуги (англ. Software-as-a-Service, SaaS). Потребителят не инсталира програмни продукти на своя компютър, а използва тяхната функционалност през Интернет портал, на който има персонален заплатен акаунт за ползване (Фиг. 7). Photopatrol предлага две основни онлайн услуги – услуга за вграждане на цифрови водни знаци (изпълняващи ролята на сигнатури) в изображения и услуга за сканиране на предварително зададени Интернет портали за присъствие на вградени водни знаци. Услугата за вграждане на водни знаци в изображения е реализирана с помощта на модерни технологии, използвани в Интернет браузърите и Java аплети. За съжаление тя е сравнително сложна за използване и не се препоръчва за начинаещи Интернет потребители. Услугата за сканиране на Интернет портали е подобна на тази, предоставена от Digimarc. Тя сканира предварително зададени Интернет адреси за присъствието на изображения, принадлежащи на клиенти на Фотопатрул, което се установява посредством вградените водни знаци. Ако такива изображения бъдат намерени, то тяхното местоположение се съобщава на съответния клиент.

Технологията, използвана от Photopatrol осигурява устойчивост срещу JPEG трансформации. Методът е сляп и може да работи с произволни носещи изображения.

Самостоятелна програма с графичен потребителски интерфейс е SignMyImage (актуална версия 3.06) [94]. Програмата има удобен потребителски интерфейс (Фиг. 8) и може да вгражда идентификационен текстов низ (стринг), състоящ се от до 10 знака. Авторът на програмата предлага също така услуга за сканиране на Интернет портали като тези, предоставяни от Digimarc и Фотопатрул [95]. Използваният метод за цифрово маркиране е устойчив срещу JPEG компресия и декомпресия при ниски нива на компресиране. Освен това методът е сляп и може да работи с произволни носещи изображения.

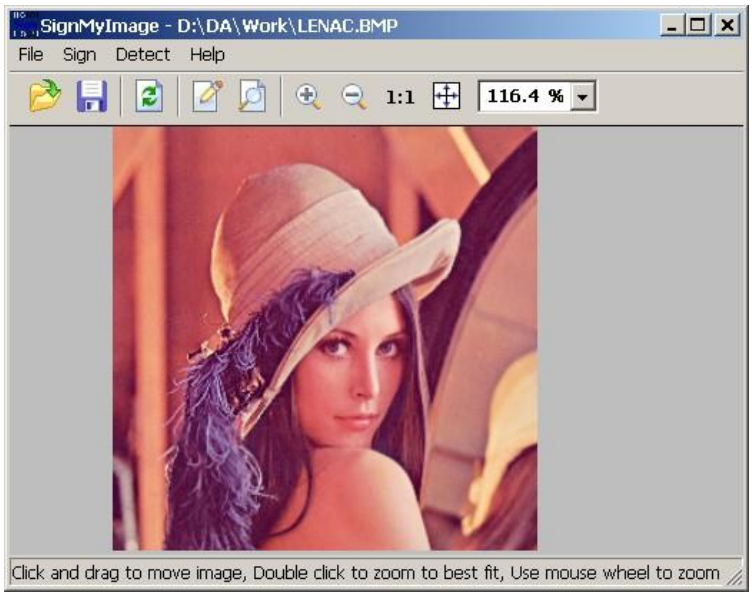


Фиг. 7: Потребителски интерфейс на Photopatrol (в браузър Internet Explorer)

Друга самостоятелна програма за цифрово маркиране е Icemark (актуална версия 1.2) [96]. Тя може да вгражда до 20 байта информация в носещи изображения. Вградената информация е устойчива срещу JPEG трансформации при ниски нива на JPEG компресия. Използваният метод за цифрово маркиране е „сляп" и работи с произволни носещи изображения.

Друга алтернатива е самостоятелната програма с графичен потребителски интерфейс Eikonamark (актуална версия 4.8) [97]. Програмата може да вгражда до 8 байта в произволни носещи изображения. Скритата информация е устойчива срещу JPEG компресия, но не и срещу декомпресия или рекомпресия. Използваният метод за цифрово

маркиране е сляп. Авторите предлагат също така услуга за претърсване на Интернет портали за изображения, които съдържат цифрови водни знаци (използвани като сигнатури), вградени от Eikonamark [98].



Фиг. 8: Графичен потребителски интерфейс на SignMyImage

Нов набор от програмни решения в областта на цифровото маркиране се предлага от сингапурската компания DataMark [99]. Основният продукт на компанията е StegMark SDK, който предоставя комплект от програмни библиотеки за няколко популярни езика за програмиране. Тези библиотеки могат да се използват от клиентите на компанията за реализиране на функции за цифрово маркиране в техните собствени програмни продукти и решения.

Кратка обща оценка на методите за криене на данни, използвани в описаните програмни продукти и услуги, е представена в Табл. 2 („Н.Р.” обозначава “Не се Разглежда”). Разглеждат се свойствата, дискутирани в раздел 2.1. Поради някои свои ограничения, методите за цифрово маркиране на DataMark не са включени в оценката.

Различията между стеганографските решения и решенията в областта на цифровото маркиране са ясно видими. Продуктите, създадени за стеганографски цели, могат да работят с произволни носещи изображения и данни за вграждане, но не са толкова устойчиви срещу JPEG трансформации колкото разгледаните продукти за цифрово маркиране. От друга страна, продуктите за цифрово маркиране могат да вграждат само малки по размер, предварително дефинирани типове данни – най-често идентификационни номера, но с добра степен на устойчивост срещу JPEG трансформации.

Нито едно от съществуващите програмни решения не взема предвид адаптивността към нуждите на клиентите. Тези решения са изградени монолитно, на базата на конкретни и завършени методи и не могат да се адаптират към променящите се изисквания

на потребителите в Интернет, които налагат промени в свойствата на методите за криене на данни в мултимедия.

Табл. 2: Продукти и услуги за криене на данни в мултимедия – обща оценка

Продукт / Услуга	Адаптивност	Устойчивост срещу JPEG трансформации			Възможност за работа с произволна мултимедия и вграждани данни			
		Компресия	Декомпресия	Рекомпресия	Произволна мултимедия	„Сляп метод"	Произволни данни	Безпогрешно възстановяване на данните
Steganos Privacy Suite [79]	не	частично	не	Н.Р.	да	да	да	да
JPHide [80]	не	частично	не	Н.Р.	да	да	да	да
InvisibleSecrets [81]	не	да	не	да	да	да	да	да
Digimarc [90]	не	да	да	да	да	да	не	да
Photopatrol [92]	не	да	да	да	да	да	не	да
SignMyImage [94]	не	частично	частично	Н.Р.	да	да	не	да
Icemark [96]	не	частично	частично	Н.Р.	да	да	не	да
Eikonamark [97]	не	да	не	Н.Р.	да	да	не	да

* Н.Р. = Не се Разглежда

2.3 Заключение

В резултат на направения обзор могат да се идентифицират следните недостатъци на съществуващите методи за криене на данни в мултимедия, базирани на DCT трансформации:

1. Съществуващите методи и продукти за криене на данни в мултимедия са монолитни и с конкретна приложна насоченост. Те предлагат на потребителя константен набор от свойства и не могат да се адаптират към промени в изискванията. Както е показано в Глава 5, съществува компромис между качеството на крайното изображение, количеството вграждана информация и свойствата, предоставяни от всеки метод (напр. устойчивост срещу JPEG трансформации или други промени в съдържанието на мултимедията). Подобряването на някои от тези три критерия води неминуемо до влошаването на друг. Поради тази причина свойства, които не са необходими на даден потребител, не би трябвало да се реализират от методите, за да може да се осигури по-привлекателен компромис (напр. по-добро качество на изображението). Създаването на монолитни решения ограничава такъв вид гъвкавост по отношение на предоставяните свойства, което се явява важен недостатък,

ако се обслужват различни видове клиенти в динамична среда като Интернет. Потенциалът за подобрение лежи в осигуряването на възможност за динамично адаптиране на методите чрез разделянето им на отделни модули, което на този етап не се разглежда от авторите на традиционните методи и програмни решения за криене на данни в мултимедия.

2. Много малко на брой методи са разработени,за да бъдат устойчиви срещу JPEG декомпресия или рекомпресия и те не винаги могат да работят с произволни изображения и вградени данни. Тези свойства улесняват адаптивността на методите и подобряват тяхната приложимост в Интернет-базирани сценарии, но всеки от представените методи и продукти ги реализира по различен начин. Всеки метод има различна степен на устойчивост срещу JPEG трансформации и налага различни ограничения върху възможността за работа с произволна мултимедия и данни за вграждане. Следователно, ако дадена компания трябва да използва няколко различни метода за криене на данни в няколко приложни области, то тя трябва изрично да вземе предвид разликите между методите.

В тази дисертация се дискутира нов модулен подход за криене на данни в мултимедия за целите на мрежови приложения. Той е разработен и реализиран под формата на програмно решение с цел преодоляването на недостатъците на съществуващите методи.

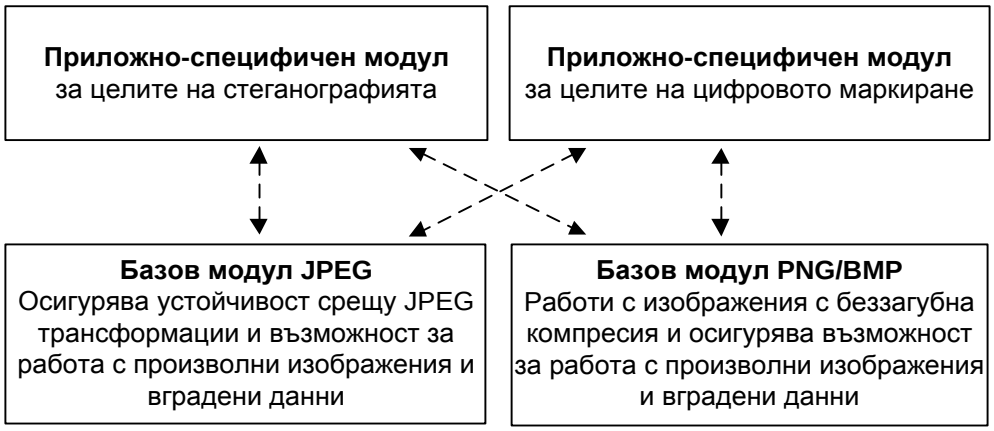
Глава 3

Модулен подход за проектиране на методи за криене на данни

В тази глава се дискутира модулното проектиране на методите за криене на данни в мултимедия и се представят неговите предимства. Един модулен метод за целите на стеганографията и един модулен метод за целите на цифровото маркиране се създават специално за приложение в Интернет-базирани сценарии и техните свойства се описват и дискутират в подробности.

3.1 Модулно проектиране – общ поглед и архитектура

При реализацията на методите за криене на данни те се разглеждат като съставени от два вида модули –базов модул и приложно-специфичен модул. Двата вида модули могат да се комбинират в различни конфигурации (Фиг. 9).

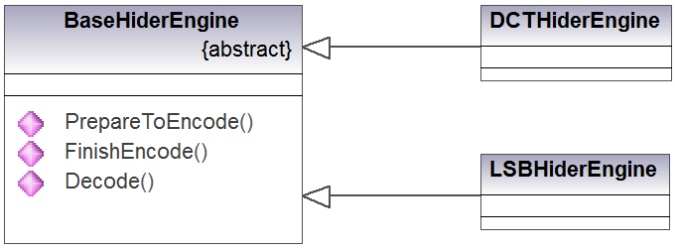


Фиг. 9: Модулен подход за криене на данни в мултимедия за целите на мрежови приложения

Базовите модули са отговорни за обработката на графичните формати и реализирането и гарантирането на важни общи свойства, характерни за голям брой методи, като например устойчивостта срещу JPEG трансформации и работата с произволни двоични данни за криене. Тези свойства се подобряват или използват за реализирането на нови свойства от приложно-специфичните модули. Приложно-специфичните модули се раз-

работват в зависимост от изискванията на потребителите и се адаптират за приложения в различни области.

Модулите се проектират с помощта на класическото обектно-ориентирано програмиране (англ. object-oriented programming, OOP) [100], [43]. При този програмен подход изпълняваната компютърна програма се състои от множество обекти, които комуникират помежду си чрез стартиране на блокове програмен код, свързани с обектите, които се наричат методи на обекта. Всеки обект се дефинира посредством т.нар. клас, описан в един или повече текстови файла. Класът представлява пълно описание на обекта и определя блоковете програмен код (методи), свързани с него, както и променливите в работната памет (свойства), които съхраняват информация за обекта. Характерна особеност на подхода е, че множество обекти могат да бъдат създадени от описанието в един единствен клас и могат да съществуват по едно и също време в работната памет. Също така модерните многопроцесорни машини позволяват изпълнението на множество програмни инструкции едновременно, което прави възможно няколко обекта да се обработват в паралел. На по-високо ниво това означава, че множество двойки от изображения и данни за вграждане могат да се обработват по едно и също време от различни методи за криене на данни, съставени от едни и същи или различни модули. Програмната реализация на всеки базов или приложно-специфичен модул се осъществява чрез отделен клас, описан в няколко файла. Ако даден модул ще се използва за вграждане на данни, то по време на изпълнение на програмата от съответстващия му клас се създава нов обект.



Фиг. 10: Базов модул

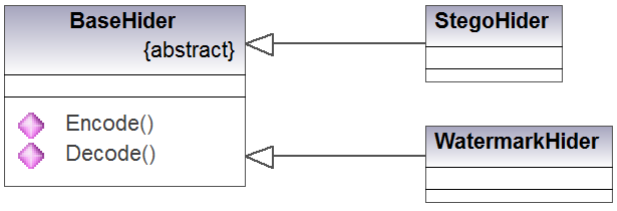
На Фиг. 10 е показана т.нар. диаграма на класовете, част от графичния език за моделиране на обектно-ориентирани програми UML. Този тип диаграми показват използваните в програмата класове, техните методи и свойства и връзките между тях. На фигурата са изобразени само класовете, описващи базовите модули, и връзките помежду им. Съществуват общо три класа – абстрактен базов клас, наречен *BaseHiderEngine (базова машина за криене)* и два производни класа, наречени *DCTHiderEngine (DCT машина за криене)* и *LSBHiderEngine (LSB машина за криене)*. Абстрактният базов клас не може да се използва директно за създаване на обект в програмата – само двата производни класа съответстват на използваеми базови модули, като всеки клас реализира по един базов модул.

Базовият клас *BaseHiderEngine* се използва за дефинирането на публични методи за подготовка на кодирането (*PrepareToEncode*), за завършване на кодирането (*FinishEncode*) и за декодиране (*Decode*). С помощта на тези методи базовите модули комуникират с приложно-специфичните модули. Стрелките в диаграмата показват, че класовете *DCTHiderEngine* и *LSBHiderEngine* наследяват класа *BaseHiderEngine*, което означава, че всеки от тях трябва да предостави валидна програмна реализация на публичните методи, дефинирани в *BaseHiderEngine*.

Приложно-специфичните модули имат подобна структура, която е показана на диаграмата на класовете във Фиг. 11. Всеки приложно-специфичен модул се реализира в свой собствен клас, който трябва да наследи базовия абстрактен клас, наречен *BaseHider*, и да предостави програмна реализация на публичните методи за кодиране (*Encode*) и декодиране (*Decode*). Реализирани са общо два приложно-специфични модула чрез класа за стеганографско криене на данни, наречен *StegoHider* и класа за криене на водни знаци за цифрово маркиране, наречен *WatermarkHider*.

Всяко външно приложение или част от програма започва процеса за вграждане на данни чрез стартирането на метода за кодиране (*Encode*) на приложно-специфичния модул. Процесът на вграждане се състои от три етапа (Фиг. 12):

1. В началото се изпълнява методът за подготовка на кодирането (*PrepareToEncode*) на базовия модул. Той има за задача да раздели изображението на блокове и да определи максималното количество битове, които могат да бъдат скрити във всеки блок.
2. В съответствие с количеството информация, което може да се вгради, приложно-специфичният модул определя точния брой и стойностите на битовете, които ще се вградят във всеки отделен блок.
3. Най-накрая се стартира методът за завършване на кодирането (*FinishEncode*), който извършва същинското кодиране на битовете в изображението.

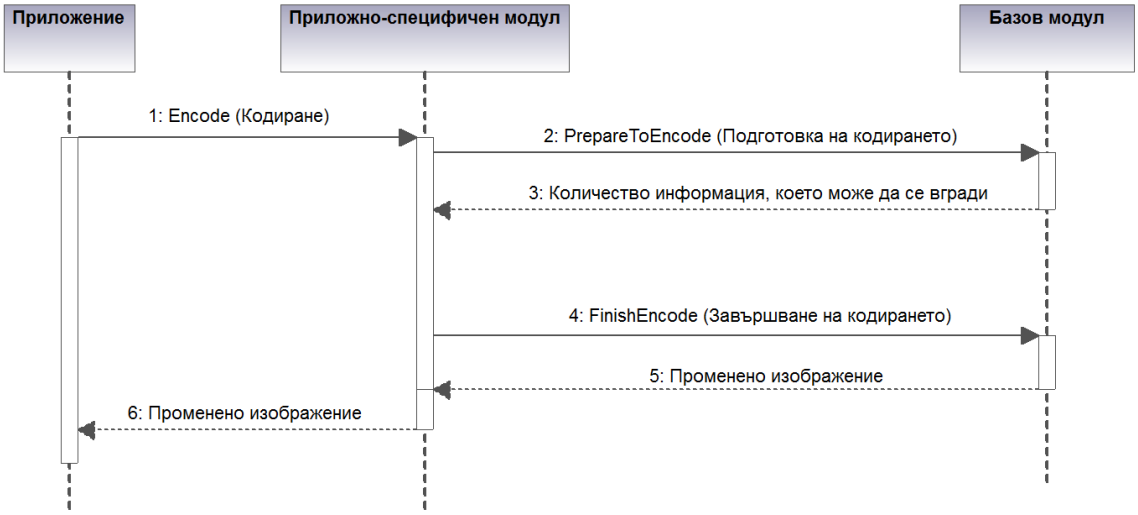


Фиг. 11: Приложно-специфичен модул

Процесът на декодиране на вградени данни се извършва чрез стартирането на метода за декодиране (*Decode*) на приложно-специфичния модул. За разлика от процеса на вграждане, той се състои от два етапа (Фиг. 13):

1. В началото се изпълнява методът за декодиране (*Decode*) на базовия модул, които декодира вградените данни.
2. След това тези данни се обработват от приложно-специфичния модул, който предава крайния резултат от тази обработка на стартиращото го приложение.

Докато базовият и приложно-специфичният модул реализират горните методи както е описано, те могат да придават произволни свойства на методите за криене на данни в мултимедия. Например, класът *DCTHiderEngine* осигурява устойчивост срещу JPEG трансформации, но не може да вгражда големи количества информация. Класът *LSBHiderEngine*, който кодира битовете на данните в най-младшите битове на RGB стойностите на всеки пиксел от изображението, не осигурява устойчивост срещу JPEG компресия, но може да скрие значително по-голямо количество информация с по-малко влошаване на качеството на изображението.

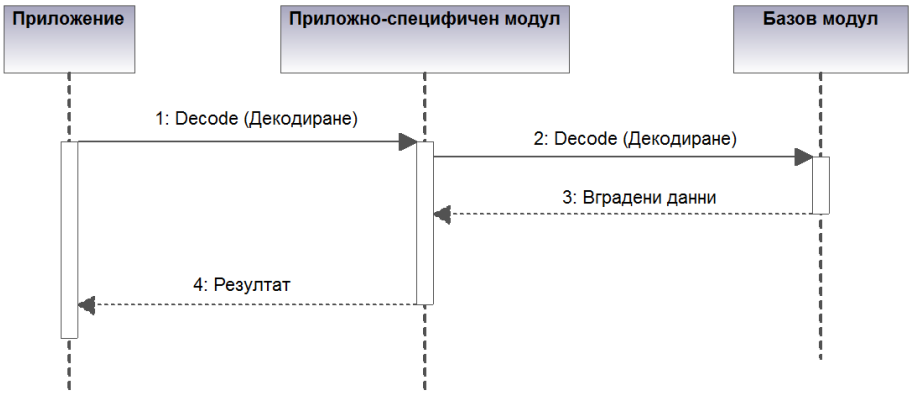


Фиг. 12: Вграждане на скрити данни

Базовите модули могат да се използват в комбинация с различни приложно-специфични модули като *StegoHider* или *WatermarkHider*. Класът *StegoHider* реализира стеганографски приложно-специфичен модул, чиято цел е да вгради максимално количество информация в изображението. Класът *WatermarkHider* реализира приложно-специфичен модул за цифрово маркиране, който открива неправомерни промени в изображението. Разработеният метод може да възстанови вграден цифров воден знак при наличие на значителни по размер промени. От своя страна това налага ограничение в максималния размер (в битове) на водния знак.

Предложената модулна архитектура за реализация на методите за криене на данни позволява създаването и адаптирането на приложно-специфичните модули съобразно променящите се изисквания на потребителите, като в същото време се запазват всички свойства, реализирани от базовия модул. Следователно, базовият модул трябва да пре-

достави основни свойства, които са от голямо значение за всички използвани методи за криене на данни и не се променят. От друга страна, приложно-специфичният модул трябва да предостави специализирани и често пъти по-сложни за реализиране свойства от високо ниво на методите за криене на данни, които са зависими от конкретното приложение и конкретните нужди на потребителя. Свойствата, реализирани от цялостния метод, представляват комбинация от свойствата, осигурени от базовия и от приложно-специфичния модул. Освен това е гарантирано, че методи, използващи един и същ базов модул, разполагат с еднаква реализация на свойствата, които този базов модул предоставя. Това подобрява надеждността на методите и улеснява тяхната употреба. Допълнителни детайли за модулната архитектура на методите са дадени в Глава 4.



Фиг. 13: Декодиране на данни:

3.2 Базов модул, устойчив срещу JPEG трансформации

Базовият модул, представен в този раздел, е разработен, за да позволи създаването на модулни методи, които вграждат информация в изображения, компресирани и запаметени във JPEG формат. Той може да се използва както от стеганографски приложно-специфични модули, така и от приложно-специфични модули за цифрово маркиране.

3.2.1 Основни цели при разработването на модула

Основните цели при разработването на този базов модул са както следва:

1. Да се осигури устойчивост срещу JPEG трансформации: компресия, декомпресия и рекомпресия. Обработката на данните, пряко свързана с JPEG формата трябва да се реализира изцяло (капсулира) в базовия модул. По този начин приложно-специфичните модули остават независими от използвания в Интернет формат за компресия.
2. Да се осигури възможност за работа с произволни изображения и данни за вграждане. За тази цел данните трябва да се третират като произволна поредица от битове.

3. Да се осигури контекстно-независимо прочитане на вградените данни, т.е. без нужда от допълнителна информация, отнасяща се до оригиналното изображение преди криенето на данните. Тези т.нар. „слепи“ методи за криене на данни дават максимална свобода на приложно-специфичните модули по отношение на реализирането на специализираните, зависими от приложението свойства на метода.
4. Да се гарантира безпогрешното прочитане и възстановяване на скритата информация. По този начин приложно-специфичните модули могат да разчитат на надеждното съхранение на информацията в изображението и не е необходимо да реализират от своя страна методи за откриване и корекция на грешки.

Едновременното реализиране на тези четири цели при разработката на базовия модул далеч не е тривиална задача. JPEG е формат, реализиращ загубна (англ. lossy) компресия на изображения [47], и постигането на безпогрешно възстановяване на скритата информация след JPEG трансформации не е рутинна задача. Освен това използването на произволни вграждани данни и изискването за „сляп” метод налагат задълбочено познаване на алгоритмите за JPEG компресия и декомпресия, дефинирани в JPEG стандарта [47] и в описанието на формата JPEG File Interchange Format (JFIF) [48]. Тъй като стандартът не дефинира напълно алгоритъма за JPEG компресия, познаването на основните практически реализации на JPEG е от съществено предимство.

Постигането на горните цели ще позволи едновременното използване на криене на данни и криптография. В зависимост от приложно-специфичните модули, с които този базов модул се комбинира, е възможно вграждането на криптографски сигнатури или друга криптографска информация, генерирана на базата на публични/частни ключове. Тя би могла да се използва например за проверка на автентичността на дадено изображение и идентифициране на фалшифицирани негови региони (вж. раздел 0).

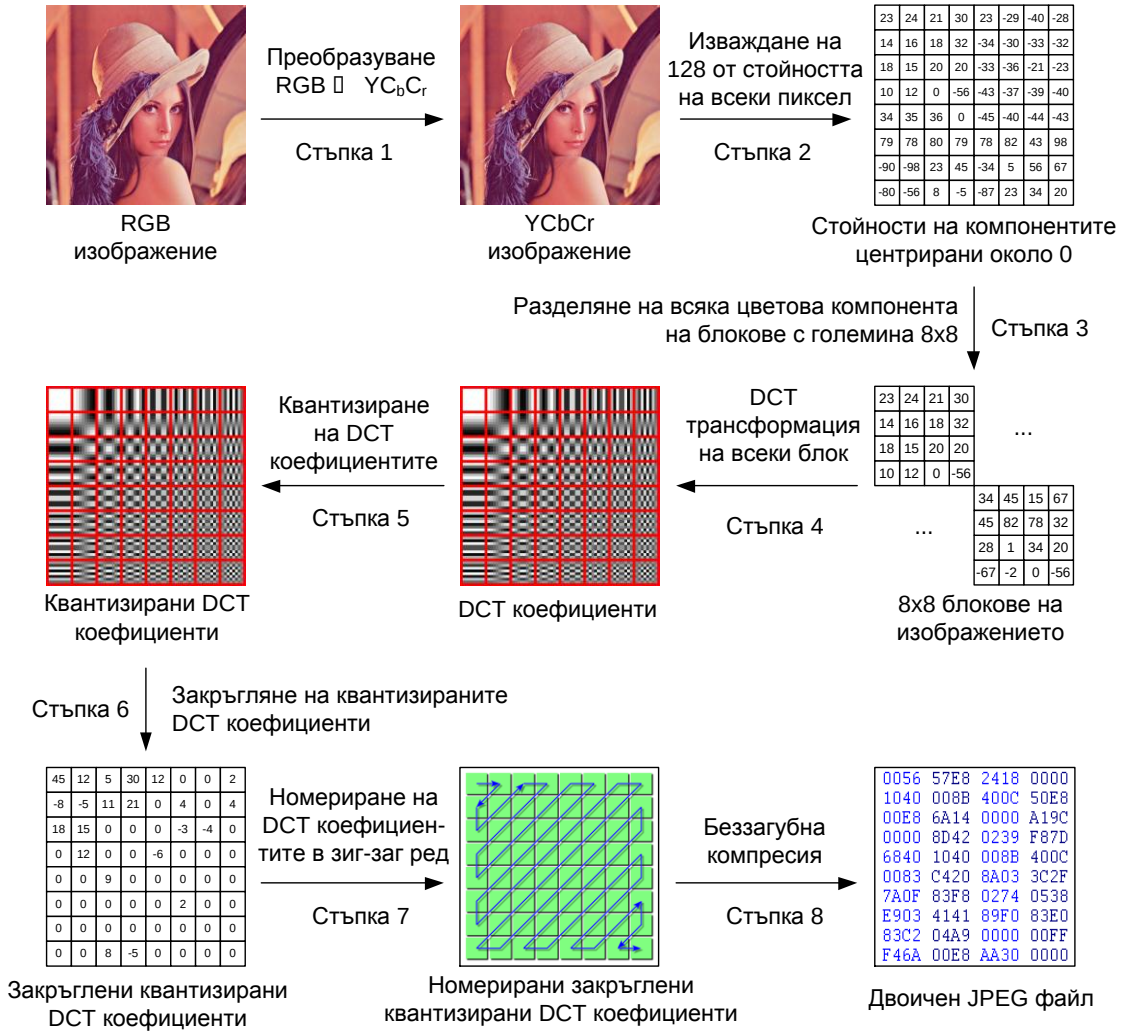
3.2.2 Общ поглед върху стандарта JPEG

В този раздел са представени накратко най-важните особености на стандарта за компресия на изображения JPEG, имащи отношение към методите за криене на данни в мултимедия. Целта е да се подготви и улесни изложението на кодиращите и декодиращите методи, създадени за базовия модул.

Стандартът JPEG е създаден от Joint Photographic Experts Group през 1992г. [47], [101]. Той реализира степени на компресия от около 10:1 и се състои от:

- Няколко етапа на математически трансформации;
- Етап на загубно компресиране;
- Няколко етапа на беззагубно компресиране, базирани на предсказващо (англ. predictive) и Хъфман (англ. Huffman) кодиране, както и на кодиране с дължини на сериите (англ. run-length coding).

Изображението, преди началото на компресията, се представя като матрица от пиксели, която се състои от редове и колони. Всеки елемент на матрицата е пиксел от изображението. Той обикновено се представя като вектор, образуван от три скаларни стойности – най често в т.нар. RGB цветово пространство ([102], стр. 47). Всяка компонента на вектора – червена (R), зелена (G) или синя (B) – е с големина един байт и може да заема стойности от интервала *[0; 255]*. Алгоритъмът за кодиране на изображение в JPEG формат приема тази матрица от пиксели като входни данни и изпълнява върху нея следните трансформационни стъпки (Фиг. 14):



Фиг. 14: Кодиране на изображение в JPEG - общ поглед

1. Преобразуване на стойността на всеки пиксел от RGB цветово пространство в т.нар. YC_bC_r цветово пространство. Y компонентата обозначава интензитета на пикселите, а компонентите C_b и C_r обозначават съответно количеството на син и червен цвят в пикселите. Преобразуването се осъществява посредством следната линейна трансформация:

$$\begin{bmatrix} Y \\ C_b \\ C_r \end{bmatrix} = \begin{bmatrix} 0 \\ 128 \\ 128 \end{bmatrix} + \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.1687 & -0.3313 & 0.5 \\ 0.5 & -0.4187 & -0.0813 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}.$$

Тази стъпка подобрява възприемания от човешкото око резултат от компресията. Човешкото око е значително по-чувствително към интензитета на пикселите, отколкото към техния цвят. Това свойство позволява на JPEG стандарта да компресира цветовите компоненти C_b и C_r в по-голяма степен отколкото интензитета Y, без това да предизвика значително влошаване на възприеманото качество на изображението.

2. Изваждане на константата *128* от всички компоненти на пикселите в YC_bC_r цветовото пространство. По този начин стойностите на пикселите се отместват и заемат интервала $[-128; 127]$, който е центриран около стойността *0*. Тази стъпка подобрява резултата от дискретната косинусова трансформация (DCT).
3. Разделяне на всяка компонента на изображението на блокове с големина 8×8 пиксела. Ако броят на пикселите по хоризонтала или вертикала не се дели без остатък на 8, се добавят фиктивни колони или редове в изображението. JPEG компресията осигурява най-добър резултат тогава, когато стойностите на съседни пиксели в даден блок имат сходни по големина стойности. Средностатистически (но и в зависимост от типа на изображението), повечето блокове с големина 8×8 са съставени от пиксели с приблизително еднакви стойности, което позволява постигането на подобра компресия в сравнение с по-големи размери на блока.
4. Извършване на двумерна DCT трансформация на всеки отделен блок посредством следната математическа формула:

$$A_{k,l} = \frac{c_k \cdot c_l}{4} \sum_{m=0}^{7} \sum_{n=0}^{7} P_{m,n} . cos\left[\frac{(2m+1).k.\pi}{16}\right] . cos\left[\frac{(2n+1).l.\pi}{16}\right],$$

където $P_{m,n}$ обозначава блока от пиксели, $A_{k,l}$ обозначава DCT коефициентите, $c_k = \begin{cases} \frac{1}{\sqrt{2}}, & \text{за } k = 0 \\ 1, & \text{за } k \neq 0 \end{cases}$, $c_l = \begin{cases} \frac{1}{\sqrt{2}}, & \text{за } l = 0 \\ 1, & \text{за } l \neq 0 \end{cases}$ и $k, l, m, n \in [0; 7]$.

DCT трансформацията лежи в основата на JPEG компресиращия алгоритъм. Тя използва подход, подобен на трансформацията на Фурие ([103], стр. 301) и представя изображението като сума от стандартизиран набор честоти, представени чрез косинус функции. Ниските честоти са концентрирани в горния ляв регион на DCT блока, а високите честоти се намират в долния десен регион. DCT трансформацията е подготвителна стъпка към използването на едно важно свойство на човешката зрителна система – тя е значително по-чувствителна към изменения в ниските честоти на дадено изображение, отколкото към изменения във високите честоти. Следователно, високите честоти могат да се компресират до голяма степен или дори да бъдат премахнати от изображението без това да предизвика значително влошаване

на възприеманото качество. Ниските честоти се подлагат на по-малка компресия, което запазва до голяма степен качеството на изображението.

5. Елементите на всеки 8×8 DCT блок се разделят на съответните стойности, зададени в 8×8 квантизираща таблица Q, която се определя в зависимост от фактора на JPEG компресия (скаларна стойност в интервала *[1; 100]*, зададена от потребителя):

$$\mathrm{B}_{k,l} = \mathrm{A}_{k,l} \,/\, Q_{k,l} \text{ , } \mathrm{k,l} \in [0; 7] \text{ ,}$$

където $\mathrm{B}_{k,l}$ обозначава квантизираните DCT коефициенти.

Тази квантизираща стъпка подготвя DCT коефициентите за загубна компресия. Голям квантизиращ елемент $Q_{k,l}$ води до по-голяма степен на компресия, но и до по-голяма загуба на качество. Следователно, елементите на квантизиращата таблица $Q_{k,l}$, намиращи се в нейния горен ляв ъгъл и съответстващи на по-ниските честоти, имат относително малки стойности. Елементите, намиращи се в долния десен ъгъл на таблицата и съответстващи на по-високите честоти, имат относително големи стойности. Стойностите в квантизиращите таблици за различни фактори на компресия не са част от JPEG стандарта и се определят от производителите на програми за създаване на JPEG изображения (англ. JPEG encoders) с помощта на емпирични методи [47].

6. Закръгляване на квантизираните DCT коефициенти във всеки блок:

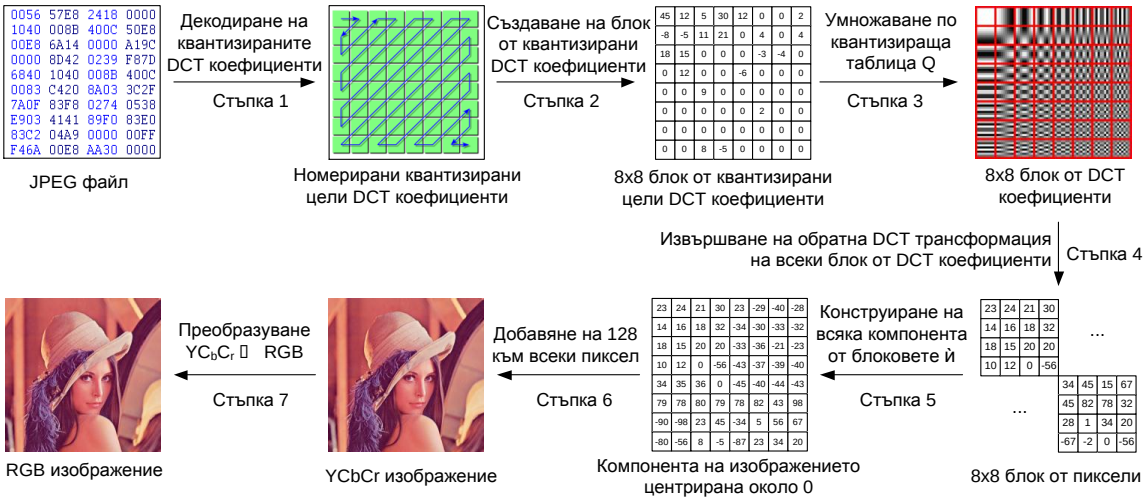
$$\mathrm{C}_{k,l} = \left[B_{k,l}\right] \text{ , } \mathrm{k,l} \in [0; 7] \text{ ,}$$

където $\mathrm{C}_{k,l}$ обозначава закръглените квантизирани DCT коефициенти, а $\left[B_{k,l}\right]$ обозначава закръгляване на $B_{k,l}$ до цяло число.

Това е основната операция, предизвикваща загуба на качество (и следователно загубна компресия) в JPEG стандарта. Закръгляването на квантизираните DCT коефициенти предизвиква необратима загуба на информация. Ефектът от закръгляването върху амплитудите на честотите и следователно върху качеството на изображението зависи от големината на елементите от квантизиращата таблица $Q_{k,l}$.

7. Номериране на закръглените квантизирани DCT коефициенти в ред „зигзаг“, като се започва от горния ляв ъгъл на DCT блока и се завършва при долния десен ъгъл. По този начин DCT коефициентите, съответстващи на ниски честоти получават малки номера, а DCT коефициентите, съответстващи на високи честоти получават големи номера. Тази стъпка цели подобряването на резултата от последващото беззагубно кодиране с дължини на сериите, тъй като много от стойностите на DCT коефициентите, съответстващи на високи честоти са равни на 0, поради големите квантизиращи коефициенти в таблицата Q.
8. Прилагане на беззагубна компресия, използваща предсказващо кодиране, кодиране с дължини на сериите и Хъфман кодиране върху закръглените квантизирани DCT коефициенти.

Алгоритъмът за декодиране на изображение от JPEG формат в матрица от пиксели се състои от обратните стъпки, като се започва от компресирания двоичен JPEG файл като входни данни (Фиг. 15):



Фиг. 15: Декодиране на JPEG изображение - общ поглед

1. Прилагане на беззагубна декомпресия върху двоичния JPEG файл, която дава като резултат номерираните (с линеен пореден номер) квантизирани DCT коефициенти (цели числа), съответстващи на всеки 8×8 блок от изображението.
2. Създаване на двумерен 8×8 DCT блок от номерираните квантизирани DCT коефициенти.
3. Умножаване на квантизираните DCT коефициенти $C_{k,l}$ по съответните елементи от квантизиращата таблица $Q_{k,l}$, използвана при кодирането и запомнена като част от направляващото описание на файла на JPEG формата:
$$A_{k,l} = C_{k,l} \times Q_{k,l}\ ,\ k,l \in [0;7].$$
4. Извършване на обратна двумерна дискретна косинусова трансформация (inverse DCT - IDCT) на всеки DCT блок $A_{k,l}$, получен в предната стъпка, за да се извлекат стойностите на пикселите:
$$P_{m,n} = \sum_{k=0}^{7}\sum_{l=0}^{7} \frac{c_k.c_l}{4}.A_{k,l}.\cos\left[\frac{(2m+1).k.\pi}{16}\right].\cos\left[\frac{(2n+1).l.\pi}{16}\right],$$
където $c_k = \begin{cases} \frac{1}{\sqrt{2}}, & \text{за } k = 0 \\ 1, & \text{за } k \neq 0 \end{cases}$, $c_l = \begin{cases} \frac{1}{\sqrt{2}}, & \text{за } l = 0 \\ 1, & \text{за } l \neq 0 \end{cases}$, $k,l,m,n \in [0;7]$.
5. Конструиране на различните (най-често три на брой) компоненти на изображението от техните съставни 8×8 блокове. Ако са били добавени фиктивни колони или редове в изображението, на този етап те се отстраняват (реалната дължина и височина на изображението в пиксели е запомнена в направляващото описание на JPEG файла).

6. Добавяне на константата *128* към всички компоненти на изображението, за да се премахне центрирането около *0* и стойностите на пикселите да заемат интервала $[0; 255]$.
7. Преобразуване на стойността на всеки пиксел, зададен чрез компонентите на YC_bC_r цветовото пространство, в RGB цветовото пространство посредством следната линейна трансформация:

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1.402 \\ 1 & -0.34414 & -0.71414 \\ 1 & 1.772 & 0 \end{bmatrix} \begin{bmatrix} Y \\ C_b - 128 \\ C_r - 128 \end{bmatrix}.$$

Този базов модул на модулните методи за криене на данни включва модел на JPEG стандарта за компресия до стъпката на прилагане на беззагубна компресия върху закръглените квантизирани DCT коефициенти. Алгоритъмът за вграждане на данни оперира основно върху квантизираните DCT коефициенти, както е обяснено в следващите раздели.

3.2.3 Базов модул: кодиране

Кодирането на информацията в базовия модул се състои от два етапа:

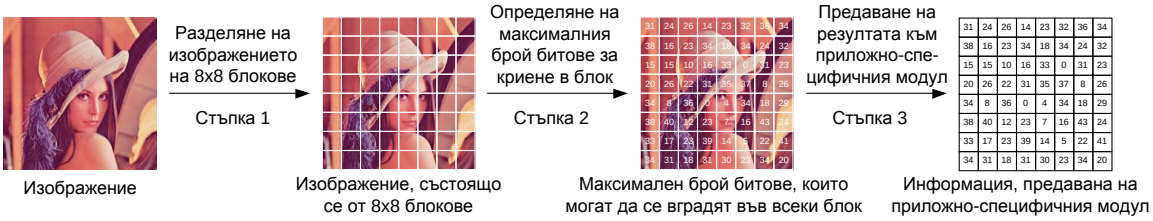
1. Етап на подготовка (*PrepareToEncode*) и
2. Етап на завършване (*FinishEncode*) на кодирането (вж. раздел 3.1).

Етапът на подготовка преминава през следните стъпки (Фиг. 16):

1. Изображението се разделя на същите 8×8 блокове от пиксели, които са характерни за JPEG стандарта. Също така се изчислява броят на блоковете от пиксели по хоризонтала и вертикала както следва:
$$n_x = d_x / b_x \, , n_y = d_y / b_y \, ,$$
където n_x и n_y обозначават съответно броя блокове по хоризонтала и вертикала, d_x и d_y обозначават съответно ширината и височината на изображението, а b_x и b_y обозначават размерите на блока по хоризонтала и вертикала. За размерите на този базов модул важи: $b_x = b_y = 8$.
2. Следващата задача на етапа на подготовка е да определи максималното количество информация (в битове), което може да бъде скрито във всеки блок от пиксели. Информацията се вгражда в закръглените квантизирани DCT коефициенти $C_{k,l}$. Нейното максимално количество може да зависи от различни характеристики на блока: текстура, брой на цветовете в блока, наличие и посока на градиентни преходи и др.

   JPEG стандартът премахва голяма част от високите честоти в дадено изображение и запазва до голяма степен ниските честоти. Това означава, че повечето DCT коефициенти, които са индексирани с голям номер след стъпката на номериране на

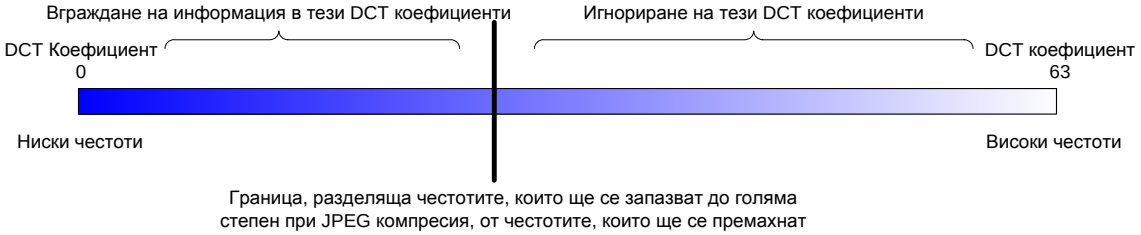
коефициентите (и които съответстват на високи честоти в изображението), ще бъдат премахнати, докато DCT коефициентите, които са индексирани с малък номер ще присъстват в изображението след JPEG компресия.



Фиг. 16: Базов модул – етап на подготовка (*PrepareToEncode*)

Алгоритъмът, разработен в дисертацията, използва това свойство и определя граница в поредицата от *64* номерирани DCT коефициента на даден блок. Тази граница разделя честотите в изображението, които ще се запазя до голяма степен, от тези които ще се премахнат (Фиг. 17). Информацията се крие само в DCT коефициентите, които имат номер, по-нисък или равен на номера, при който е поставена границата.

Границата се определя динамично в зависимост от характеристиките на блока от изображението. Тя обикновено се поставя между DCT коефициентите с номера 30 и 45, като се позволява вграждане на данни във всички DCT коефициенти, имащи по-малки номера (DCT коефициентите вляво от границата на Фиг. 17). Това означава, че ако се вгражда един бит във DCT коефициент, то тогава могат да се вграждат между 30 и 45 бита на блок в зависимост от изображението. Това максимално количество на вграждани битове се изчислява отделно за всеки блок.

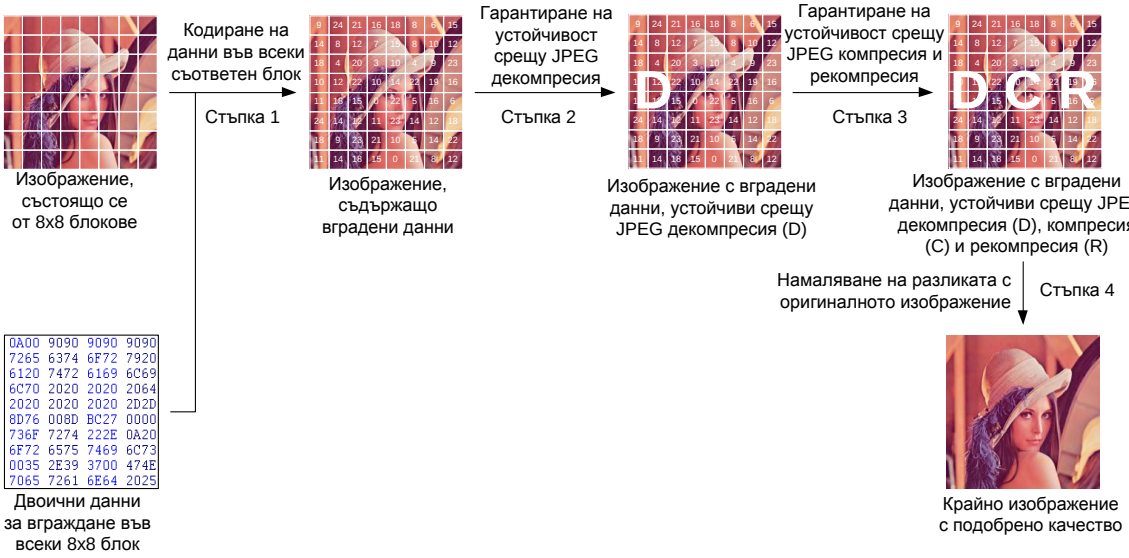


Фиг. 17: DCT коефициенти - подбор

3. Броят на блоковете от пиксели по хоризонтала и вертикала, изчислен в стъпка *1* и максималното количество битове за вграждане във всеки блок, определено в стъпка *2*, се предават на приложно-специфичния модул.

Етапът на завършване на кодирането (Фиг. 18) приема като входни данни (от страна на приложно-специфичния модул) конкретните битове, които трябва да бъдат

вградени във всеки блок от изображението. Стойностите на битовете са произволни, като единственото ограничение засяга броя на битовете, вграждани в даден блок. Този брой трябва да бъде по-малък или равен на максималния брой битове, които могат да бъдат вградени в блока, както е определено от етапа на подготовка на кодирането. Ако това ограничение е изпълнено (т.е. данните за вграждане не са твърде много за дадения блок), то алгоритъмът изпълнява следните стъпки:



Фиг. 18: Базов модул – етап на завършване на кодирането (*FinishEncode*)

1. Кодиране на стойностите на битовете в най-младшите битове на квантизираните DCT коефициенти на блока, като се започва с DCT коефициента с пореден номер едно. По време на кодирането се взема предвид необходимостта да се минимизира ефекта от промените на квантизираните DCT коефициенти и да се подобри общото качество на изображението – т.е. да се минимизира разликата между оригиналното изображение и изображението, съдържащо вградените данни. Използват се техники за Модулация на Индекса на Квантизация (англ. Quantized Index Modulation, QIM, вж. раздел 2.2.1).
2. Гарантиране на устойчивост на вградените данни срещу JPEG декомпресия (описана по-подробно в раздел 3.2.4).
3. Гарантиране на устойчивост на вградените данни срещу компресия и рекомпресия (описани по-подробно в раздел 3.2.4).
4. Намаляване на разликата между оригиналното изображение и изображението, съдържащо вградените данни. Предните стъпки са разработени така, че да променят изображението в колкото е възможно по-малка степен. Въпреки това, те внасят неизбежни промени в стойностите на пикселите след кодирането на данните. Тези промени могат допълнително да бъдат редуцирани посредством лека промяна или

добавяне на подходящи DCT коефициенти след завършването на кодирането, като по този начин качеството на крайното изображение се повишава.

Стъпките за вграждане на данни, описани по-горе, се изпълняват за всеки един блок от изображението. След като алгоритъмът приключи, крайното изображение, съдържащо вградените данни, се предава на приложно-специфичния модул.

3.2.4 Гарантиране на устойчивост срещу JPEG компресия, декомпресия и рекомпресия

Постигането на устойчивост на вградените данни срещу JPEG трансформации не е тривиална задача поради предвидените в JPEG стандарта степени на свобода при разработката на JPEG енкодери и декодери, както и поради целочислените програмни реализации на цветовите пространства и математическите трансформации. Поради тази причина освен JPEG стандарта е необходимо и много добро познаване на по-популярните конкретни JPEG реализации, тъй като е невъзможно да се предвиди кои JPEG програмни библиотеки ще бъдат използвани при евентуална обработка на изображенията [48].

Вграждането на данни в най-младшите битове на квантизираните DCT коефициенти ни дава само по себе си устойчивост срещу JPEG компресия, съпроводена със сравнително добро качество на изображението при сравнително голям размер на вградените данни [6]. Този начин на вграждане е достатъчен за вероятностна оценка на присъствието на предварително определени данни или сигнатури в изображението, но е съпроводен с проблеми, ако непознати вградени данни трябва да бъдат възстановени с абсолютна точност от изображението. Този проблем може да се облекчи, ако се използват кодове за корекция на грешки. Те обаче не могат да гарантират стопроцентово надеждно възстановяване на вградените данни, ако не се вземат специални мерки за предотвратяване на загубата на вградени данни при декомпресия и рекомпресия на изображението.

Декомпресията на изображение от JPEG до матрица от RGB пиксели винаги включва грешки, породени от две основни причини:

1. Първата причина е свързана с грешки от закръглявания при целочислените програмни реализации на JPEG трансформациите (стъпки 1 – 4, раздел 3.2.2).
2. Втората причина е свързана с ограниченото представяне на цветовите пространства – често подмножества на множеството на естествените числа $S = \{0,1,2, \dots ,255\}$.

Целочислената програмна реализация на JPEG трансформациите (крайна точност на всички променливи) води до малки грешки от закръгляване, които в някои случаи водят до опасност от обръщане на стойността на даден бит от вградените данни. Тази ситуация може да бъде разрешена сравнително лесно чрез пресмятането на поредица

от прави и обратни трансформации (стъпки 1 до 3 от алгоритмичното описание по-долу). Коефициентите $C_{k,l}$, които достигат до стъпка 4 не са съпроводени от проблеми, причинени от грешки от закръглявания.

Ограниченото целочислено представяне на цветовите пространства е причината за по-сериозен проблем, свързан със загубната компресия. Тя постига намаляване на информацията чрез изобразяване на множество сходни по стойности блокове от пиксели в един и същи DCT блок. Обратното изображение е уникално. Ако някои от оригиналните стойности на пикселите преди компресията имат стойности, близки до границите на множеството *S*, то някои стойности на пикселите, получени след декомпресия, биха могли да попаднат извън границите на *S*. Повечето програми за обработка на изображения променят тези стойности на 0 или 255. В резултат на това променениият блок от пиксели би могъл да бъде изобразен в друг DCT блок, което води до загуба на вградените данни. Нашето решение е да се скалират предварително стойностите на пикселите, така че обратното изобразяване при декомпресията винаги да дава валидни стойности (стъпки 4 и 5 от алгоритмичното описание по-долу).

Коефициентът на скалиране α се задава емпирично в стъпка 0. Ако няма разлика в реконструираните стойности на пикселите (в стъпка 5 от алгоритмичното описание), то α може да се увеличи динамично с малка стойност. Това не се оказа необходимо при провеждането на нашите експерименти (вж. Глава 5). Чрез провеждането на множество итерации се постига фин контрол на качеството на изображението и стойностите на пикселите се променят във възможно най-малка степен, необходима за постигането на устойчивост срещу JPEG декомпресия.

Нека сега разгледаме устойчивостта на вградените данни срещу JPEG рекомпресия. JPEG стандартът не дефинира JPEG квантизиращите таблици, отговорни за загубната компресия, и вследствие от това те са различни при отделните производители на програми за обработка на изображения като Adobe, Microsoft, Corel, и др. На този етап се гарантира устойчивостта срещу рекомпресия с $\forall Q^{(j)} | Q_{k,l}^{(j)} \leq Q_{k,l}$ за $\forall k, l \in \{0,1,2, \dots ,7\}$, където j е индекс на итерация, покриващ всички комбинации от квантизиращи таблици $Q^{(j)}$, които биха могли да се използват от програмните системи, а квантизиращата таблица Q се определя в зависимост от избраното от потребителя ниво на JPEG компресия.

Първоначалните експерименти показват, че вградените данни са сравнително устойчиви срещу JPEG рекомпресия, като устойчивостта зависи от *Q*, количеството вградени данни и текстурите на изображението. Във всички случаи, ако не се вземат допълнителни мерки, вградените данни не могат да бъдат възстановени надеждно след рекомпресията. За да се гарантира надеждно възстановяване на данните, в този базов модул се използва изменена техника на Модулация на Индекса на Квантизация [84]. Посредством квантизиращата стъпка $q_{k,l}$ могат да се изберат множество нови стойности $C_{k,l}^{(i)}$ на всеки един коефициент $C_{k,l}$, които кодират идентични вградени данни (стъп-

ки 7 и 8 от алгоритмичното описание по-долу). Коефициентът β приема първоначална стойност 1. Ако за дадено $Q^{(j)}$ и $\forall(k,l) \in Z_{data}$ е изпълнено $C_{k,l}^{(i)} = E_{k,l}^{(i,j)}$, то целият JPEG блок се приема за устойчив срещу рекомпресия с $Q^{(j)}$. Изборът и броят на квантизиращите таблици $Q^{(j)}$ зависи от програмните системи и библиотеки, които се използват за генериране на JPEG изображение – например библиотеката с отворен код IJG, програмните системи Adobe Photoshop, Paint Shop Pro, и др. На този етап, прототипната реализация на базовия модул използва всички възможни $Q^{(j)}$ с цел по-доброто изследване на предложения метод.

За постигане на по-добра оптимизация на бързодействието и евентуално по-добро качество на изображението (измервано като минимална средна квадратична грешка между оригиналното изображение и изображението, съдържащо вградени данни), отделните цикли на итерация (стъпки 1 до 5 и 6 до 9 от алгоритмичното описание) биха могли да бъдат обединени в един единствен итерационен цикъл. Последно подобряване на качеството на изображението би могло да се постигне също така чрез поетапната замяна на някои от квантизираните DCT коефициенти със съответните необработени коефициенти $B_{k,l}$. След това, изображението се реконструира и предава обратно на потребителя.

Алгоритмично описание:

Нека приемем, че коефициентите $C_{k,l}$ вече са пресметнати и данни са вградени в някои от тях. Нека също така дефинираме множеството $Z_{data} = \{(k,l)|C_{k,l} \text{ съдържа вградени данни}\}$, $k,l \in \{0,1,2,\dots,7\}$. Тогава алгоритъмът за постигане на устойчивост срещу JPEG трансформации може да се представи посредством следните стъпки:

0. Определят се първоначалните стойности: $i = 1, u^{(0)} = 0, v^{(0)} = 255, \alpha = 1$.
1. Пресмята се нов блок от пиксели $P_{m,n}^{(1)}$ от блока $C_{k,l}$ чрез извършването на стъпки 3 до 7 от алгоритъма за декодиране на изображение от JPEG формат (раздел 3.2.2).
2. Пресмята се нов блок DCT коефициенти $C_{k,l}^{(1)}$ от блока $P_{m,n}^{(1)}$ чрез извършването на стъпки 1 до 6 от алгоритъма за кодиране на изображение в JPEG формат (раздел 3.2.2).
3. Ако $\exists(k,l) \notin Z_{data}|C_{k,l} \neq C_{k,l}^{(1)}$, то за $\forall(k,l) \notin Z_{data}$ се присвоява $C_{k,l} = C_{k,l}^{(1)}$ и се преминава на стъпка 1.
4. Пресмята се нов блок от пиксели $P_{m,n}^{(2)}$ от блока $C_{k,l}$ чрез извършването на стъпки 3 до 7 от алгоритъма за декодиране на изображение от JPEG формат (раздел 3.2.2).
5. Ако $\exists(m,n)|P_{m,n}^{(2)} \notin \{u^{(0)}, u^{(0)}+1, \dots, v^{(0)}\}$, то за $\forall(m,n)|P_{m,n}^{(2)} < u^{(0)}$ се пресмята:

$$s^{(i)} = \max_{(m,n)|P_{m,n}^{(2)} < u^{(0)}} \left(\alpha \left| P_{m,n}^{(2)} - u^{(0)} \right|\right)$$

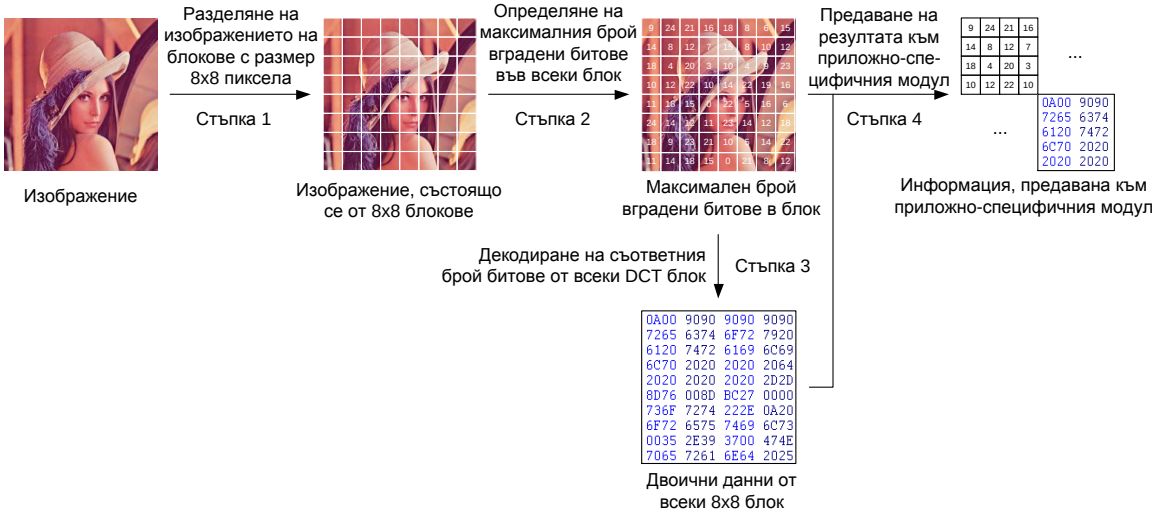
и за $\forall(m,n)|P_{m,n}^{(2)} > v^{(0)}$ се пресмята:

$$t^{(i)} = \max_{(m,n)|P_{m,n}^{(2)} > v^{(0)}} \left(\alpha \left| P_{m,n}^{(2)} - v^{(0)} \right|\right)$$

Пресмятат се: $u^{(i)} = u^{(i-1)} + s^{(i)}$ и $v^{(i)} = v^{(i-1)} - t^{(i)}$. За $\forall(m,n)|P_{m,n} < u^{(i)}$ се присвоява $P_{m,n} = u^{(i)}$ и за $\forall(m,n)|P_{m,n} > v^{(i)}$ се присвоява $P_{m,n} = v^{(i)}$. Увеличава се i с 1 и се преминава на стъпка 1.

6. Присвоява се $i = 0$, $\beta = 1$, избира се валиден индекс $j|\exists Q^{(j)}$ и за $\forall(k,l)$ се присвоява $C_{k,l}^{(0)} = C_{k,l}$.
7. Пресмятат се $D_{k,l}^{(i,j)} = \left[C_{k,l}^{(i)} \times Q_{k,l}/Q_{k,l}^{(j)}\right]$ и $E_{k,l}^{(i,j)} = \left[D_{k,l}^{(i)} \times Q_{k,l}^{(j)}/Q_{k,l}\right]$.
8. Ако $\exists(k,l) \in Z_{data}|C_{k,l}^{(i)} \neq E_{k,l}^{(i,j)}$, то се пресмята $C_{k,l}^{(i+1)} = C_{k,l}^{(i)} + (-1)^i\left[(i+1) \times q_{k,l}\right]$, където $q_{k,l} = \beta \times 2^{N_{k,l}}$ и $N_{k,l}$ е броят битове, съдържащ се в коефициента $C_{k,l}$. Увеличава се i с 1 и се преминава на стъпка 7.
9. Повтарят се стъпки 6 до 8 за $\forall(k,l) \in Z_{data}$ и $\forall j|\exists Q^{(j)}$.

3.2.5 Базов модул: декодиране



Фиг. 19: Базов модул – декодиране (*Decode*)

За разлика от процеса на кодиране, процесът на декодиране (*Decode*) се състои само от един етап, който е подобен на етапа на подготовка (*PrepareToEncode*) при кодирането (Фиг. 19):

1. Разделяне на входното изображение на блокове с размер 8×8 пиксела. Броят на блоковете в хоризонтална и вертикална посока, както и размерът на блока се предават на приложно-специфичния модул.

2. Определяне на максималния брой битове, вградени във всеки блок, чрез алгоритъма, описан в раздел 3.2.3 (Фиг. 17).
3. Декодиране на съответния брой битове от всеки DCT блок чрез прочитане на най-младшите битове на квантизираните DCT коефициенти на блока $C_{k,l}$ (вж. раздел 3.2.2), като се започва от DCT коефициента с пореден номер едно.
4. След обработката на всички блокове от изображението, декодираните битове се предават към приложно-специфичния модул.

На това място трябва да се отбележи, че процесите на кодиране и декодиране се състоят от различен брой стъпки, които се нуждаят от различно време за изпълнение. Отнемащите най-много време стъпки в процеса на кодиране са част от етапа на завършване на кодирането (*FinishEncode*) и нямат еквивалент в процеса на декодиране. Следователно, процесът на декодиране (*Decode*) се нуждае от значително по-малко време за изпълнение в сравнение с пълния процес на кодиране, съставен от етапите на подготовка (*PrepareToEncode*) и завършване на кодирането.

3.3 Приложно-специфичен модул за целите на стеганографията. Модулен стеганографски метод

Представеният приложно-специфичен модул за целите на стеганографията е разработен, за да скрие максимално количество информация в дадено изображение. Освен това основна негова цел е и да минимизира промените в изображението, причинени от процеса на криене на данни. По този начин различни видове конфиденциална информация могат да бъдат предавани през Интернет или обработвани от Интернет-базирани приложения и програми.

Важно е да се отбележи, че комбинацията от приложно- специфичен и базов модул формира цялостен модулен метод за криене на данни в мултимедия, който наследява и комбинира всички характерни за съставните модули свойства. Приложно-специфичният модул за целите на стеганографията придава на методите, в които в включен, следните две нови свойства:

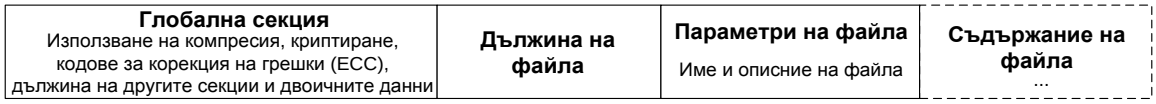
1. Първото свойство е възможността за корекция на неочаквани грешки във вградените данни чрез прилагане на кодове за корекция на грешки (англ. error-correcting codes – ECC). Това е полезно в случай на малки неочаквани промени в изображението или ако JPEG файла бъде повреден по време на запис/споделяне в Интернет.
2. Второто свойство е подобрената сигурност на метода посредством използването на генератори на псевдослучайни числа за разбъркване на информацията. Началните стойности на генераторите, използвани при кодирането на данните, са необходими за тяхното декодиране и могат да бъдат направени зависими от потребителски-дефинирани пароли.

Модулният стеганографски метод работи с произволни двоични файлове с данни като вход. Той добавя към всеки такъв файл свое собствено направляващо файлово описание (англ. header), описано в следващия раздел. Методът е контекстно независим („сляп", англ. blind) – той не се нуждае от никакъв вид допълнителна информация, за да прочете и възстанови вградените данни, което позволява неговото използване в разнообразни Интернет-базирани сценарии.

3.3.1 Направляващо файлово описание

Направляващото файлово описание съдържа информация относно обработвания файл с данни и някои важни параметри, контролиращи метода за криене на данни. То е разделено на няколко отделни секции (Фиг. 20). Първата секция е глобална и съдържа информация за употребата на компресия, криптиране или кодове за корекция на грешки, както и за дължината на другите секции и двоичните данни. Втората секция съдържа дължината на файла. Третата секция съдържа информация относно името на файла и кратко описание.

Дължината на различните секции (и следователно общата дължина на направляващото описание на файла) е променлива и зависи от индивидуалните характеристики на файла (дължина, име и др.). Приложението или неговите потребители имат известна степен на контрол върху направляващото описание. Те могат да изберат дали ще бъдат прилагани компресия, криптиране, ECC-алгоритми и дали името и описанието на файла ще бъдат пропуснати, ако не са необходими.



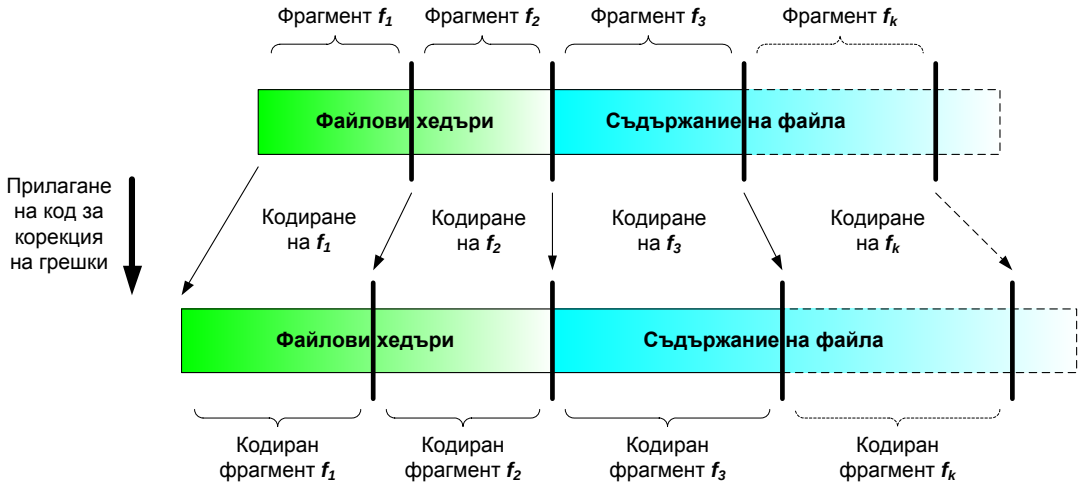
Фиг. 20: Модулен стеганографски метод – направляващо файлово описание

В тази глава няма да се дискутира в подробности прилагането на алгоритми за компресия или криптиране, тъй като то не влияе непосредствено върху същността на разработка на методите за криене на данни. На това място само ще се отбележи, че съществуват редица наложили се алгоритми за компресия (zip, lzma, bzip, gzip, Huffman, lzw, 7zip и др. [104], [105]) и криптографски алгоритми (AES, 3DES, Blowfish, Serpent и др. [20], [21]). Всеки един от тях може да се използва при предварителната обработка на файла, като е желателно компресията да се извърши преди криптирането за реализиране на по-добра ефективност.

Употребата на кодове за корекция на грешки е по избор и повишава устойчивостта на вградените данни срещу непредвидени промени в изображението. Това предимство е за сметка на лекото увеличаване на количеството данни за вграждане, което от своя страна води до леко влошаване на качеството на изображението.

3.3.2 Кодове за корекция на грешки

Кодовете за корекция на грешки повишават устойчивостта на скритата информация срещу малки неочаквани промени в изображението. Съществуват различни варианти на такива кодове: Хеминг (англ. Hamming) кодове, Рийд-Соломон (англ. Reed-Solomon) кодове, турбо кодове и др. Всеки един от тях има различни свойства и може да открие и коригира различно количество грешки в дадена поредица от двоични данни [106], [107]. По-сложните кодове увеличават вероятността за откриване и корекция на грешки, но водят до по-голямо увеличение на данните.



Фиг. 21: Кодове за корекция на грешки

Кодовете за корекция на грешки, използвани в този метод за криене на данни в мултимедия, се базират на алгоритъма на Хеминг и позволяват надеждното откриване и корекция на грешки с дължина един бит в определен сегмент от двоичните данни. Важно е да се подчертае, че тяхната употреба е по избор и зависи от настройките на приложението, което използва метода за криене на данни. Независимо от употребата на такива кодове базовият модул, разработен в тази дисертация, гарантира устойчивостта на данните срещу JPEG трансформации.

Алгоритъмът на Хеминг работи с поток/поредица (англ. stream) от номерирани битове. Той добавя нови, допълнителни битове за корекция на грешки на всички позиции с пореден номер, равен на $2^k, k \in [0; \infty)$. По време на това добавяне, съществуващите битове с данни се изместват надясно към по-големите поредни номера, за да направят място на битовете за корекция на грешки. Техните поредни номера се увеличават и общата дължина на потока от битове нараства (Фиг. 21).

Стойността на всеки бит за корекция на грешки се определя посредством функция на четност (англ. parity function), чиито аргументи се състоят от стойностите на битове, намиращи се на строго определени позиции в потока. Умелият подбор на тези позиции

позволява откриването на точното местоположение на произволен сгрешен бит в потока. За да се установи наличието и позицията на сгрешен бит в потока, битовете за корекция на грешки се изваждат от потока и се поставят един до друг в реда на тяхното номериране. Разгледани заедно, тези битове формират числена стойност, която е равна на нула, ако не са намерени грешки и е използван четен паритет (англ. even parity). Ако стойността е по-голяма от нула, тя обозначава позицията на сгрешен бит в потока. За да се коригира грешката, стойността на бита, заемащ обозначената позиция, трябва да се обърне.

Алгоритъмът за корекция на грешки се изпълнява след като направляващото файлово описание, дискутирано в предишния раздел, се добави към същинското съдържание на файла (Фиг. 21).

Образуваният поток от битове f се разделя на фрагменти f_k с дължина l_{f_k}. В този конкретен случай е избрана стойност $l_{f_k} = 120$ за $\forall k$, която съответства на максимално ефективното използване на 7 бита за корекция на грешки във фрагмента, като общата дължина на f_k след кодирането е 127 бита. Специално внимание се обръща на това да не се смесват направляващото описание и съдържанието на файла в един и същ фрагмент. Всеки един фрагмент се кодира самостоятелно, използвайки алгоритъма на Хеминг. Накрая отделните фрагменти се събират заедно, за да образуват нов поток от двоични данни, устойчиви срещу малко на брой, разпръснати по дължината на потока промени на битовете.

Алгоритмично описание:

Нека с $f_k^{(i)}$ се обозначи стойността на бита на позиция $i, i \in \{0,1,2,\dots,l_{f_k}-1\}$ във фрагмент k от потока на битове f. Нека за опростяване на описанието приемем, че $l_{f_k} = l_f = 120$ за $\forall k$, и $\sum_{k=1}^{n_f} l_{f_k} = n_f l_f$ е дължината на целия поток от битове, където n_f е броят фрагменти, на които потокът от битове е разделен. Нека с $h = 7$ се обозначи броят на битовете за корекция на грешки в един фрагмент. Нека с $P_{f_k}^M = P(f_k^{(i_1)}, f_k^{(i_2)}, \dots f_k^{(i_m)})$ се обозначи функцията на четност на битовете на позиции $i_1, i_2, \dots, i_m$ във фрагмент f_k, като $M = \{i_1, i_2, \dots, i_m\}$ е множеството, съдържащо тези позиции. По дефиниция: $P(0) = 0$, $P(1) = 1$, $P\left(0, f_k^{(i_1)}, f_k^{(i_2)}, \dots f_k^{(i_m)}\right) = P\left(f_k^{(i_1)}, f_k^{(i_2)}, \dots f_k^{(i_m)}\right)$, $P\left(1, f_k^{(i_1)}, f_k^{(i_2)}, \dots f_k^{(i_m)}\right) = \overline{P\left(f_k^{(i_1)}, f_k^{(i_2)}, \dots f_k^{(i_m)}\right)}$, където $\overline{0} = 1$ и $\overline{1} = 0$. Изпълняват се следните стъпки:

0. Определя се първоначалната стойност на k: $k = 1$.
1. Дефинира се фрагмент g_k с дължина $l_g = l_f + h$. Дефинира се $j = 1$.

2. За $\forall z | z \in \{2^j, 2^j + 1, \dots, 2^{j+1} - 2\}$ се присвоява $g_k^{(z)} = f_k^{(z-1-j)}$. Ако $j < h - 1$, то j се увеличава с 1 и се преминава към стъпка 1. В противен случай се продължава със стъпка 3.
3. Дефинира се $b = 0$.
4. Дефинира се $c = 2^b$. Присвоява се $g_k^{(c-1)} = 0$. Дефинира се множеството $M \subseteq \{0,1, \dots, l_g - 1\}$, съдържащо като елементи следните позиции на битове в g_k: $M = \{c - 1, c, c + 1 \dots, 2c - 2, 3c - 1, 3c, \dots, 4c - 2, \dots, (2d + 1)c - 1, (2d + 1)c, \dots, (2d + 2)c - 2, \dots, l_g - 1\}$, където $d \in \mathbb{N}$.
5. Присвоява се $g_k^{(c-1)} = P_{g_k}^M$. Ако $b < h - 1$, то b се увеличава с 1 и се преминава към стъпка 4. В противен случай се продължава със стъпка 6.
6. Ако $k < n_f$, то k се увеличава с 1 и се преминава към стъпка 1.

Разделянето на потока от битове на фрагменти води до по-голяма гъвкавост на прилагането на алгоритъма за корекция на грешки. В представения вариант може да се възстанови успешно един произволен сгрешен бит на фрагмент. Увеличаването на броя на фрагментите води до по-добри възможности за корекция на грешки. Също така използваният алгоритъм за корекция на грешки може да се различава за различните фрагменти. Изборът на алгоритъм може да бъде мотивиран от ролята и значението на всеки фрагмент в потока, от статистическите характеристики на потока или JPEG изображението, както и от други критерии, дефинирани от потребителите.

3.3.3 Разбъркване на данните

Друга незадължителна стъпка, която може да се изпълни след прилагането на алгоритъм за корекция на грешките и преди същинското кодиране на двоичните данни в JPEG изображението (етапът на завършване на кодирането в базовия модул) е стъпката на разбъркване на данните.

Разбъркването на данните използва генератор на псевдослучайни числа [108] за създаването на псевдослучайни пермутации. Тези пермутации се използват при избора на блок от изображението, в който ще бъде скрит всеки следващ пореден бит на потока от данни. Крайният резултат е псевдослучайно разпределение на вгражданите данни в рамките на изображението.

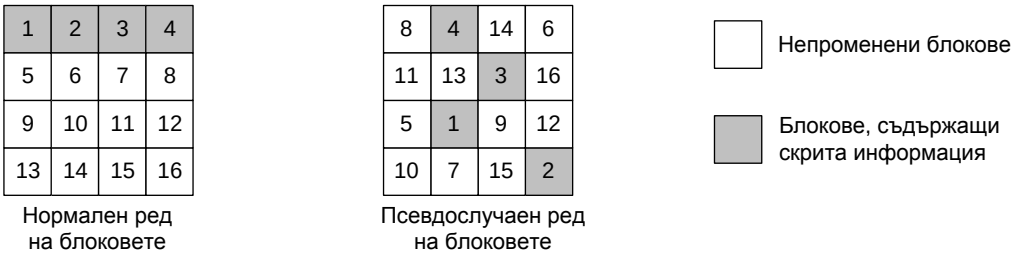
Точният тип на генератора на псевдослучайни числа не е от голямо значение, но е важно неговите начални стойности (англ. seed) да определят еднозначно генерираната поредица от псевдослучайни числа. Един примерен генератор на псевдослучайни числа, предложен от Марсалия (англ. Marsaglia) [109] е т.нар. генератор с обратно умножение с пренасяне (англ. complimentary-multiply-with-carry generator, CMWC) и е представен под формата на C код на Фиг. 22. Началните стойности на генератора се съдържат във вектора *Q*.

```
static unsigned long Q[4096], c=123;

unsigned long CMWC_4096(void){
        unsigned long long t, a = 18782LL;
        static unsigned long i = 4095;
        unsigned long x, m = 0xFFFFFFFE;
        i = (i + 1) & 4095;
        t = a * Q[i] + c;
        c = (t >> 32);
        x = t + c;
        if(x < c){ x++;c++; }
        return (Q[i] = m - x);
}
```

Фиг. 22: Генератор на псевдослучайни числа CMWC_4096, предложен от Марсалия

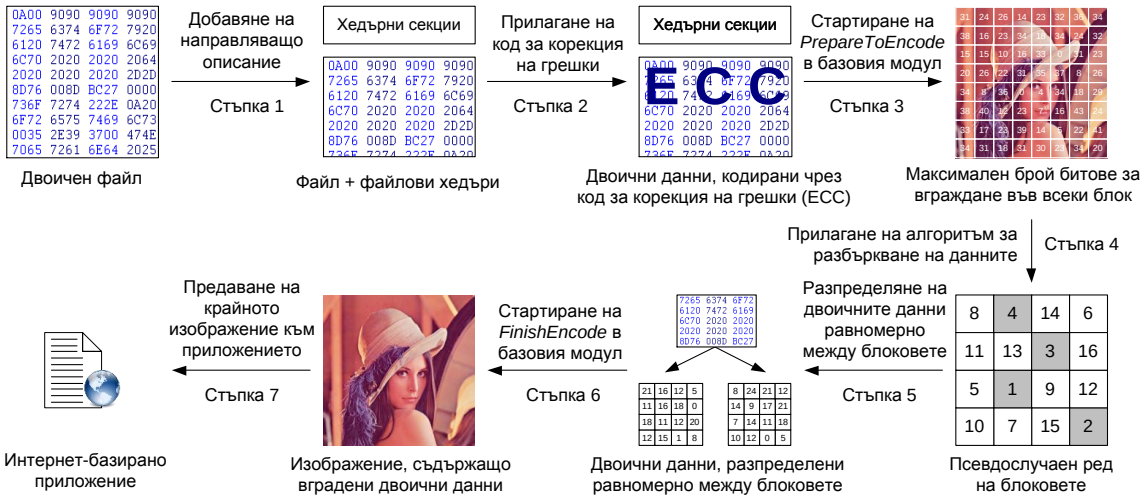
Една от важните причини за включването на стъпката за разбъркване на данните като част от метода касае сигурността на вградените данни. Началните стойности на генератора на псевдослучайни числа могат да са зависими от парола, дефинирана от потребителя, както и от дадени характеристики на изображението, което води до повишаване на нивото на сигурност на предлагания стеганографски метод.



Фиг. 23: Псевдослучаен ред на блоковете от изображението

Друга важна причина е подобрението на общото възприемано качество на изображението. Ако блоковете, в които се вграждат данни, не са поредни, а са разпръснати в изображението, то не съществува област от изображението, в която качеството да се влошава повече, отколкото в останалите области. (Фиг. 23). Ако не се прилага разбъркване на данните, блоковете, съдържащи вградени данни, биха следвали естествения блоков ред, започвайки от най-горната част на изображението, движейки се от ляво на дясно. Това би довело до влошаване на възприеманото качество в горната лява част на изображението, където се вграждат по-голямата част от данните, за сметка на долната дясна област, където някои блокове ще останат без вградени данни.

3.3.4 Кодиране на данните



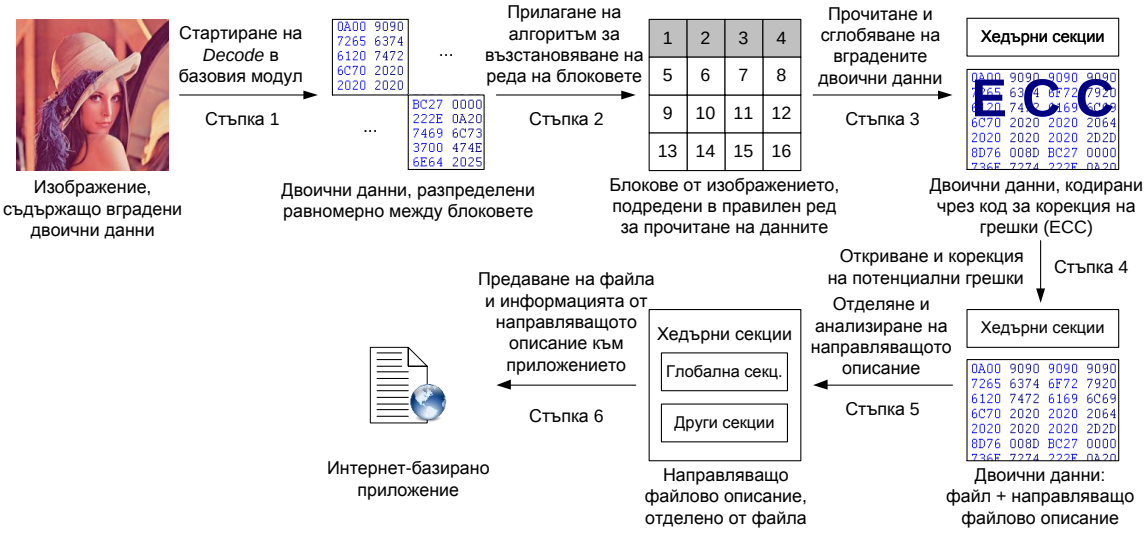
Фиг. 24: Стеганографски метод - кодиране

Процесът на кодиране включва създаването на направляващо файлово описание, както и алгоритмите за корекция на грешки и разбъркване на данните, описани в предните раздели. Неговата цел е да постигне устойчиво и визуално недоловимо криене на максимално количество данни в изображението. Състои се от стъпките, показани на Фиг. 24:

1. Създаване на направляващо файлово описание (хедър) и добавянето му към съдържанието на файла.
2. Прилагане на алгоритъм за корекция на грешки върху направляващо описание и съдържанието на файла (в този случай – алгоритъма на Хеминг, описан в раздел 3.3.2).
3. Приложно-специфичният модул стартира етапа на подготовка на кодирането чрез метода *PrepareToEncode* на базовия модул, за да получи броя на блоковете в изображението по хоризонтала и вертикала, както и максималното количество вграждани битове за всеки блок.
4. Прилагане на алгоритъм за разбъркване на данните с цел да се определи псевдослучаен ред на блоковете, съобразно който битовете от ECC-кодирани двоични данни се вграждат в изображението.
5. Двоичните данни се разпределят равномерно между блоковете от изображението, следвайки псевдослучайния ред, определен в предната стъпка.
6. Приложно-специфичният модул предава вече разпределените двоични данни на базовия модул, като стартира етапа на завършване на кодирането чрез метода *FinishEncode* на базовия модул, който има за задача да кодира данните в JPEG изображението.

7. Изображението, получено след завършването на *FinishEncode* и съдържащо вградените данни, се предава към приложението, стартирало процеса на криене на данните. Приложението поема по-нататъшното съхранение или споделяне на изображението в Интернет.

3.3.5 Декодиране на данните



Фиг. 25: Стеганографски метод - декодиране

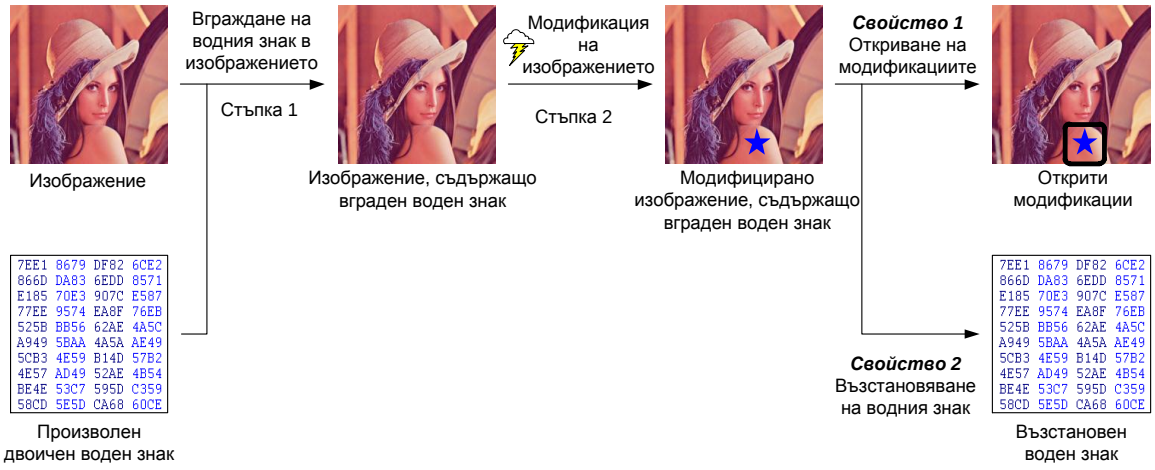
Процесът на декодиране обръща стъпките, от които се състои процесът на кодиране, описан в предишния раздел, като възстановява вградените двоични данни заедно с информацията в тяхното направляващо описание. Ако са налични грешки, те могат да бъдат открити и коригирани посредством използвания (незадължителен) алгоритъм за корекция на грешки. Процесът на декодиране се състои от следните стъпки (Фиг. 25):

1. Приложно-специфичният модул стартира процеса на декодиране чрез метода *Decode* на базовия модул, за да получи размера и броя на блоковете, от които се състои изображението в хоризонтална и вертикална посока, и двоичните данни, вградени във всеки блок.
2. Прилагане на алгоритъм за възстановяване на реда на блоковете (аналогичен на алгоритъма за разбъркване на данните), който подрежда блоковете от изображението в правилния ред за прочитане на вградените данни.
3. Прочитане и сглобяване на двоичния файл и неговото направляващо описание, разпределени равномерно между блоковете.
4. Ако се използва алгоритъм за корекция на грешки (в този случай алгоритъмът на Хеминг) и той открие грешки в двоичните данни, то те се коригират както е описано в раздел 3.3.2.

5. Направляващото описание (хедър) се отделя от двоичните данни и се анализира, за да се получи повече информация за файла.
6. Съдържанието на файла и информацията, получена от направляващото описание, се предават към приложението, стартирало процеса на декодиране на данните. Приложението поема по-нататъшното съхранение или споделяне на данните в Интернет.

3.4 Приложно-специфичен модул за целите на цифровото маркиране. Модулен метод за цифрово маркиране

Разработеният приложно-специфичен модул за целите на цифровото маркиране използва базовия модул, за да постигне устойчивост срещу JPEG трансформации по начин, подобен на стеганографския приложно-специфичен модул.



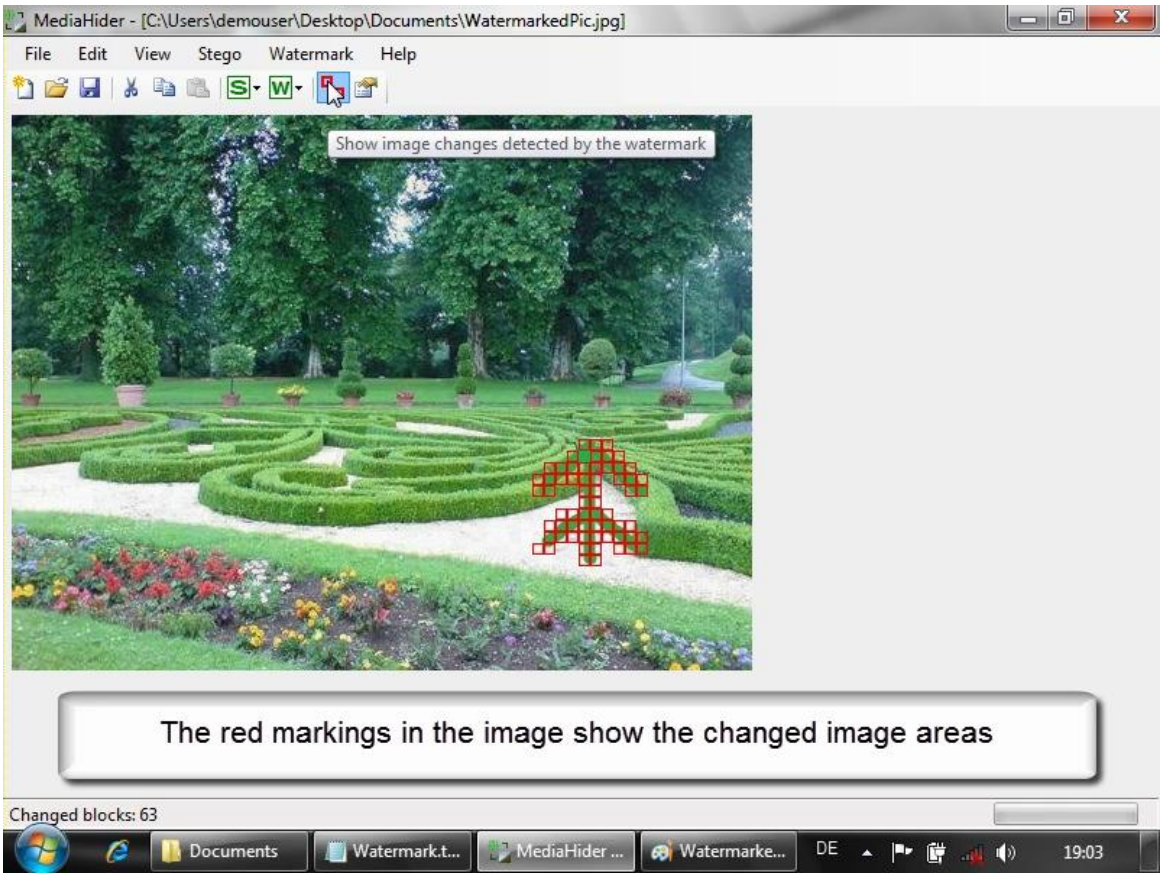
Фиг. 26: Модулен метод за цифрово маркиране – нови свойства

Тъй като методът за цифрово маркиране има различна област на приложение в сравнение със стеганографския метод, приложно-специфичният му модул има различни свойства. Вгражданите данни са значително по-малки по размер от данните, с които работи стеганографският метод. Те се наричат цифрови водни знаци и съдържат информация (т.нар. метаданни), отнасяща се до автора и авторските права, крайния потребител или самия мултимедиен файл. Освен това цифровите водни знаци се вграждат в изображението по такъв начин, че да са устойчиви срещу промени в него.

Модулният метод за цифрово маркиране наследява всички свойства, реализирани в базовия и приложно-специфичния модул. Приложно-специфичният модул реализира две нови свойства, които правят метода подходящ за използване в сценарии, нуждаещи се от проверка на автентичността на мултимедийно съдържание (Фиг. 26):

1. Откриване на области в изображението, които са били неправомерно променени след вграждането на цифровия воден знак.

2. Надеждно възстановяване на вградения воден знак в случай на направени промени в изображението.



Фиг. 27: Проверка на изображение, съдържащо неправомерно променени JPEG блокове

Главните предимства на метода за цифрово маркиране са:

1. Той не просто потвърждава наличието на предварително зададен воден знак, а може да прочете точния вграден произволен (неизвестен за декодера) воден знак, при условие, че настъпилите промени не са прекалено големи. Това свойство на метода позволява максимална гъвкавост на неговото практическо приложение, тъй като единствената информация, от която се нуждае декодерът, за прочитане на водния знак е самото изображение (както и разбира се потребителски-дефинирана парола, ако съответното приложение го изисква).
2. Методът използва потребителски-дефиниран двоичен воден знак. Той може да се състои от произволни данни, подходящи за съответното приложение, без това да се отразява на функционалността на метода.
3. Методът се ползва с всички предимства на модулната архитектура – реализация (капсулиране) на всяко свойство в определен модул, лесна поддръжка и отстраняване на грешки, възможност за подобрение чрез добавяне на нови свойства или

подобряване на съществуващите такива, без това да влияе на реализираната в други модули функционалност.

Главният недостатък, породен от новите свойства, се състои в повишената сложност на приложно-специфичния модул и намаления максимален размер на вгражданите данни, формиращи цифровия воден знак. Това е породено от вграждането на множество копия на водния знак в изображението (вж. раздели 0 и 3.4.3). Максималният размер на вгражданите данни, измерван в байтове, е поне четири пъти по-малък от максималния размер данни, вграждани от стеганографския приложно-спцифичен модул, като точните стойности зависят от характеристиките на конкретното изображение.

На Фиг. 27 е показан резултатът от проверката на изображение (авторска снимка) посредством програмната система, описана в Глава 4. Изображението съдържа известен брой неправомерно променени JPEG блокове с размер 8×8 пиксела, които са маркирани в червено (борчето в долната дясна част на изображението). По този начин лесно може да се оцени автентичността на дадено изображение и, ако има опит за фалшификация или промяна, може да се направи предположение за мотивите, стоящи зад този опит посредством разположението и информацията, съдържаща се в идентифицираните в червено области.

3.4.1 Направляващо описание и кодове за корекция на грешки

В сравнение с направляващото описание, използвано от стеганографския метод, направляващото описание на цифровия воден знак има по-проста структура. То се състои от една задължителна глобална секция и множество допълнителни секции по избор (Фиг. 28). Глобалната секция съдържа информация за прилагането на компресия, криптиране, незадължителните кодове за корекция на грешки (ECC), дължината на цифровия воден знак и секциите по избор. Секции по избор могат да съдържат допълнителна информация, свързана с водния знак и неговата употреба, зависеща от конкретния Интернет-базиран сценарий.

Фиг. 28: Метод за цифрово маркиране – направляващо описание на цифровия воден знак

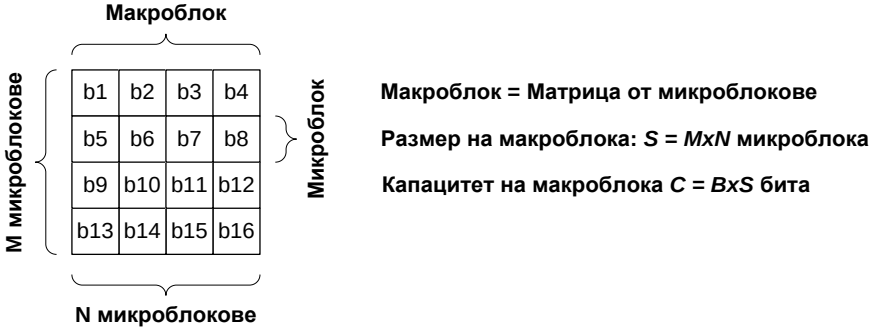
След като направляващото описание се създаде и добави към съдържанието на цифровия воден знак, получените двоични данни могат да се кодират с помощта на алгоритми за откриване и корекция на грешки по начина, описан в раздел 3.3.2,. Използването на кодове за корекция на грешки не е задължително и зависи от изискванията на приложението. Основна причина за избягването на употребата на кодове за корекция

на грешки е нуждата от по-голям размер на водния знак, който често е сериозно ограничен от алгоритмите за цифрово маркиране.

След евентуалното прилагане на незадължителните кодове за корекция на грешки, допълненият с направляващото описание воден знак е готов за вграждане в изображението.

3.4.2 Макроблокове

Алгоритъмът за цифрово маркиране разделя изображението на няколко големи области, наречени макроблокове. Всеки макроблок съдържа пълно копие на водния знак, което се вгражда в съдържанието на макроблока и служи като негова сигнатура. Ако даден макроблок се промени, вградената в него сигнатура също се променя. Декодерът на алгоритъма може по-късно да открие тези промени и да идентифицира тяхното местоположение в изображението посредством сравнение на сигнатурите на различните макроблокове.



Фиг. 29: Макроблокове – структура, размер и капацитет

Всеки макроблок се състои от матрица от микроблокове, която има M реда и N колони и така съставя макроблок с големина $S = M \times N$ микроблока (Фиг. 29). Микроблоковете съответстват на блоковете на изображението, получени като резултат от етапа на подготовка на кодирането (*PrepareToEncode*) в базовия модул. По дефиниция всеки микроблок съдържа B бита вградени данни, които изпълняват ролята на негова сигнатура и се използват за откриването на промени в микроблока. Следователно, общият капацитет C на даден макроблок е равен на сумата на капацитетите на микроблоковете, които съставят макроблока: $C = B \times S$ бита. Броят и размерите на макроблоковете зависят от размера на изображението и общата дължина L на водния знак.

Размерът на изображението е важен, тъй като той директно определя броя на микроблоковете (вж. раздел 3.2.3). По-голямото изображение съдържа повече микроблокове и позволява формирането на повече и/или по-големи макроблокове и съответно вграждането на по-дълги водни знаци.

Дължината на водния знак определя непосредствено долната граница на размера на макроблоковете. Те трябва да бъдат достатъчно големи, за да позволят вграждането на

целия воден знак, т.е. трябва да бъде изпълнено неравенството $\mathrm{L} \leq C$, което води до следното ограничение на размера на макроблока:

$$S = M \times N \geq \frac{\mathrm{L}}{B}.$$

Освен това водният знак се вгражда многократно в изображението – веднъж във всеки макроблок. Многократното вграждане на водния знак има връзка със степента на устойчивост срещу промени на изображението и се дискутира подробно в следващия раздел. Увеличаването на броя на макроблоковете води до повишаване на вероятността за успешно възстановяване на вградения воден знак след внасяне на промени в изображението. За да осигури достатъчно добра вероятност за възстановяване на водния знак, приложно-специфичният модул използва винаги минимум 4 макроблока.

Нека дефинираме съотношенията *RC* и $RC^{(i)}$, съответно за цялото изображение и за всеки един макроблок i по следния начин:

$$RC = n_y \,/\, n_x \,,\; RC^{(i)} = M^{(i)} \,/\, N^{(i)} \,,$$

където n_x и n_y обозначават съответно броя микроблокове по хоризонтала и вертикала в изображението, а $M^{(i)}$ и $N^{(i)}$ обозначават съответно броя на редовете и колоните от микроблокове в даден макроблок i. Съотношението $RC^{(i)}$ се избира за всеки макроблок i по такъв начин, че $RC^{(i)} \approx RC$ за $\forall i$. По този начин се цели увеличаване на устойчивостта на макроблоковете и вградените в тях водни знаци срещу промени в изображението. Освен това при конструирането на всеки макроблок i се предпочита използването на малки размери. Търсят се:

$$M^{(i)}, N^{(i)} | S^{(i)} = M^{(i)} \times N^{(i)} \approx \min_{a,b | M_a^{(i)} \times N_b^{(i)} \geq L/B \text{ и } M_a^{(i)}/N_b^{(i)} \approx RC} \left(M_a^{(i)} \times N_b^{(i)} \right),$$

където с $S^{(i)}$ се обозначава броят на микроблоковете в макроблок i, а с индексите a и b се обозначават съответно различните възможни стойности на броя редове и колони, съставящи макроблока. Това води до по-голям брой макроблокове и съответно по-голям брой вградени копия на водния знак, което увеличава устойчивостта на вграждането.

Следващият раздел разглежда използването на макроблоковете за откриване на промени в изображението и за възстановяване на вградения воден знак от промененото изображение.

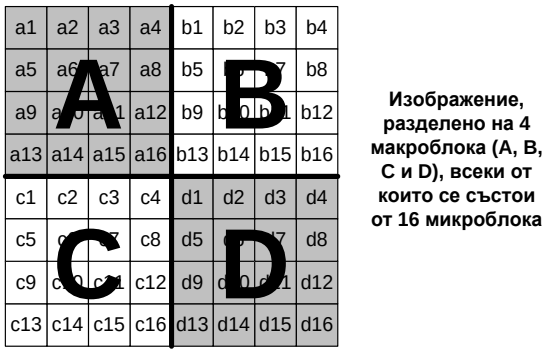
3.4.3 Откриване на промени в изображението и възстановяване на водния знак

Откриването на промени в изображението и възстановяването на вградения воден знак се извършват посредством анализ на съдържанието на макроблоковете, от които е съставено изображението (Фиг. 30). По време на кодирането идентични копия на цифровия воден знак се вграждат във всеки макроблок X ($X \in \{A, B, C, D, \ldots\}$). Битовете, от които е съставен водния знак, се разпределят равномерно между микроблоковете x_i ($x \in \{a, b, c, d, \ldots\}, i \in N$), така че всеки микроблок x_i съдържа B бита от водния знак

(Фиг. 31). По време на разпределянето на битовете може да се приложи алгоритъмът за разбъркване на данни, описан в раздел 3.3.3 с цел промяна на реда на микроблоковете, в които се вгражда всяка следваща група от *B* бита от водния знак.

По време на декодирането вградените във всеки макроблок копия на водния знак се прочитат и сравняват помежду си. Ако промените в изображението не са били прекалено големи по размер, повечето копия ще бъдат идентични с оригиналния воден знак. Останалите копия ще съдържат разлики, чието положение ще отговаря на променените микроблокове в макроблока, съдържащ промененото копие. По този начин е възможно да се локализират променените области от изображението и едновременно с това да се възстанови оригиналният вграден воден знак, при положение че са останали достатъчен брой непроменени микроблокове и съответно макроблокове.

На Фиг. 30 и Фиг. 31 е разгледано изображение, разделено на 4 макроблока *A, B, C* и *D*, всеки от които се състои от 16 микроблока x_1 до $x_{16}, x \in \{a, b, c, d\}$. Следователно, съществуват общо 4 копия на водния знак – едно копие във всеки макроблок. Всяко копие е разпределено равномерно между шестнадесетте микроблока на съответния макроблок, както е показано на Фиг. 31. Всеки микроблок x_i съдържа $B = 2$ бита от водния знак. Големината на целия воден знак и съответно на всяко негово копие е $L = 16 \times 2 = 32$ бита.

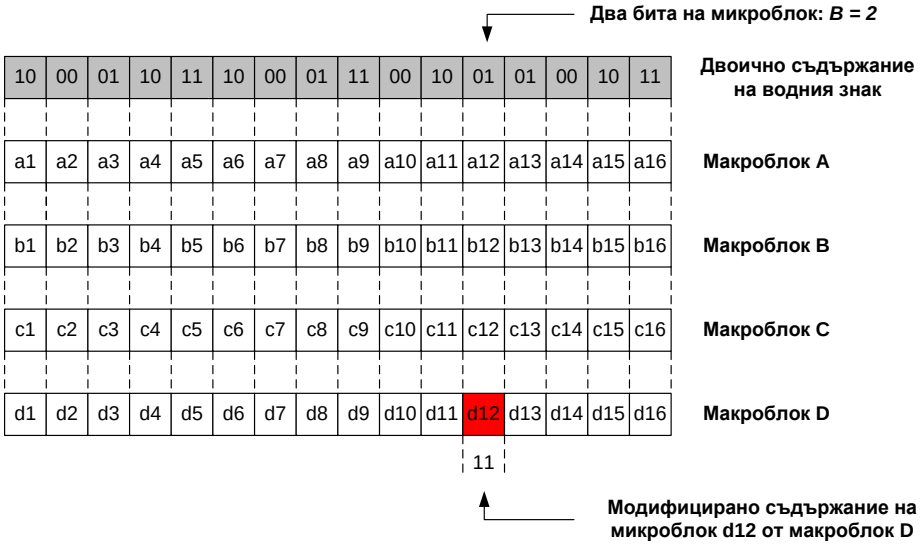


Фиг. 30: Макроблокове в изображение

За да бъдат открити евентуални промени в изображението, четирите копия на водния знак се прочитат от изображението и се сравняват едно с друго. Копията, прочетени от макроблоковете *A, B* и *C* са еднакви: за $\forall i \in \{1,2,\dots,16\}$: $q_{a_i} = q_{b_i} = q_{c_i}$, където с q_{x_i} се обозначава частта от водния знак, скрита в микроблок x_i. Копието, прочетено от макроблок *D,* съдържа единствена разлика, намираща се в микроблок d_{12}: $q_{d_{12}} = (11)_2 \neq (01)_2$. Това означава, че микроблок d_{12} от изображението е бил променен след вграждането на водния знак. В този случай оригиналният воден знак q може да се възстанови, като двоичното му съдържание съответства на трите идентични копия, прочетени от макроблоковете *A, B* и *C*: $\mathrm{q} = (\mathrm{q}_{\mathrm{a}_1}\mathrm{q}_{\mathrm{a}_2} \dots \mathrm{q}_{\mathrm{a}_{16}})_2$.

Тъй като базовият модул осигурява устойчивост срещу JPEG трансформации, вградените водни знаци остават непроменени след промени на изображението, предизви-

кани от JPEG компресия, декомпресия или рекомпресия. Следователно JPEG трансформациите няма да се отчетат като промени от алгоритъма, описан по-горе.



Фиг. 31: Откриване на промени в изображението

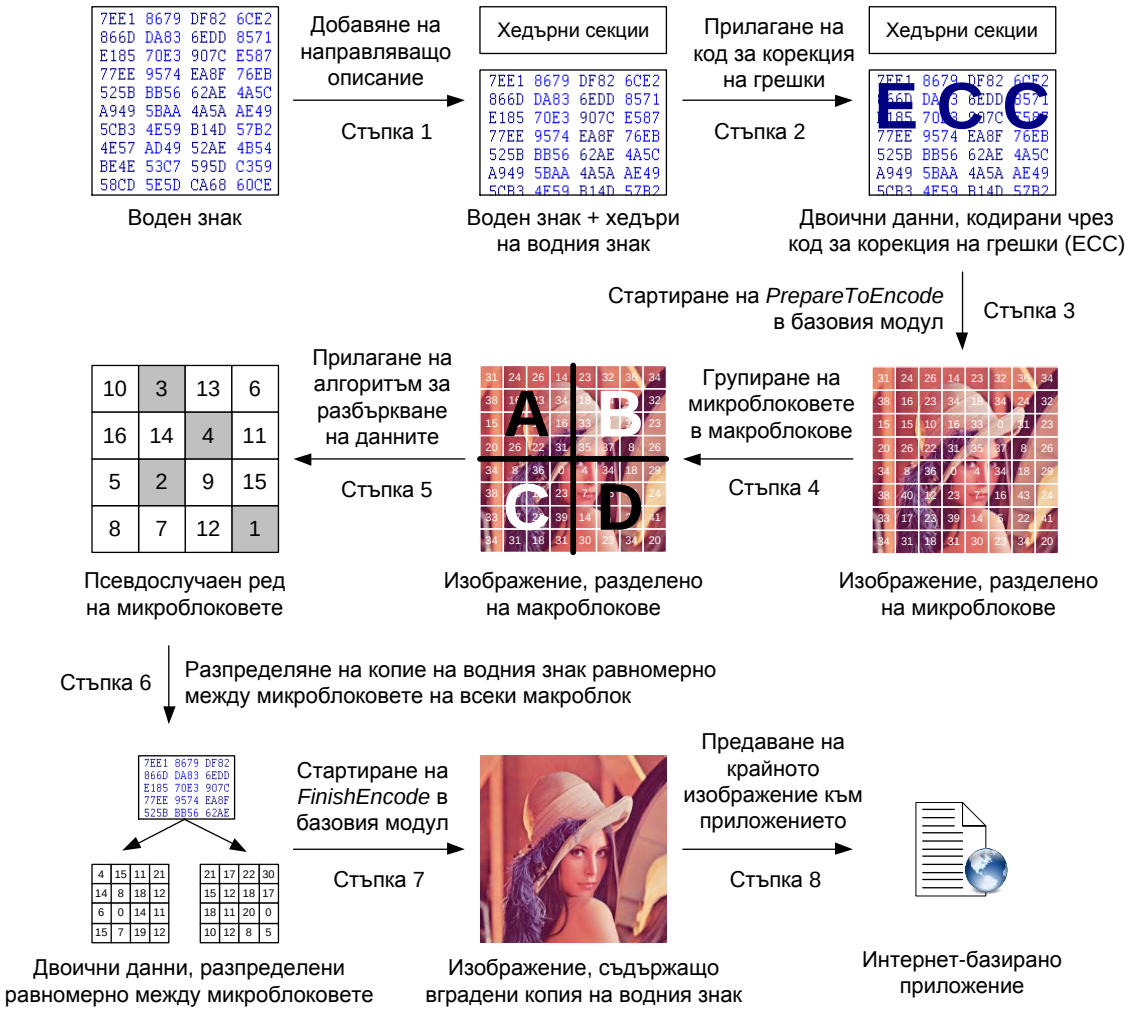
Методите за цифрово маркиране, които позволяват определени видове промени на изображенията (като например компресия) се наричат полукрехки (англ. semi-fragile) [6], [14]. Те са подходящи за приложение в Интернет-базирани сценарии, където компресията на мултимедия е необходимост и не представлява неправомерна промяна на мултимедията. Свойството „полукрехкост" на модулния метод за цифрово маркиране, разработен в тази дисертация, възниква в резултат от комбинацията на използваните базов и приложно-специфичен модул.

3.4.4 Кодиране на водния знак

Процесът на кодиране, който реализира вграждането на водния знак в макроблоковете на изображението, се състои от следните стъпки (Фиг. 32):

1. Създава се и се добавя направляващото описание (хедър) на водния знак.
2. При необходимост, алгоритъмът за корекция на грешки (алгоритъм на Хеминг в този случай) се прилага върху водния знак и добавеното направляващо описание.
3. Приложно-специфичният модул стартира етапа на подготовка (*PrepareToEncode*) в базовия модул, за да получи общия брой блокове (микроблокове) в изображението във всяка посока, както и техния капацитет - максималното количество на вграждане битове за всеки микроблок.
4. Използвайки информацията от предходната стъпка, микроблоковете се групират в макроблокове с подходящо разположение и големина. Големината на всеки макроблок трябва да е достатъчна, за да побере едно копие на водния знак.

5. Прилага се алгоритъм за разбъркване на данните, за да промени реда на микроблоковете в рамките на всеки макроблок. По този начин се подобрява сигурността на метода и се повишава визуалното качество на изображението, както е описано в раздел 3.3.3.

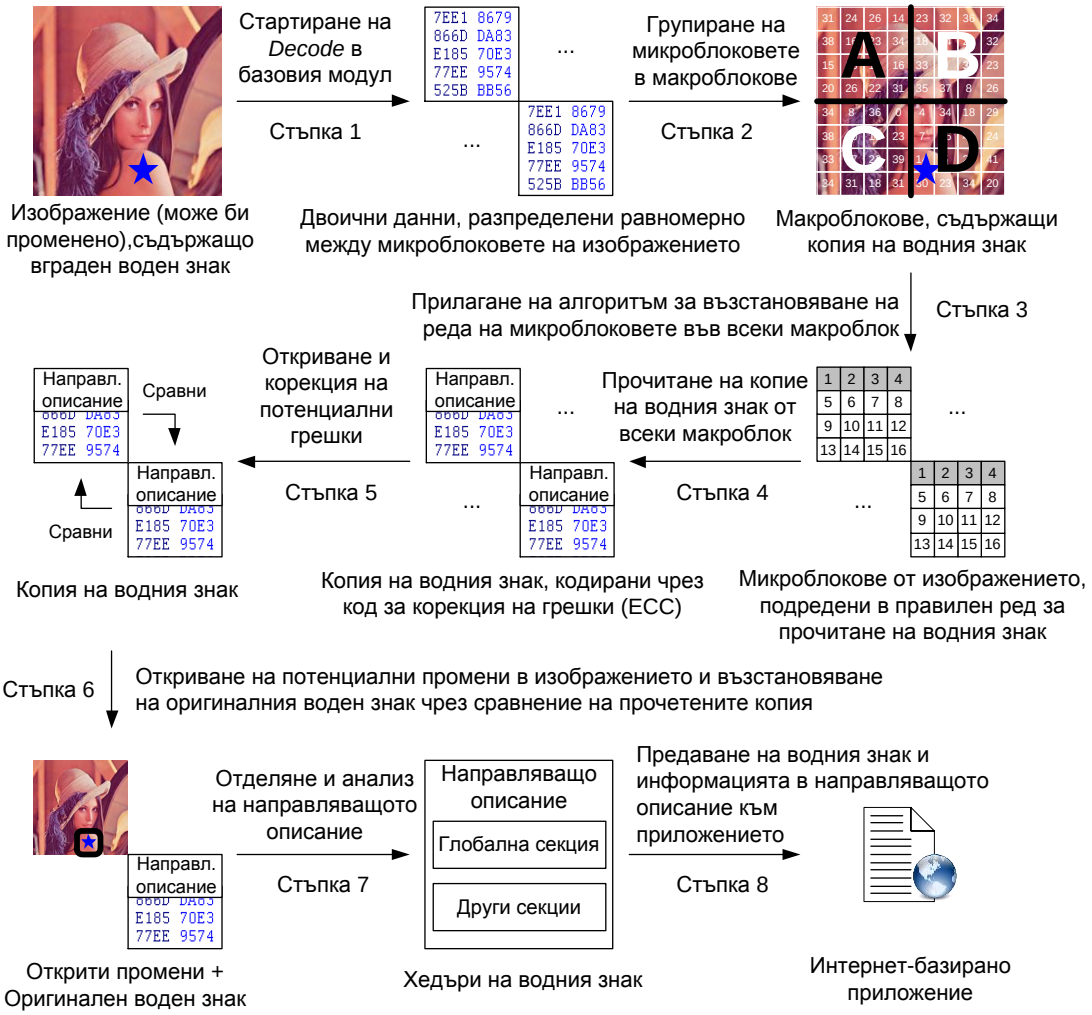


Фиг. 32: Метод за цифрово маркиране - кодиране

6. Копие на водния знак се разпределя равномерно между микроблоковете на всеки макроблок, следвайки псевдослучайния ред на микроблоковете, определен в предходната стъпка.
7. Разпределените двоични данни за всеки микроблок се предават на етапа на завършване на кодирането (*FinishEncode*) в базовия модул, за да бъдат кодирани в JPEG изображението.
8. Изображението, получено като резултат от етапа на завършване на кодирането и което съдържа няколко вградени копия на водния знак, се предава към приложени-

ето, стартирало процеса на криене на водния знак. Приложението поема по-нататъшното съхранение или споделяне на изображението в Интернет.

3.4.5 Декодиране на водния знак



Фиг. 33: Метод за цифрово маркиране - декодиране

Процесът на декодиране открива промени в изображението и прочита вградения воден знак. Този процес се състои от следните стъпки (Фиг. 33):

1. Входното изображение се предава на процеса на декодиране (*Decode*) в базовия модул, за да се получи размера и броя на микроблоковете, от които се състои изображението във хоризонтална и вертикална посока, както и вградените във всеки блок двоични данни.
2. Използвайки информацията, получена в предходната стъпка, микроблоковете се групират в съответните им макроблокове, всеки от който съдържа вградено копие на оригиналния воден знак.

3. Прилага се алгоритъм за възстановяване на реда на микроблоковете (аналогичен на алгоритъма за разбъркване на данните). Той подрежда микроблоковете от изображението в правилния ред за прочитане на водния знак.
4. Прочитат се копията на водния знак, вградени във всеки макроблок.
5. Ако е необходимо, алгоритъмът за корекция на грешки, описан в раздел 3.3.2 (в този случай алгоритъм на Хеминг), се прилага върху всяко копие от водния знак, за да се възстановят евентуални случайни грешки.
6. Получените копия на водния знак се сравняват помежду си, за да се открият променени микроблокове (ако има такива) и за да се определи водния знак, който е бил вграден в изображението.
7. Направляващото описание (хедър) на водния знак се отделя от двоичните данни и се анализира, за да се получи допълнителна информация, свързана с водния знак.
8. Координатите на открити променени микроблокове в изображението, двоичният воден знак и информацията, получена от анализа на направляващото описание, се предават към приложението, което е стартирало процеса на декодиране на водния знак. Приложението поема по-нататъшното съхранение и обработка на резултатите.

3.5 Базов модул за изображения с беззагубна компресия

Базовият модул, представен в този раздел, е разработен с цел верификация на модулния подход и употреба на приложно-специфичните модули за целите на стеганографията и цифровото маркиране, представени съответно в раздели 0 и 0, върху некомпресирани изображения (напр. BMP) или изображения, компресирани без загуба на информация (напр. PNG). В сравнение с базовия модул, устойчив срещу JPEG трансформации, представен в раздел 3.2, този базов модул не е устойчив срещу загубна компресия, но предоставя възможност за вграждане на значително по-голямо количество данни. Освен това при неговото използване в комбинация с приложно-специфичния модул за цифрово маркиране, точността на определяне на неправомерни промени в изображението е до пиксел.

Базовият модул за изображения с беззагубна компресия се нуждае от значително по-малко системни ресурси (изчислително време и системна памет) в сравнение с базовия модул, устойчив срещу JPEG трансформации. Това е следствие от сравнително опростената функционалност на модула, която е концентрирана изцяло върху максимизирането на количеството вграждани данни при минимално изменение в изображението без оглед на устойчивостта срещу компресия или други трансформации на изображението.

3.5.1 Основни цели при разработването на модула

Като пряко следствие от дефиницията на модула, разсъжденията, изложени в раздел 2.1.2, не са взети предвид при неговата разработка и не са част от спецификацията му. Разсъжденията, изложени в раздел 2.1.3, продължават да бъдат от първостепенно значение, аналогично на базовия модул, устойчив срещу JPEG трансформации (виж раздел 3.2). Основните цели при разработването на този базов модул могат да бъдат обобщени както следва:

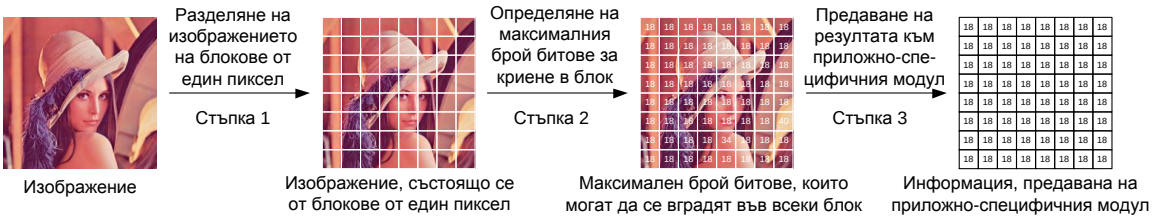
1. Да се осигури максимално количество на вгражданите в дадено изображение данни без оглед на степента на устойчивост срещу загубна компресия.
2. Да се осигури възможност за работа с произволни изображения и данни за вграждане посредством третирането на данните като произволна поредица от битове.
3. Да се осигури възможност за прочитане на вградените данни от носещото изображение, без да е необходима информация, отнасяща се до първоначалния оригинал преди криенето на данните.
4. Да се гарантира безпогрешното прочитане и възстановяване на скритата информация без необходимост от кодове за корекция на грешки.

Едновременното реализиране на тези четири цели при разработката на базовия модул дава достатъчно свобода на приложно-специфичните модули за реализирането на разнообразни приложни сценарии, които не включват загубна компресия.

3.5.2 Базов модул: кодиране

Кодирането на информацията в базовия модул според спецификацията на модулното проектиране се състои от познатите два етапа на подготовка (*PrepareToEncode*) и на завършване (*FinishEncode*) на кодирането.

Етапът на подготовка преминава през следните стъпки (Фиг. 34):



Фиг. 34: Базов модул – етап на подготовка (*PrepareToEncode*)

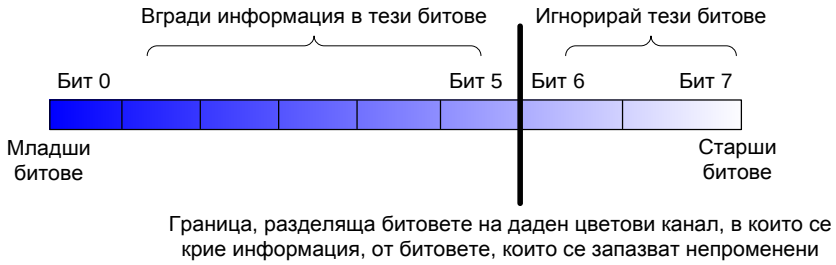
1. Изображението се разделя на блокове от пиксели с големина 1×1 пиксел, т.е. всеки блок от изображението се състои от един единствен пиксел. Броят на блоковете от пиксели по хоризонтала и вертикала е равен съответно на ширината и височината на изображението, измерени в пиксели:

$$n_x = d_x\,,\, n_y = d_y\,,$$

където n_x и n_y обозначават съответно броя блокове по хоризонтала и вертикала, а d_x и d_y обозначават съответно ширината и височината на изображението.

Размерите на блока по хоризонтала (b_x) и вертикала (b_y) за този базов модул са винаги равни на 1 пиксел.

2. Определя се максималното количество информация (в битове), което може да бъде скрито във всеки блок от пиксели. Информацията се вгражда в най-младшите битове на стойностите на цветовите канали на всеки пиксел. Нейното максимално количество зависи от типа на изображенията - цветни или с нива на сивото. Изображенията с нива на сивото се състоят само от *1* цветови канал, който представя интензитета на пикселите. Цветните изображения разполагат с *3* цветови канала, които могат да се интерпретират по различни начини. Най-често всеки един от трите канала представя един от трите основни адитивни цвята: червен, зелен или син, които се използват в устройства, генериращи светлина: монитори, проектори и др. (RGB цветово пространство, [102]). Всеки цветови канал се състои традиционно от 8 бита. Представеният базов модул вгражда информация в максимално 6 от най-младшите битове на даден канал (Фиг. 35). Това води до максимално количество на вградената информация в даден пиксел/блок, равно на *6* бита за изображения с нива на сивото и $3 \times 6 = 18$ бита за цветни изображения.

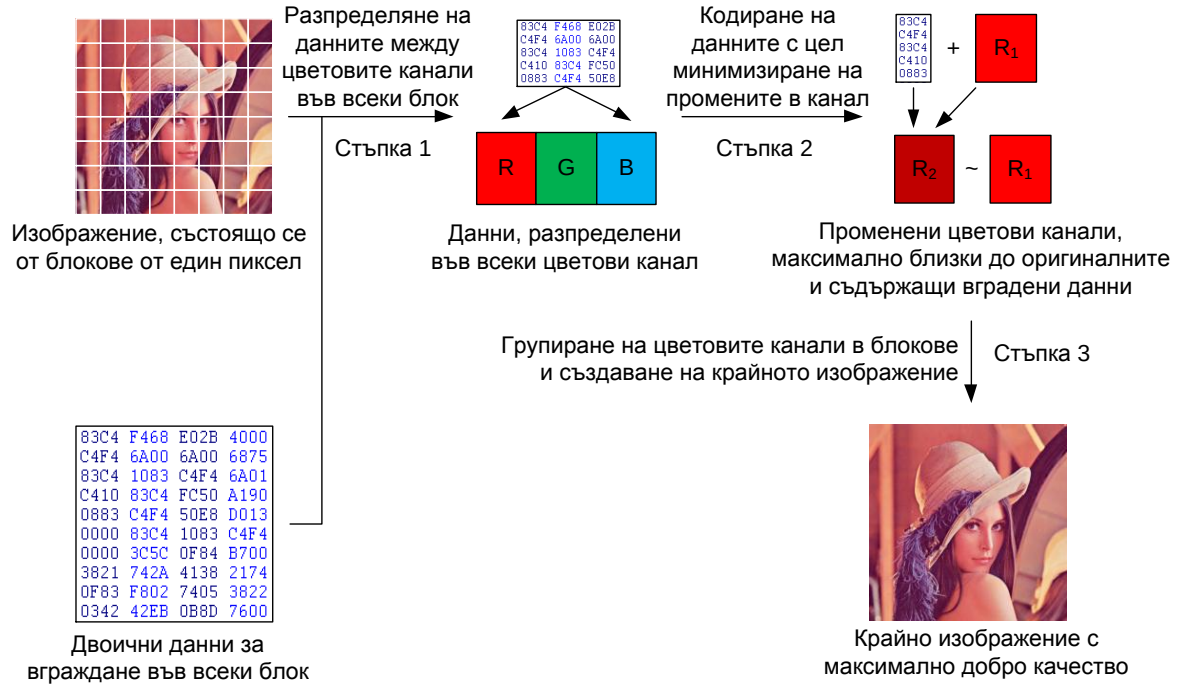


Фиг. 35: Разпределение на битовете в даден цветови канал

На практика количеството информация, вграждано в цялото изображение се определя от цялостния размер на вграждания файл, който може да се компресира, криптира и кодира посредством кодове за корекция на грешки. Същинското разпределение на информацията между блоковете се определя от приложно-специфичните модули, които следват техни собствени критерии, подходящи за съответното приложение.

3. Броят на блоковете от пиксели във всяка посока се предава към приложно-специфичния модул заедно с максималното количество на информация за вграждане във всеки блок.

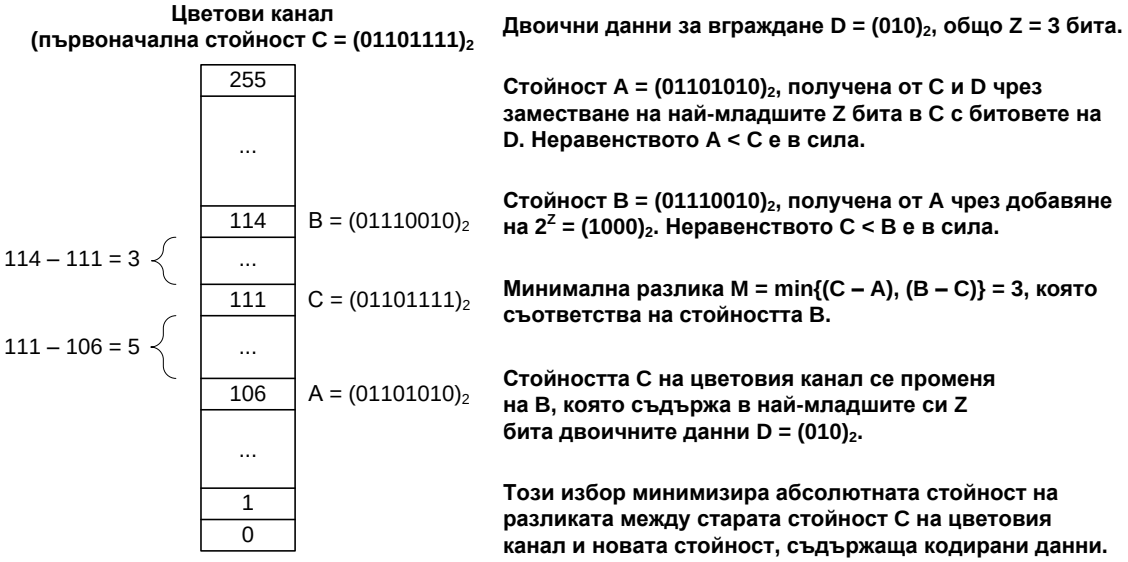
Подобно на базовия модул, описан в раздел 3.2, етапът на завършване на кодирането приема като входни данни от страна на приложно-специфичния модул конкретните битове, които трябва да бъдат вградени във всеки блок от изображението (Фиг. 36). Броят на битовете трябва да е по-малък или равен на максималния брой битове, които могат да бъдат вградени в блока, както е определено от етапа на подготовка на кодирането. Ако това ограничение е изпълнено, т.е. данните за вграждане не са твърде много за даденото изображение, алгоритъмът изпълнява следните стъпки:



Фиг. 36: Базов модул – етап на завършване на кодирането (*FinishEncode*)

1. Данните се разпределят равномерно между отделните цветови канали на всеки блок от изображението – най-често червен (R), зелен (G) и син (B) канал, но методът може да се използва и с други цветови пространства, напр. YC_bC_r. С разпределянето се цели повишаването на субективното качество на изображението чрез избягване на прекомерната промяна на единичен цветови канал. Разпределението става с помощта на псевдослучайни пермутации, базирани на генератора на псевдослучайни числа на Марсалия (вж. раздел 3.3.3 и [109]). Това цели повишаване на сигурността на подхода и гарантиране на равномерна промяна на качеството на изображението. Използва се набор от няколко псевдослучайни пермутации, които разделят данните на три приблизително равни псевдослучайни части. Всяка от тях се вгражда в един цветови канал.
2. Кодиране на данните в най-младшите битове на съответния цветови канал. При това кодиране се използва вариант на т.нар. Модулация на Индекса на Квантизация (англ. Quantized Index Modulation, QIM, вж. раздел 2.2.1), който цели минимизира-

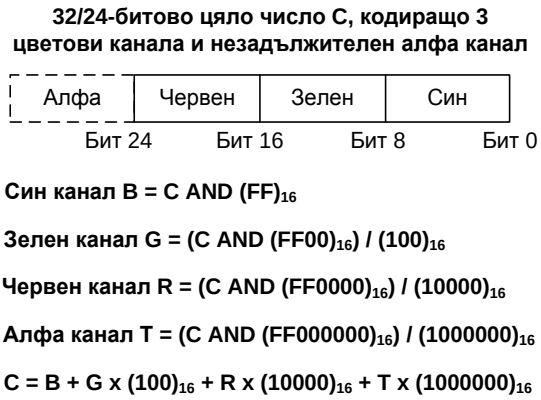
нето на промените в канала. Използваният алгоритъм анализира стойността *C* на цветовия канал в блока (цяло число в интервала [*0*; *255*] при *8*-битов канал). Генерират се две нови числови стойности *A* и *B*, които са максимално близко до *C* и кодират в себе си желаните двоични данни *D*, като *A* е по-малко или равно на *C*, а *B* е по-голямо или равно на *C*. Сравняват се разликите *(C - A)* и *(B - C)* и за нова стойност на цветовия канал се избира това число *A* или *B*, което съответства на минималната разлика. На Фиг. 37 е показан пример за кодирането на двоичните данни $D = (010)_2$ в *8*-битов цветови канал, имащ първоначална стойност $C = (01101111)_2$. По този начин ефектът от промените в цветовия канал се свежда до минимум и се подобрява общото качество на изображението – минимизира се разликата между оригиналното изображение и изображението, съдържащо вградените данни.



Фиг. 37: Кодиране на данните в най-младшите битове на даден цветови канал

3. Групиране на цветовите канали в блокове и създаване на крайното изображение. При групирането, както и при разделянето на един блок на три цветови канала трябва да се има предвид, че блокът най-често се представя като 24-битово или 32-битово цяло число, което съдържа кодирани стойностите на трите цветови канала, както и незадължителен алфа-канал (при 32-битова стойност), определящ степен на прозрачност (общо 256 нива, кодирани посредством най-старшите 8 бита). Фиг. 38 показва най-разпространеното кодиране на 8-битови RGB-цветови канали.

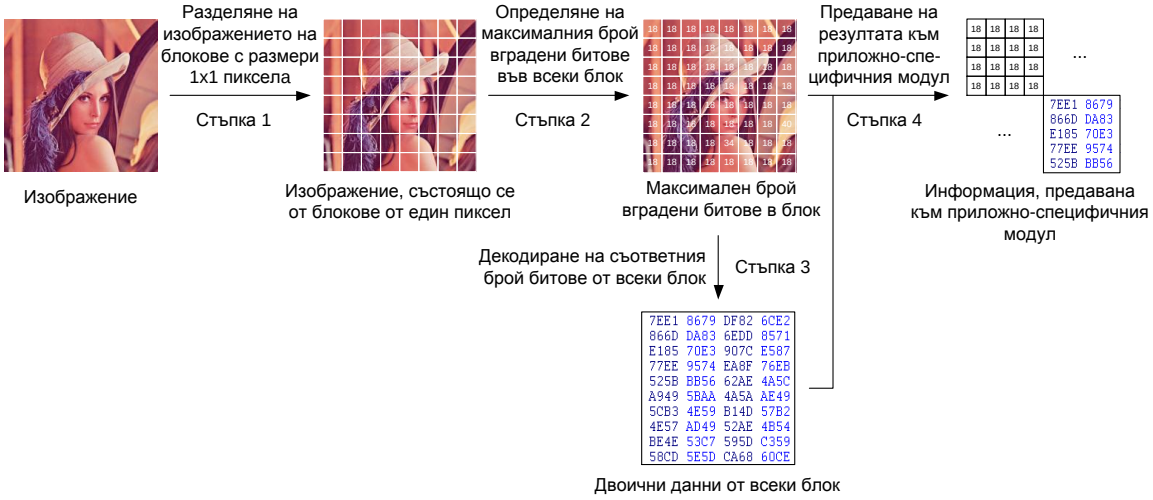
След като стъпките, описани по-горе, се изпълнят за всеки блок от изображението, крайното изображение с вградените данни се предава обратно на приложно-специфичния модул.



Фиг. 38: Кодиране на RGB-цветови канали в 24-битово или 32-битово цяло число

3.5.3 Базов модул: декодиране

За разлика от процеса на кодиране, процесът на декодиране се състои само от един етап (Фиг. 39), който е подобен на етапа на подготовка на кодирането:



Фиг. 39: Базов модул – декодиране (*Decode*)

1. Разделяне на входното изображение на блокове с размери 1×1 пиксела. Броят на блоковете в хоризонтална и вертикална посока, както и размерът на блока се предават на приложно-специфичния модул.
2. Определяне на максималния брой вградени битове във всеки блок посредством граничния алгоритъм, описан в предния раздел (Фиг. 35).
3. Декодиране на съответния брой битове от всеки блок чрез прочитане на най-младшите битове на цветовите канали в блока, като се вземат предвид и реконструират използваните набори от псевдослучайни пермутации.

4. След обработката на всички блокове от изображението, декодираните битове се предават към приложно-специфичния модул.

Аналогично на базовия модул, устойчив срещу JPEG трансформации, процесите на кодиране и декодиране се състоят от различен брой стъпки, които се нуждаят от различно време за изпълнение. Отнемащите най-много време стъпки в процеса на кодиране са част от етапа на завършване на кодирането и нямат еквивалент в процеса на декодиране. Следователно, процесът на декодиране се нуждае от значително по-малко време за изпълнение в сравнение с пълния процес на кодиране, съставен от етапите на подготовка и завършване на кодирането.

3.6 Заключение

При модулния подход за проектиране на методи за криене на данни всяко свойство на даден метод се реализира или в базовия, или в приложно-специфичния модул, от които методът е съставен. Това улеснява поддръжката на метода и отстраняването на грешки. Също така методите могат сравнително лесно да се подобряват чрез добавяне на нови свойства или усъвършенстване на вече съществуващите такива. Освен това чрез разделянето на методите на модули се постига гъвкавост при прилагането им в различни приложни Интернет-базирани сценарии.

Базовият модул, устойчив срещу JPEG трансформации, реализира всички стъпки на вграждането, които имат отношение към загубната компресия на JPEG-стандарта. Тази негова характеристика позволява на приложно-специфичните модули (и техните разработчици) да се концентрират върху реализирането на други важни за потребителите свойства на методите за криене на данни в мултимедия.

Стеганографският метод комбинира сравнително прост приложно-специфичен модул с базовия модул, устойчив на JPEG трансформации. Основна цел на този приложно-специфичен модул е максимизирането на размера на вгражданите двоични данни.

Методът за цифрово маркиране използва по-сложен приложено-специфичен модул, който позволява откриването на променени региони в изображението и надеждното възстановяване на водния знак в случай на такива промени. Това прави метода подходящ за употреба в сценарии, нуждаещи се едновременно от проверка на автентичността на мултимедийно съдържание и доказване на авторство, или в сценарии за маркиране на мултимедийни копия.

Базовият модул за изображения с беззагубна компресия реализира изцяло всички операции от ниско ниво, необходими за вграждане на двоични данни в некомпресирани изображения или изображения, подложени на беззагубна компресия. Подобно на базовия модул, устойчив срещу JPEG трансформации, той позволява на приложно-специфичните модули и техните разработчици да се концентрират върху реализирането на допълнителни приложни свойства на методите за криене на данни. Чрез комби-

ниране по различен начин на представените два приложно-специфични модула и два базови модула могат да бъдат реализирани общо четири различни метода за криене на данни, които се изследват в практическо отношение в следващите глави.

Като кратко обобщение може да се отбележи, че и четирите модулни метода работят надеждно и дават високо качество на получените изображения. Те покриват широк диапазон от Интернет-базирани сценарии, осигурявайки адаптивност и възможност за избор на най-подходящата комбинация от свойства, необходими за съответното приложение. Наборът от модулни методи може да се разшири сравнително лесно и бързо чрез добавяне на нови базови или приложно-специфични модули, всеки от които увеличава значително броя на модулните методи, които са на разположение на потребителите и автоматизираните приложения в Интернет.

В резултат на научноизследователската работа върху задачите, изложена в тази глава, се постигат следните научно-приложни приноси:

1. Разработен е нов модулен подход за криене на данни и е дефиниран наборът от методи за комуникация между модулите.
2. Създаден е собствен метод за постигане на устойчивост срещу JPEG компресия, декомпресия и рекомпресия, който работи с произволни изображения и данни за вграждане, като гарантира възстановяването на вградените данни с точност до бит. Методът е реализиран в контекстно-независим базов модул.
3. Проектиран е контекстно-независим базов модул за работа с изображения с беззагубна компресия, използващ модулация на индекса на квантизация. Той работи с произволни изображения и данни за вграждане и гарантира възстановяването на скритата информация с точност до бит.
4. Създаден е стеганографски метод за криене на данни, включващ стеганографски приложно-специфичен модул, направляващо описание на вгражданите файлове, кодове за корекция на грешки и разбъркване на данните.
5. Създаден е метод за цифрово маркиране, включващ приложно-специфичен модул за цифрово маркиране, направляващо описание на вгражданите водни знаци и възможности за откриване на променени блокове в изображението и възстановяване на вградения воден знак в случай на промени.

Модулният подход за криене на данни и разработените методи са представени в публикуваната в „Доклади на БАН“ авторска публикация [110], издадената от IEEE авторска публикация [111], отличения с награда авторски доклад [112] и публикации, цитирани в дисертационния труд под номера [113], [114] и [115].

Глава 4

Програмна реализация на модулните методи

Като предпоставка за извършването на оценка и анализ на функционалността и качествата на предложените методи за криене на данни в мултимедия, те са реализирани като програмна система посредством програмната платформа .NET на Майкрософт. Тази платформа предлага няколко специфични предимства, които улесняват и ускоряват процесите на проектиране, разработка и тестване на създадените компютърни програми [116], [117], [118]:

1. Няколко популярни езика за програмиране като C/C++, C# и Visual Basic могат да бъдат използвани заедно, като всяка част от методите е написана на най-подходящия за целта език.
2. Платформата предлага отлична функционалност за проследяване и отстраняване на грешки (англ. debugger), която е незаменима при разработването на алгоритмите.
3. Наличие на обширни библиотеки, които предлагат често използвана стандартизирана функционалност (напр. създаване на графичен потребителски интерфейс, комуникация с файловата система, множество криптографски алгоритми и др.) и структури от данни, улесняващи разработката на компютърни програми.
4. Осигуряване на Интернет-базирана комуникация чрез мрежови услуги (англ. web services), чрез протоколите за комуникация на приложно ниво HTTP и FTP или чрез протоколите за комуникация на по-ниско ниво TCP и UDP. Платформата поема грижата за много от детайлите, необходими за комуникация чрез съответните протоколи.
5. Лесна интеграция с други технологии на Майкрософт (напр. с Интернет сървъра IIS и Sharepoint), както и възможност за пренасяне (англ. porting) към други операционни системи или хардуерни платформи на по-късен етап от разработката [119] .

Тези предимства правят .NET платформата подходяща за създаването на програмен прототип, който реализира модулните методи за криене на данни и служи като основа при тяхната верификация в рамките на различни Интернет-базирани сценарии.

Освен дискутираните до момента възможности на модулните методи, програмната реализация предлага възможност за предварителна обработка на вгражданите данни. Те могат да бъдат компресирани с помощта на програмната библиотека с отворен код *zlib*, която използва леко изменен вариант на алгоритъма *LZ77* (Lempel–Ziv 1977) [120]. След това се предлага възможност за класическо криптиране според стандартите *3DES*

и *AES*, като се използва парола, въведена от потребителя. Тези предварителни обработки на данните могат да бъдат извършени и с помощта на други помощни програми, избрани от потребителя, но за подобряване на ефективността са интегрирани в разработения програмен прототип.

Прототипната реализация на модулните методи за криене на данни минава през два етапа:

1. Създаване на архитектура на програмната система, която показва разделянето на системата на компоненти, всеки от които реализира една или повече функции и
2. Реализиране на отделните компоненти чрез класове (при използване на обектно-ориентирани програмни езици).

В следващите раздели се представя архитектурата на програмната система и се описват най-важните компоненти и тяхната програмна реализация.

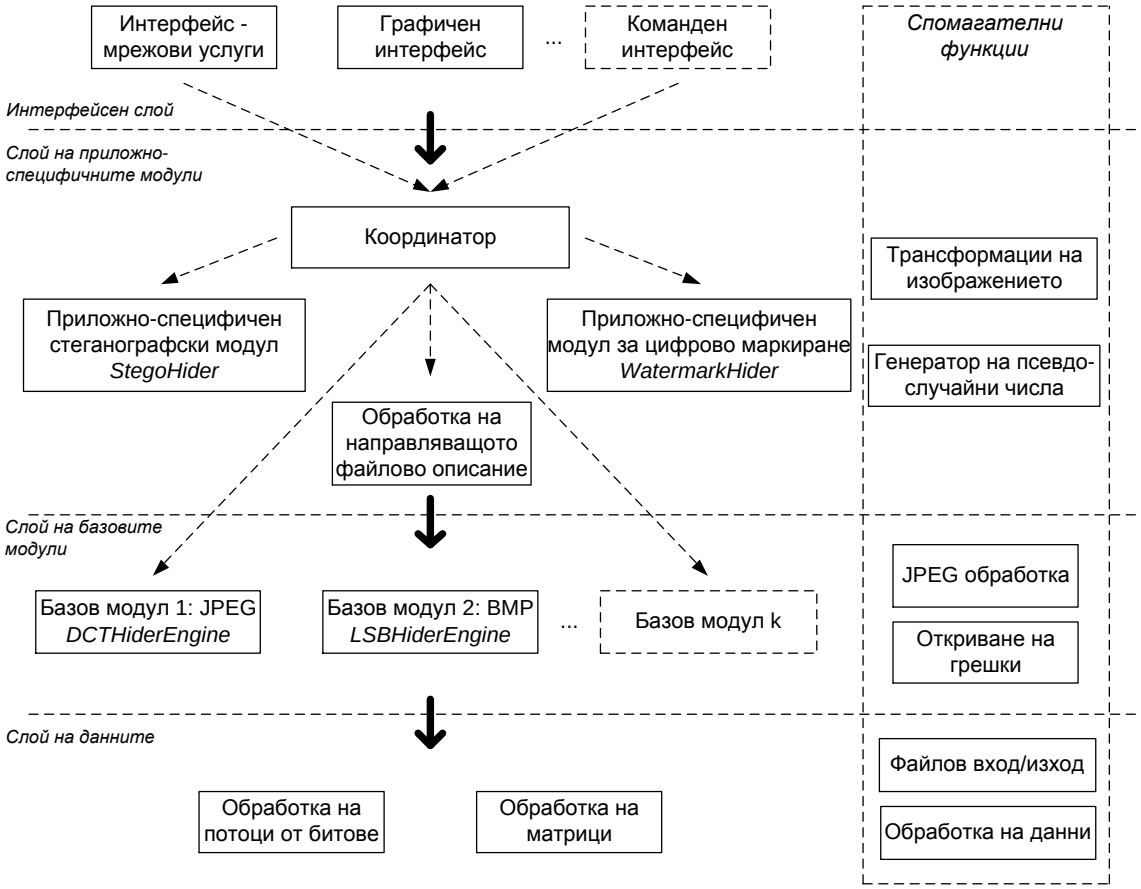
4.1 Архитектура на програмната система

Архитектурата на програмната система е представена на Фиг. 40. Тя се състои от множество компоненти, групирани в четири слоя – слой на данните, слой на базовите модули, слой на приложно-специфичните модули и слой на потребителския интерфейс (интерфейсен слой). Също така няколко функционални блока, принадлежащи на различни слоеве, са групирани в пространство на имената (англ. namespace) на *спомагателните функции*. Те допълват вградената функционалност, предлагана от библиотеките на .NET платформата и програмния език C++.

Слоят на данните съдържа всички функции, свързани с базовото управление и *обработка на данните*, както и *файловия вход/изход*. Той предоставя разнообразни функции за преобразуване на данните и съдържа дефинициите на два нови типа данни - *поток от битове* и *матрица*. *Потокът от битове* позволява лесен достъп до отделните битовете, от които е съставена поредица от двоични данни и по същество е подобен на потоците от байтове, предлагани от библиотеките на .NET платформата и програмния език C++. Типът данни *матрица* позволява различни манипулации на матрици, чрез които често се представят обработваните изображения. *Обработката на данните* включва множество функции, необходими при работа с масиви, битови маски и при промяна на типовете на данните. *Файловият вход/изход* съдържа функции за проверка на типовете файлове, промяна на имената им и работа с временни файлове.

Слоят на базовите модули реализира логиката за криене на данни в мултимедия от ниско ниво. Той се състои от базовите модули на всеки от описаните в Глава 3 модулни методи. Всеки базов модул реализира свойствата на методите, имащи отношение към даден формат на изображението (в този случай JPEG и като база за сравнение PNG/BMP). Към този слой принадлежат и компонент, осигуряващ подобрена функци-

оналност за откриване на грешки при работа с матрици и числа с плаваща запетая, както и компонент, отговорен за извършването на различни етапи от JPEG обработката на изображения.



Фиг. 40: Програмна архитектура – общ поглед

Слоят на приложно-специфичните модули съдържа всички приложно-специфични модули на методите за криене на данни, както и т.нар. координатор. Приложно-специфичните модули придават на методите свойства от високо ниво, свързани с конкретна област на приложение. *Координаторът* е специален компонент, който се грижи за комбинирането на базови и приложно-специфични модули в цялостен метод, осигуряващ свойствата, необходими на конкретен потребител за конкретна задача. *Обработката на направляващото описание* на вгражданите файлове се извършва в отделен компонент. Той също така реализира компресията, криптирането и прилагането на кодове за корекция на грешки. *Генераторът на псевдослучайни числа* реализира функции за генериране на различни типове псевдослучайни числени стойности. В отделен компонент се поддържат редица *трансформации на изображението* като преобразувания на цветови пространства, прилагане на филтри, създаване на хистограми.

Интерфейсният слой реализира с помощта на различни компоненти комуникацията между краен потребител или външно приложение и модулните методи за криене на

данни. Компонентите в този слой определят необходимите параметри за работата на методите, стартират и спират тяхната дейност и получават крайните резултати от процесите на кодиране и декодиране на данни и водни знаци.

Както е показано на Фиг. 40, компонентите от горните слоеве използват функционалността, предоставяна от компонентите от долните слоеве. На границите между слоевете компонентите комуникират помежду си, като горните слоеве предават параметри на долните слоеве и получават от тях резултати, които обработват и предават нагоре. Взаимодействията между слоя на приложно-специфичните модули и слоя на базовите модули, както и между слоя на базовите модули и слоя на данните, водят до цялостната реализация на представените в предишната глава модулни методи за криене на данни. Взаимодействието между интерфейсния слой и слоя на приложно-специфичните компоненти прави възможно използването на методите от крайни потребители или външни приложения в различни среди с помощта на вече утвърдили се технологии.

Следващите глави описват в детайли съставните елементи и начина на работа на компонентите в пространството на имената на *спомагателните функции*, както и на всеки един от представените по-горе слоеве. Повечето от компонентите се реализират програмно чрез един или няколко съответни класа, като компонентите от слоя на данните и слоя на базовите модули се реализират предимно чрез класове, написани на C++, а компонентите от слоя на приложно-специфичните модули и интерфейсния слой се реализират предимно чрез класове, написани на Visual Basic .NET (съкратено VB.NET) или C#. Тъй като програмната система работи върху .NET платформата, то всеки клас, написан на C++ се прави достъпен чрез специална обвивка (англ. wrapper) за класовете, написани на VB.NET или C#.

4.2 Пространство на имената на спомагателните функции

Компонентите, принадлежащи на пространството на имената на спомагателните функции реализират някои основни функции, които не са част от .NET платформата или C++, но са необходими за реализацията на модулните методи за криене на данни. Тези функции трябва да бъдат достъпни за всички компоненти, показани на Фиг. 40. Някои от тези функции се реализират в класове, написани на VB.NET, а други в класове, написани на C++, в зависимост от тяхната роля и област на основно приложение.

Компонентът за *обработка на данни* (програмна реализация чрез C++) предоставя функции за преобразуване и замяна на масиви от данни. Реализирани са специални функции за преобразуване на прости типове данни (напр. цели числа в даден интервал) с поддръжка на препълване на типа (англ. overflow), както и някои основни математически функции, които предоставят фин контрол върху закръгляването на скаларни стойности.

Компонентът за *файлов вход/изход* (програмна реализация чрез C++) управлява всички файлови операции. Той обработва отварянето и създаването на файлове с изоб-

ражения и предоставя функции за разпознаване на типа на файла, извършване на сравнение между два файла и манипулация на файловото име. Освен това този компонент реализира прехвърлянето на файлове от зададени Интернет URL адреси, което е неизбежно при използването на методите в Интернет-базирана среда.

Компонентът за *откриване на грешки* (програмна реализация чрез C++ и VB.NET) съдържа функции за сравнение на масиви и матрици, които улесняват проследяването на грешки в методите за криене на данни. Компонентът открива местоположението и големината на разликите в елементи на масиви и матрици, които би трябвало да бъдат идентични. Също така се поддържат функции за оценка на разликите между числа с плаваща запетая.

Компонентът за *JPEG обработка* (програмна реализация чрез C++) реализира функции от ниско ниво, които пряко касаят JPEG формата за кодиране на изображения. Той извършва трансформации на цветови пространства, необходими за кодирането на изображенията в JPEG формат, и реализира оптимизирани изчисления на дискретната косинусова трансформация. Освен това компонентът извършва анализ на блоковете с големина 8×8 пиксела, които съставят JPEG изображението. Той пресмята броя на блоковете и стойностите на техните DCT коефициенти на избрани етапи от алгоритъма за JPEG компресия и декомпресия (за подробна информация относно JPEG стандарта, вж. раздел 3.2.2). При програмната реализация се използва популярната библиотека с отворен код *libjpeg*, разработвана от IJG [121], [122].

Генераторът на псевдослучайни числа (програмна реализация чрез C++ и VB.NET) предоставя методи за генериране на псевдослучайни пермутации, необходими за разбъркването на данните при вграждането им в блоковете на изображението. Като компонент с общо предназначение, той може да се използва и за генерирането на отделни псевдослучайни цели, реални или булеви стойности.

Компонентът за *трансформации на изображението* (програмна реализация чрез C++ и VB.NET) реализира различни трансформации, преобразуващи изображения в масиви и масиви в изображения. Той реализира трансформации между различни цветови пространства и съдържа функции за анализ на изображението. Компонентът може да конвертира .NET обекти, представляващи изображения, в двумерни масиви и обратно. Двумерните масиви съдържат представянето на изображението в зададено цветово пространство. Това е необходимо, защото платформата .NET има свои собствени обекти за съхранение на информацията за изображенията. Тези обекти улесняват използването на изображенията на високо ниво в интерфейсния слой, но не са подходящи за директно използване от отделните модули при обработката на изображения на ниско ниво.

4.3 Слой на данните

Слоят на данните съдържа два компонента, които осигуряват поддръжка съответно на потоци от битове и на матрици. Тези два типа данни играят съществена роля при реализацията на методите за криене на данни и за тях не се предлага добра вградена поддръжка от платформата .NET или C++. Това прави необходимо създаването на допълнителни компоненти, които реализират двата типа данни и позволяват тяхното лесно използване от останалите слоеве на програмната система. Поддръжката на потоци от битове се реализира програмно от клас, написан на VB.NET, а поддръжката на матрици – от класове, написани съответно на VB.NET и C++.

Платформата .NET не поддържа потоци от битове. Основополагащата единица на потоците от данни, реализирани в .NET, обикновено е байт или Unicode-символ. Тези единици позволяват обработката на двоичните и текстовите файлове, предавани през Интернет, но са прекалено големи за целите на криене на данни в мултимедия поради следните причини:

1. Големината на свободното пространство, което може да се използва за криене на данни, е силно ограничена. Ако направляващото файлово описание и съдържанието на вгражданите данни трябва да бъдат подравнявани до граници на байт, се акумулират загуби на полезно пространство, които могат да достигнат до няколко байта. Това не е желателно – особено при методи за цифрово маркиране.
2. Използването на кодове за корекция на грешки предполага, че данните, които се кодират, се състоят от битове, а не от байтове или Unicode-символи.
3. Често се случва цял байт или Unicode-символ да не може да се вгради в един единствен блок от изображението. В такъв случай байтът или Unicode-символът трябва да се разглеждат като поредица от битове, които се разпределят между няколко блока от изображението.

Следователно, основополагащата единица на използваните потоци от данни трябва да се намали от един байт или Unicode-символ до един бит. Тъй като стандартните потоци от данни в .NET не поддържат такава малка единица, е необходимо създаването на нов компонент, реализиращ потоци от битове. Този компонент е реализиран програмно от клас, наречен „Bitstream“, който наследява основния клас „stream” в платформата .NET и осигурява правилната програмна реализация на неговите виртуални методи. Следователно, новият клас, поддържащ потоци от битове, има същите методи като традиционните класове, управляващи потоци от данни в .NET, и може да се използва по аналогичен начин. Освен това са реализирани няколко нови метода, които отговарят за преобразуването от обичайни байт-базирани или Unicode-базирани потоци към потоци от битове и обратно. Такова преобразуване се налага, защото всички взаи-

модействия между методите за криене на данни и външни приложения използват стандартните потоци от данни, предоставени от платформата .NET.

Друга полезна функционалност, интегрирана в новия клас за обработка на потоци от битове, е способността да борави с кодове за корекция на грешки, базирани на алгоритъма на Хеминг, представен в раздел 3.3.2. Реализирани са методи за добавяне на допълнителните битове за корекция на грешки на съответните места в потока от битове, методи за откриването и коригирането на евентуални грешки в потока, както и методи за определяне на общия размер на данните в потока със или без битовете, отговарящи за корекцията на грешки.

Реализацията на алгоритъма за корекция на грешки като част от класа за обработка на потоци от битове позволява простото му използване с помощта само на няколко метода от високо ниво. Това също така осигурява възможност за бърза предварителна оценка на размера на данните преди и след прилагането на алгоритъма за корекция на грешки в случай на промяна на потока от битове.

Компонентът за обработка на матрици е реализиран програмно от клас, наречен „Matrix“. Той съдържа реализацията на математическото понятие матрица – двумерна таблична структура, състояща се от реални числа, за която са дефинирани операции като събиране, изваждане, умножение и др. Този клас е необходим, защото цифровите изображения по своята същност са матрици от пиксели. Много процедури за обработка на изображения като трансформациите на цветови пространства или някои от етапите на алгоритъма за компресия JPEG работят или върху цялата матрица на изображението или върху по-малки нейни под-матрици. Реализацията на математическа матрица в класа съдържа обичайните матрични операции събиране, изваждане, умножение, транспозиция, сравнение, както и извличането или промяната на под-матрици. Поддържа се и преобразуване към и от стандартни двумерни масиви от реални (тип *double*) или цели (тип *int*) числа.

Освен това класът за обработка на матрици съдържа няколко метода, разработени специално за употреба в алгоритмите за криене на данни. Реализирани са методи за закръгляване на всеки елемент от дадена матрица по един от няколко специфични начина, както и методи за поелементно умножение или делене на две матрици (всеки елемент $p_{i,j}$ от дадена матрица с размерност $M \times N$ се умножава по или разделя на съответстващия му елемент $q_{i,j}$ от друга матрица, имаща същата размерност $M \times N$). Такива методи са необходими например при извършването на стъпката на квантизация в JPEG алгоритъма (вж. раздел 3.2.2). Също така могат да се наложат ограничения на стойностите на елементите в матрицата – например всеки елемент да принадлежи на интервала *[0; 255]*, което е полезно при трансформации на цветовите пространства.

Двата нови типа данни – потоци от битове и матрици – се използват от слоевете на базовите и приложно-специфичните модули, за да се реализират и отделят в собствени компоненти някои критични операции от ниско ниво, свързани с интензивни изчисления. Оптимизираната и добре тествана реализация на компонентите чрез обектно-

ориентирани класове, увеличава производителността и намалява сложността на модулите, използвани в методите за криене на данни.

4.4 Слой на базовите модули

Слоят на базовите програмни модули съдържа различни компоненти, всеки от които може да се използва като базов модул „от ниско ниво" в модулните методи за криене на данни. Базовите модули се реализират програмно от класове, написани на VB.NET или C++. Всеки такъв клас трябва да реализира публичните абстрактни методи *PrepareToEncode, FinishEncode* и *Decode* на абстрактния клас *BaseHiderEngine* (вж. раздел 3.1).

На този етап са разработени два различни базови модула, реализирани програмно в класовете *DCTHiderEngine* и *LSBHiderEngine*. Класът *DCTHiderEngine* (реализиран програмно чрез C++) съдържа кода на базовия модул, описан в раздел 3.2, и е създаден, за да се използва с JPEG изображения. Класът *LSBHiderEngine* (реализиран програмно чрез VB.NET) реализира базов модул за използване с PNG/BMP (или други некомпресирани или беззагубно компресирани) изображения. Малка примерна част от кода на този клас е показана на Фиг. 41.

```
215     Public Overrides Sub Analyze(ByVal DecryptionPassword As String, ByRef WidthInBlocks As UInte
216         WidthInBlocks = _Cols : HeightInBlocks = _Rows : BlockSize = 1
217         Dim BlockNum As UInteger = WidthInBlocks * HeightInBlocks
218         HidingBoundariesTotalSum = BlockNum * MinHidingBoundary
219         If _ForceGreyLevel Then
220             MinHidingBoundary = 8 - MSBNotForHide
221             MaxHidingBoundary = MinHidingBoundary
222             NumOfAnalysisChannels = 1
223         Else
224             MinHidingBoundary = 24 - 3 * MSBNotForHide
225             MaxHidingBoundary = MinHidingBoundary
226             NumOfAnalysisChannels = 3
227         End If
228         ReDim HidingBoundaries(BlockNum - 1)
229         For i As Integer = 0 To BlockNum - 1
230             HidingBoundaries(i) = MinHidingBoundary
231         Next i
232         AnalysisChannels = New Short(NumOfAnalysisChannels - 1)(,)(,) {}
233         If _ForceGreyLevel Then
234             Util.CreateGreyArrayFromByteArray(_ImageBytes, _Cols, _Rows, Grey, isGrey)
235             AnalysisChannels(0) = New Short(HeightInBlocks - 1, WidthInBlocks - 1)(,) {}
236             For i As UInteger = 0 To HeightInBlocks - 1
237                 For j As UInteger = 0 To WidthInBlocks - 1
238                     AnalysisChannels(0)(i, j) = New Short(0, 0) {}
239                     AnalysisChannels(0)(i, j)(0, 0) = Grey(i, j)
240                 Next j
241                 If StopFlag <> 0 Then Exit Sub
242                 Progress += 1
243             Next i
244             If StopFlag <> 0 Then Exit Sub
245         Else
246             Dim Offset As UInteger = 0
247             AnalysisChannels(0) = New Short(HeightInBlocks - 1, WidthInBlocks - 1)(,) {}
```

Фиг. 41: Примерна част от кода на класа „LSBHiderEngine", реализиран чрез VB.NET

Докато класът *LSBHiderEngine* има сравнително проста реализация и съдържа само няколко допълнителни функции за промяна на битове и за стеганализ в допълнение

към трите задължителни публични метода, то класът *DCTHiderEngine* е значително по-сложен. Една от причините за това се крие в стъпките, които съставят алгоритъма за загубна компресия JPEG. Те трябва да бъдат изпълнени от кода на базовия модул до вкл. последната стъпка на загубно кодиране, описана в раздел 3.2.2. Поради използването на редица математически трансформации са необходими допълнителни функции в кода за извършването на нужните изчисления. Тези функции и тяхната ефективност и бързина на изпълнение са от особено значение за метода *PrepareToEncode*, който пресмята максималното количество битове, които могат да се вградят във всеки блок чрез анализиране на съответните квантизирани DCT коефициенти.

```
915 void DCTHiderEngine::Analyze(unsigned int* WidthInBlocks, unsigned int* HeightInBlocks, unsigned short* BlockSize, unsigned
916     if(WidthInBlocks != NULL) *WidthInBlocks = BlockNumCol;
917     if(HeightInBlocks != NULL) *HeightInBlocks = BlockNumRow;
918     if(BlockSize != NULL) *BlockSize = BLOCK;
919     if(HidingOrder != NULL) *HidingOrder = HIDING_ORDER_JPEG_ZIG_ZAG;
920     if(MinHidingBoundary != NULL) *MinHidingBoundary = DCT_MinHidingBoundary + 1 - DCT_BeginHidingAtCoeffIndex;
921     if(MaxHidingBoundary != NULL) *MaxHidingBoundary = BLOCK_LENGTH - 1 - DCT_BeginHidingAtCoeffIndex;
922     if(!_AnalysisDone){
923         PrepareQuantMatrices(_JRatio);
924         AnalyzeNoSubsampl(StopFlag, Progress);
925         if(*StopFlag == 0) _DecodeDone = true;
926     }else{
927         *Progress = BlockNumRow * BLOCK;
928     }
929     if(HidingBoundariesTotalSum != NULL) *HidingBoundariesTotalSum = _HidingBoundariesTotalSum;
930 }
931
932 inline void DCTHiderEngine::AnalyzeNoSubsampl(BYTE* StopFlag, unsigned int* Progress){
933     Matrix Block_Red(BLOCK, BLOCK, 0, 255), Block_Green(BLOCK, BLOCK, 0, 255), Block_Blue(BLOCK, BLOCK, 0, 255);
934     Matrix Block_Y(BLOCK, BLOCK), Block_Cb(BLOCK, BLOCK), Block_Cr(BLOCK, BLOCK);
935     Matrix DCTM(BLOCK, BLOCK), DCTM_Div_Rounded(BLOCK, BLOCK);//, DCTM_Rounded(BLOCK, BLOCK); //, DCTM2(BLOCK, BLOCK), DCTM3
936     DeletePrepareDCTCoeffsMatrices();
937     DeleteFinishDCTCoeffsMatrices();
938     DeleteDecodeDCTCoeffsMatrices();
939     DeleteAnalyzeDCTCoeffsMatrices();
940     InitAnalyzeDCTCoeffsMatrices();
941     unsigned int Counter = 0;
942     unsigned int OffsetI = 0;
943     //_HidingBoundariesTotalSum = 0;
944     //*Progress = 0;
945     for(unsigned int i = 0; i < BlockNumRow; i++, OffsetI += BLOCK, (*Progress) += BLOCK){
946         unsigned int OffsetJ = 0;
947         for(unsigned int j = 0; j < BlockNumCol; j++, OffsetJ += BLOCK){
948             if(_JCR == NULL){
949                 Util::ExtractArrayElementsToMatrix(wBMP->_RedM, OffsetI, OffsetJ, BLOCK, BLOCK, &Block_Red);
950                 Util::ExtractArrayElementsToMatrix(wBMP->_GreenM, OffsetI, OffsetJ, BLOCK, BLOCK, &Block_Green);
951                 Util::ExtractArrayElementsToMatrix(wBMP->_BlueM, OffsetI, OffsetJ, BLOCK, BLOCK, &Block_Blue);
952                 JPEGProcessor::CreateYCbCrMatricesFromRGB(Block_Red, Block_Green, Block_Blue, Block_Y, Block_Cb, Block_Cr);
953                 CalculateDCTMatrix(&Block_Y, &DCTM);
954             }else{
955                 Util::ExtractArrayElementsToMatrix(_JCR->GetCoeffsLum(), OffsetI, OffsetJ, BLOCK, BLOCK, &DCTM);
956             }
```

Фиг. 42: Примерна част от кода на класа „DCTHiderEngine”, реализиран чрез C++

Друг важен фактор, който значително повишава сложността на кода на *DCTHiderEngine*, произлиза от изчисленията, необходими за извършването на стъпки 2, 0 и 4 от етапа на завършване на кодирането (*FinishEncode*), описани в раздел 3.2.3. Те осигуряват устойчивостта на вградените данни срещу различните видове JPEG трансформации (компресия, декомпресия и рекомпресия), както и подобряването на качеството на изображението след вграждането на данните. Всяка стъпка се извършва от няколко функции, координирани от метода *FinishEncode* с цел намаляване на изчислителното време и поддържане на минимална разлика между оригиналното изображение и изображението, съдържащо вградени данни. Отделните стъпки образуват своеобразен тип контролни точки, които често изискват значително количество процесорно

време и задължително трябва да бъдат преминати успешно, преди да се премине към следваща стъпка. След като последната стъпка завърши обработката на изображението, то се предава към по-горния слой на приложно-специфичните модули.

Сложността на алгоритмите за криене на данни и изчислителното време, необходимо за тяхното изпълнение, вървят почти винаги ръка за ръка. Затова е важно критичните секции от кода на класовете, реализиращи програмно базовите модули като *DCTHiderEngine*, да се кодират чрез език от сравнително ниско ниво – C++ или дори асемблер. Това осигурява максимална бързина на изпълнение. Тъй като тези секции от кода по правило не съдържат интерфейси и използват основно математически и матрични трансформации, то те могат да бъдат пренаписани сравнително лесно на друг програмен език и достъпът до тях може да се извършва чрез клас-обвивка (англ. wrapper class) от високо ниво, написан на .NET. По този начин слоят на приложно-специфичните модули и интерфейсният слой могат да използват предимствата на гъвкавостта и интеграцията с други платформи и системи, предоставени от платформата .NET, без да е необходимо да жертват за тази цел предимствата относно производителността, характерни за програмните езици от по-ниско ниво. Примерна част от кода на класа *DCTHiderEngine*, написан на C++, е показана на Фиг. 42.

Характерна за разработените базови модули е разликата в сложността, скоростта на изпълнение и необходимата памет между процесите на кодиране и декодиране. Декодирането на вградените данни обикновено е по-просто, по-бързо и изисква по-малко количество RAM-памет в сравнение с кодирането. Програмната реализация на декодирането се нуждае от по-малко редове програмен код. Основните причини за тази разлика са:

1. По време на процеса на кодиране трябва да бъдат реализирани и гарантирани свойствата на модулния метод, като например устойчивостта на вградените данни срещу JPEG трансформации.
2. По време на кодирането, трябва да се определи и извърши подходящата оптимизация за подобряване на качеството на крайното изображение.

По време на процеса на декодиране свойствата на метода вече са гарантирани. Оптимизация на качеството на изображението не е необходима и единствената задача, която трябва да се изпълни е същинското извличане на вградените данни от изображението. Гореописаната особеност се наблюдава също при процесите на кодиране и декодиране в библиотеките за възпроизвеждане и създаване на видео съдържание (англ. codecs) [123]. Значителни оптимизации на програмния код и мощен централен и/или графичен процесор са необходими, за да се направи възможно извършването на видео кодиране в реално време. От друга страна, декодирането и визуализирането на видео данните са значително по-бързи процеси и могат лесно да се извършват в реално време.

4.5 Слой на приложно-специфичните модули

Слоят на приложно-специфичните модули се състои от компоненти, реализиращи приложно-специфичните модули за криене на данни, компонент за обработка на направляващото файлово описание и координаторен компонент. Приложно-специфичните модули се използват като модули от високо ниво в модулните методи за криене на данни и се реализират програмно от класове, написани на VB.NET. Всеки такъв клас наследява абстрактния клас *BaseHider* и предоставя програмна реализация на публичните методи *Encode* и *Decode* (вж. раздел 3.1).

Разработени са общо два различни приложно-специфични модула: един за целите на стеганографията и един за целите на цифровото маркиране. Когато тези модули се използват в комбинация с базовия модул, създаден за вграждане на данни в JPEG изображения – *DCTHiderEngine*, те реализират съответно стеганографския метод за криене на данни и метода за цифрово маркиране. Двата модула се реализират програмно съответно от класовете *StegoHider* class и *WatermarkHider*.

```
23  Public Class StegoHider
24
25      Protected _Head As Header
26      Protected _HiderEng As BaseHiderEngine
27      Protected _StreamRND, _BlockRND, _AuxRND As RandomGenerator_Base
28      Protected _BlockNumRow, _BlockNumCol As UInteger, _BlockSize As UShort
29
30      Protected Const BlockRandomization As Boolean = True
31
32      Public Function Encode(Optional ByVal EncryptionPassword As String = Nothing, Optional ByRef St
33          If _HiderEng Is Nothing Then Throw New Exception("Hider Engine not set.")
34          If _Head Is Nothing Then Throw New Exception("Header stream not set.")
35          Dim HidingBoundaries() As Byte = Nothing
36          Dim HidingBoundariesTotalSum As UInteger, MaxHidingBoundary As Byte
37          _HiderEng.PrepareToEncode(_BlockNumCol, _BlockNumRow, HidingBoundaries, , _BlockSize, , Max
38          If StopFlag <> 0 Then Return Nothing
39          Dim BlockLen As UInteger = _BlockNumRow * _BlockNumCol
40          Dim HiddenBits(BlockLen - 1)() As Byte, InfoHiddenTill(BlockLen - 1) As Byte
41          Dim TempBit As Short, flFinished As Boolean
42          Dim perm(BlockLen) As UInteger
43          Dim kOffset As UInteger = 0
44          If BlockLen > 0 Then
45              For i As UInteger = 0 To BlockLen - 1
46                  HiddenBits(i) = New Byte(HidingBoundaries(i) - 1) {}
47              Next i
48              For k As Byte = 1 To MaxHidingBoundary
49                  If BlockRandomization Then _StreamRND.createPermutationI(BlockLen, perm) Else Randor
50                  Dim Offset As UInteger = 0, j As UInteger
51                  For i As UInteger = 0 To _BlockNumRow - 1
52                      For j = 0 To _BlockNumCol - 1
53                          If Not flFinished Then
```

Фиг. 43: Примерна част от кода на класа „StegoHider", реализиран чрез VB.NET

Класът *StegoHider* съдържа единствено задължителните публични методи, които осигуряват поддръжка на разпределянето на двоичните данни между блоковете на изображението, идентифицирани от използвания базов модул. Процесът на разпределяне на данните е бърз и изисква сравнително малко количество код и програмни ресурси. Времето за изпълнение на методите от приложно-специфичния модул обикновено съставлява малка част от времето, необходимо за изпълнение на програмния код в

базовия модул. Примерна част от кода на класа, написан на VB.NET, е показана на Фиг. 43.

За разлика от стеганографския приложно-специфичен модул, класът *WatermarkHider* съдържа по-сложен програмен код. В допълнение към двата задължителни публични метода, той използва няколко допълнителни функции, които контролират динамичното разделяне на изображението на макроблокове и реализират откриването на променени области (с точност до един микроблок) от изображението по време на процеса на декодиране (вж. раздели 0 и 3.4.3). Тъй като този приложно-специфичен модул добавя две нови свойства към цялостния модулен метод за криене на данни, програмната реализация е по-бавна в сравнение с тази в класа *StegoHider*. Въпреки това, тя има значително по-кратко време за изпълнение от методите на базовия модул, поддържащ JPEG изображения.

```
329     Public Sub SetStegoHiderProperties(Optional ByVal ErrorCorrection As Boolean = True, Optional ByVal Encryption As Boolean = Tr
330         _ErrorCorrection = ErrorCorrection
331         _Encryption = Encryption
332         _EncAlg = EncryptionAlgorithm
333         _SecureDataHiding = SecureDataHiding
334         _SaveFileName = SaveFileName
335         _Head = Nothing
336     End Sub
337
338     Public Sub SetWatermarkHiderProperties(Optional ByVal ErrorCorrection As Boolean = True, Optional ByVal Encryption As Boolean
339         _ErrorCorrection = ErrorCorrection
340         _Encryption = Encryption
341         _EncAlg = EncryptionAlgorithm
342         _SecureDataHiding = SecureDataHiding
343         _SaveFileName = SaveFileName
344         _Head = Nothing
345     End Sub
346
347     Public Sub CreateHiderEngine()
348         If _Heng IsNot Nothing Then Exit Sub
349         Select Case _HiderEngineMode
350             Case enumHiderEngineMode.DCT
351                 _Heng = New DCTHiderEngine(_ImageBytes, _ImageWidth, _ImageHeight, _WithstandJPEGCompressionRatio, _JPCoeffReader)
352             Case enumHiderEngineMode.LSB
353                 _Heng = New LSBHiderEngine(_ImageBytes, _ImageWidth, _ImageHeight, _ForceGreyLevelBMPProcessing)
354         End Select
355     End Sub
356
357     Public Sub Hide(Optional ByRef StopFlag As Byte = 0, Optional ByRef Progress As UInteger = 0)
358         CreateHiderEngine()
359         CreateHeader()
360         Select Case _HiderMode
361             Case enumHiderMode.Stego
362                 _HiderClass = New StegoHider(_Heng, _Head)
363             Case enumHiderMode.Watermark
364                 _HiderClass = New WatermarkHider(_Heng, _Head)
365         End Select
366         If _SecureDataHiding Then
367             _ResImg = _HiderClass.Encode(_Pass, StopFlag, Progress)
368         Else
369             _ResImg = _HiderClass.Encode(Nothing, StopFlag, Progress)
370         End If
371     End Sub
```

Фиг. 44: Примерна част от кода на координатора, реализиран чрез VB.NET

За разлика от базовите модули, процесите на кодиране и декодиране на данни в разработените приложно-специфични модули си приличат по отношение на сложност на кода, бързина на изпълнение и необходима памет. Частите от модула, които се използват при кодиране и при декодиране, съдържат приблизително еднакъв брой редове програмен код и се нуждаят от еднакво време за изпълнение. Тъй като производителността на кода в приложно-специфичните модули не е от такова критично значение както при базовите модули, то реализацията им чрез VB.NET може да се използва при внедряването на програмното осигуряване за реални приложения. С това се цели из-

ползването на другите предимства, характерни за програмните платформи от високо ниво – гъвкавост при извършване на промени, лесно пренасяне към други операционни системи и лекота при отстраняване на грешки в кода.

Друг важен компонент, принадлежащ към слоя на приложно-специфичните модули, е този за обработка на направляващото файлово описание. Той е отговорен за съставянето, добавянето, анализа и отстраняването на направляващото описание в/от двоичните данни. Освен това той контролира употребата на опционалните кодове за корекция на грешки. Всички съществени резултати, предоставени от този компонент, се предават на координатора. Програмната реализация се осъществява в този случай чрез VB.NET.

Координаторът е последният важен компонент от слоя на приложно-специфичните модули. Неговата основна задача е управлението на цялостните процеси на криене и извличане на данни. Той осигурява връзката между интерфейсния слой, останалите компоненти от слоя на приложно-специфичните модули и компонентите от слоя на базовите модули. Примерна част от програмната реализация на координатора като клас, написан на VB.NET, е показана на Фиг. 44. Координаторът има следните задачи:

1. Предоставяне на възможност за контрол на важните параметри на процесите за криене и извличане на данни като напр. максималното ниво на компресия, срещу което вградените данни са устойчиви, или употребата на незадължителните кодове за корекция на грешки;
2. Избор на подходяща комбинация от базови и приложно-специфични модули, които да съставят цялостния метод за криене на данни. Този избор зависи от информацията, предоставена от интерфейсния слой за параметрите на текущото приложение и/или предпочитанията на крайните потребители;
3. Настройване и стартиране на обработката на направляващото файлово описание по отношение на текущия процес на кодиране или декодиране на данните;
4. Предаване на резултатите в подходящ формат (изображение при кодиране, двоични данни и метаданни при декодиране) обратно към интерфейсния слой след завършване на процеса на кодиране или декодиране.

4.6 Интерфейсен слой

Интерфейсният слой осъществява връзката между методите за криене на данни в мултимедия и външния свят –автоматизирани мрежови приложения и/или реални потребители. Могат да бъдат реализирани различни типове потребителски интерфейси, подходящи за различни видове приложения и потребители: самостоятелните десктоп приложения могат да се възползват от графичен потребителски интерфейс (англ. graphical user interfac, GUI) или команден интерфейс (англ. command-line interface,

CLI), докато Интернет-базираните приложения се нуждаят от подходящ интерфейс за мрежови услуги (англ. web service interface).

Интерфейсите, реализирани до момента, са:

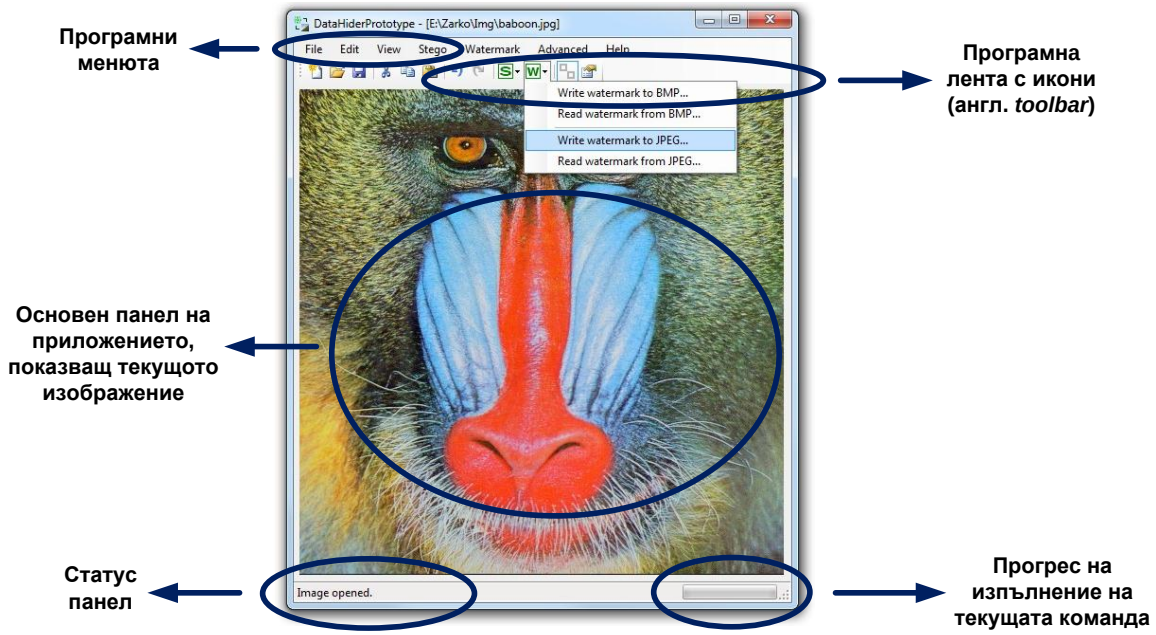
1. Графичен потребителски интерфейс за използване на методите за криене на данни под формата на самостоятелно десктоп приложение;
2. Графичен потребителски интерфейс, позволяващ старт и анализ на групови обработки на мултимедийни файлове и файлове с данни;
3. Приложен програмен интерфейс (англ. application programming interface, API) за мрежови услуги, подходящ за употреба в Интернет-базирани сценарии.

Следващите раздели описват накратко всеки един от гореизброените интерфейси.

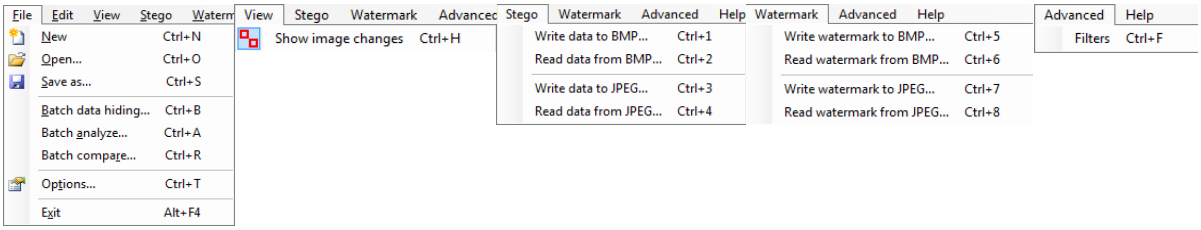
4.6.1 Графичен потребителски интерфейс

Разработеният графичен потребителски интерфейс е показан на Фиг. 45 и Фиг. 27 в раздел 0. Потребителят може да види обработваното изображение преди и след прилагането на всеки метод за криене на данни в основния панел на приложението и може да стартира команди посредством програмните менюта или програмната лента с икони (англ. toolbar), намиращи се в горната част на прозореца. След стартирането и завършването на всяка команда, както и в случай на появил се проблем, статус панелът в долната лява част на прозореца на приложението дава кратка информация за състоянието на текущо изпълняваната команда. Прогресът на изпълнение на командата се показва в долната дясна част на прозореца на приложението.

Някои от по-важните програмни менюта на графичния потребителски интерфейс са показани в детайли на Фиг. 46. Меню *File* съдържа обичайните команди *New*, *Open*, *Save as* и *Exit*, както и връзки към графичните интерфейси, позволяващи старт, анализ и сравнение на групови обработки. Команда *Options* от меню *File* показва диалоговия прозорец във Фиг. 47. Той дава достъп до някои важни настройки на методите за криене на данни. В разделите *Stego* и *Watermark* се съдържат идентични настройки. Настройката *Use error correction* контролира употребата на опционалните кодове за коркция на грешки. Настройката *Save file name* определя дали името на файла ще бъде включено в стеганографското направляващо описание на файла. Настройката *Content encryption* определя използвания криптографски метод. Настройката *Secure data hiding* прави процеса на криене на данни зависим от парола, зададена от потребителя. Раздел *JPEG* съдържа две настройки. Настройката *Resistant to JPEG compression down to* определя най-високото ниво на JPEG компресия, срещу което базовият модул *DCTHiderEngine* гарантира устойчивост на вградените данни. Настройката *Save JPEG files to disk with a quality of* определя с какво ниво на JPEG компресия изображенията ще бъдат записвани на твърдия диск при избиране на команда *Save as* от меню *File*.



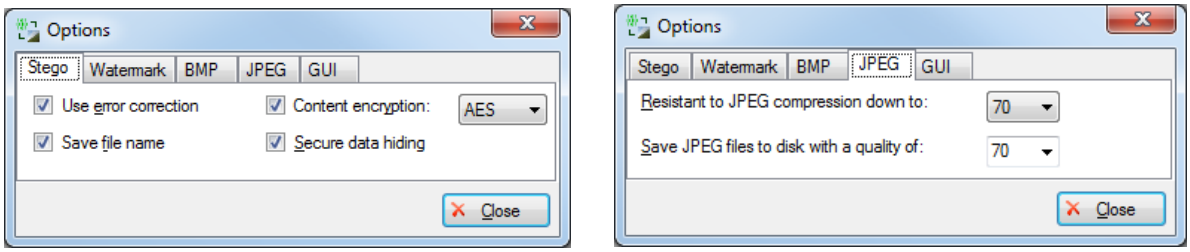
Фиг. 45: Графичен потребителски интерфейс



Фиг. 46: Графичен потребителски интерфейс – програмни менюта

Меню *View* позволява преглед на евентуални промени в изображението, открити от разработения метод за цифрово маркиране. Меню *Stego* съдържа команди, управляващи старта на стеганографските методи за криене на данни с използването на базовите *LSBHiderEngine* (за некомпресирани изображения) или *DCTHiderEngine* (за JPEG изображения). Командата *Write* показва диалогов прозорец за избор на файл с данни за вграждане, задаване на парола и стартиране на кодирането на избрания файл в изображението. Командата *Read* декодира данните, вградени в изображението (след въвеждане на парола), и след това ги записва във файл, чието местоположение и име се избират от потребителя.

Меню *Watermark* контролира методите за цифрово маркиране. То позволява кодирането на нов цифров воден знак в изображението (команда *Write*) или декодирането на вече вграден воден знак от (команда *Read*), използвайки базовите модули *LSBHiderEngine* или *DCTHiderEngine*. Меню *Advanced*, команда *Filters*, дава достъп до някои популярни филтри и средства за хистограмен анализ на изображението.



Фиг. 47: Диалогов прозорец *Options*

4.6.2 Графичен потребителски интерфейс за групови обработки

Груповите обработки се контролират от отделен графичен потребителски интерфейс, състоящ се от три прозореца:

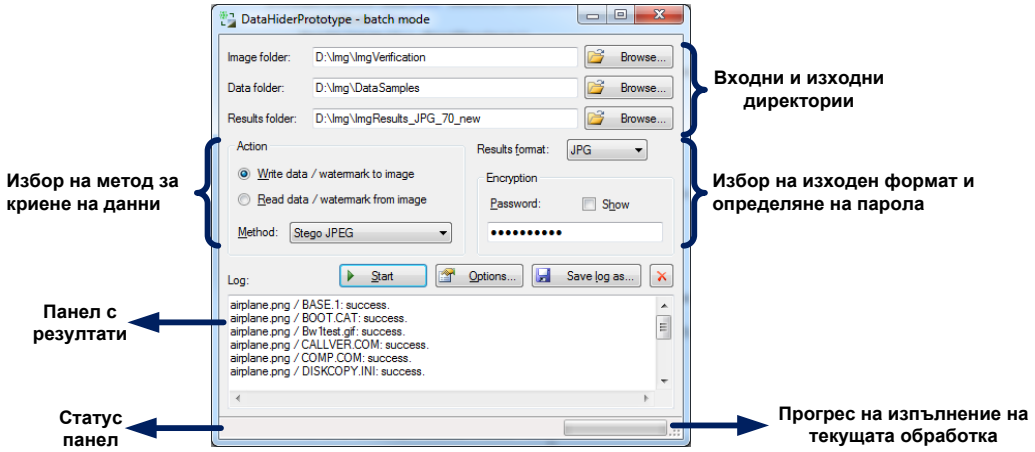
- Първият прозорец контролира същинските групови обработки на изображения и файлове с данни.
- Вторият прозорец съдържа инструменти за статистически анализ, използвани при оценката на методите за криене на данни.
- Третият прозорец съдържа инструменти за сравнение на изображения и файлове с двоични данни.

Прозорецът, контролиращ груповите обработки на изображения и файлове с данни, е показан на Фиг. 48. Полето *Image folder* указва директорията, съдържаща изображенията, които трябва да бъдат обработени. Полето *Data folder* съдържа директорията, където се намират файловете с данни. По време на кодирането алгоритъмът за групова обработка кодира всеки файл с данни от директорията *Data folder* във всяко едно изображение от директорията *Image folder* и записва резултата (изображение) в директорията *Results folder*. Ако в директорията *Image folder* се съдържат M изображения и в директорията *Data folder* се намират N файла, то броят на възможните комбинации между тях е равен на $M \times N$. Това е броят на изображенията с вградени данни, генерирани от алгоритъма за групова обработка и запомнени в директорията *Result folder*.

По време на декодирането алгоритъмът за групова обработка прочита данните, вградени във всяко изображение, съдържащо се в директорията *Image folder*, и ги записва като файлове в директорията *Data folder*. В този случай не се налага да се извършва комбиниране на файлове и броят на прочетените файлове с данни в директорията *Data folder* е равен на броя на изображенията в директорията *Image folder*.

Бутонът за избор (англ. radio button) *Write data / watermark to image* избира провеждането на групово кодиране, а бутонът за избор *Read data / watermark from image* избира провеждането на групово декодиране. Падащото меню *Method* определя метода за криене на данни, който ще се използва по време на груповата обработка. На този етап

може да се избере един от следните четири метода: *Stego LSB, Stego DCT, Watermark LSB, Watermark DCT*. Това са същите методи, които могат да бъдат избрани посредством менютата и лентата с иконите на основния графичен потребителски интерфейс на програмната система. Те са резултат от възможните комбинации между реализираните базови и приложно-специфични модули (вж. раздели 4.4 и 4.5). Падащото меню *Results format* определя графичния формат, в който ще бъдат запомнени изображенията, генерирани по време на процеса на кодиране на данните. Форматите, които се поддържат на този етап са: JPEG, BMP, GIF, PNG и TIFF.

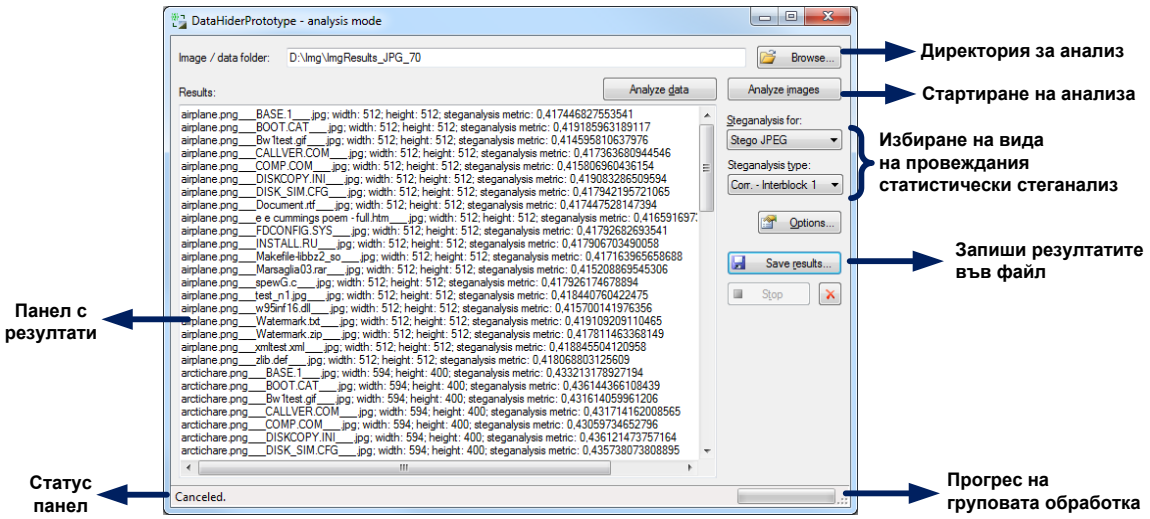


Фиг. 48: Графичен потребителски интерфейс – групови обработки

Трите командни бутона (англ. command button) над панела с резултати имат следната функция: бутонът *Start* стартира изпълнението на груповата обработка, бутонът *Options* показва диалоговия прозорец с настройки, а бутонът *Save log as* записва информацията, изведена в панела с резултати, под формата на текстов файл, чието местоположение се избира от потребителя.

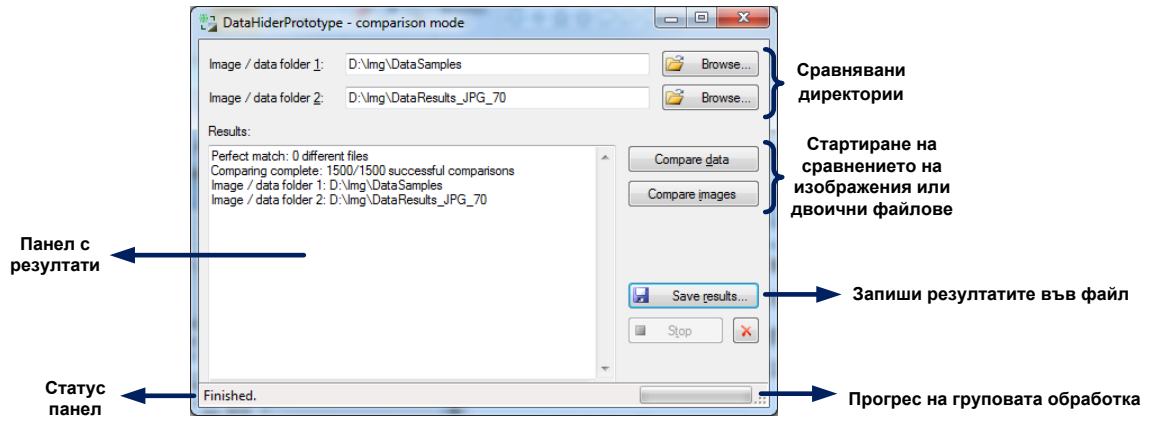
Панелът с резултати показва информация за резултата от обработката на всяка комбинация от изображение и файл с данни. Прозорецът съдържа още статус панел, който показва кое изображение се обработва в дадения момент и кой файл с данни се вгражда в него, ако е стартиран процес на групово кодиране. В долната дясна част на прозореца се изобразява графично прогресът на изпълнение на текущата групова обработка.

Вторият прозорец от графичния интерфейс за групова обработка, съдържащ инструментите за статистически анализ, е показан на Фиг. 49. Неговата основна роля е да позволи детайлния анализ на резултатите от групова обработка. Директорията *Image/data folder* съдържа файловете с изображения или двоични данни, които ще се анализират. Ако файловете съдържат двоични данни, провежданият анализ е прост - в панела с резултати се извежда дължината на файловете в битове. След като всички файлове бъдат изследвани, в панела с резултати се извежда кратко общо описание, съдържащо техния общ брой и средната файлова дължина.



Фиг. 49: Графичен потребителски интерфейс – инструменти за статистически анализ

Анализирането на изображения изисква значително по-сложен анализ - статистически стеганализ, който се дискутира в детайли в следващата глава. В дясната част на прозореца се намират две полета: *Steganalysis for* и *Steganalysis type*, с помощта на които се управлява стеганализа на изображения. Полето *Steganalysis for* определя целевия алгоритъм за криене на данни, който подлежи на изследване, а полето *Steganalysis type* определя вида на стеганализа. Като резултат от статистическия стеганализ се пресмята числова метрика. Тя се извежда в панела с резултати за всяко анализирано изображение. След като всички изображения бъдат изследвани, в панела с резултати се извежда кратка статистическа информация, съдържаща общия брой на изображенията и стойностите на математическо очакване и стандартно отклонение за пресметнатите числови метрики.



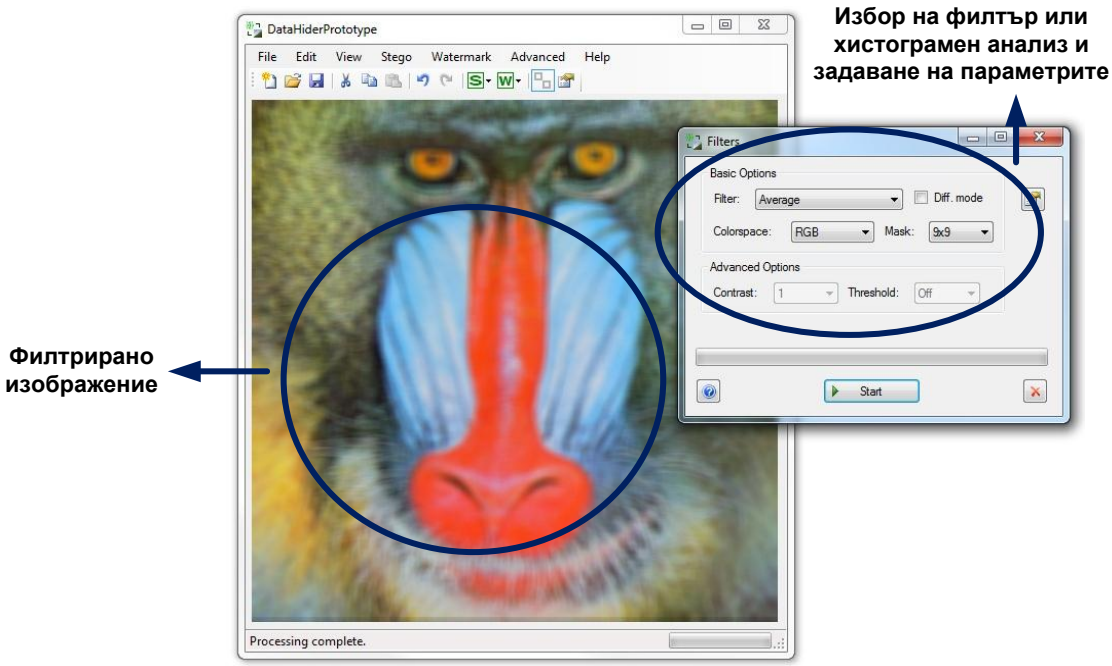
Фиг. 50: Графичен потребителски интерфейс – инструменти за сравнение

Командният бутон *Save results* записва информацията, показвана в панела с резултати, под формата на текстов файл, чието местоположение се избира от потребителя.

Статус панелът показва файла, който се анализира в даден момент. В долната дясна част на прозореца се изобразява графично прогресът по време на анализ на файловете.

Третият прозорец от графичния потребителски интерфейс за групова обработка, съдържа инструментите за сравняване на съдържанието на изображения и двоични файлове (Фиг. 50). Неговата основна роля е да осигури възможност за бърза и надеждна оценка на методите за криене на информация - тема, която се дискутира по-подробно в следващата глава.

Директориите *Image/data folder 1* и *Image/data folder 2* съдържат директориите с изображения или файлове с данни, които ще бъдат сравнявани. Алгоритмите за сравнение намират за всеки файл от директория *1* файлове със съответстващи имена от директория *2*. Когато се намерят такива файлове, всеки един от тях се сравнява с оригиналния файл от директория *1*. Ако файловете съдържат данни, се извършва двоично сравнение с точност до бит и броят на различните битове (ако е по-голям от нула) се показва в панела с резултати. След като всички намерени двойки файлове се сравнят, кратко общо описание се показва в панела с резултати. То показва общия брой на сравнените двойки и общия брой на двойките файлове с намерени различия.



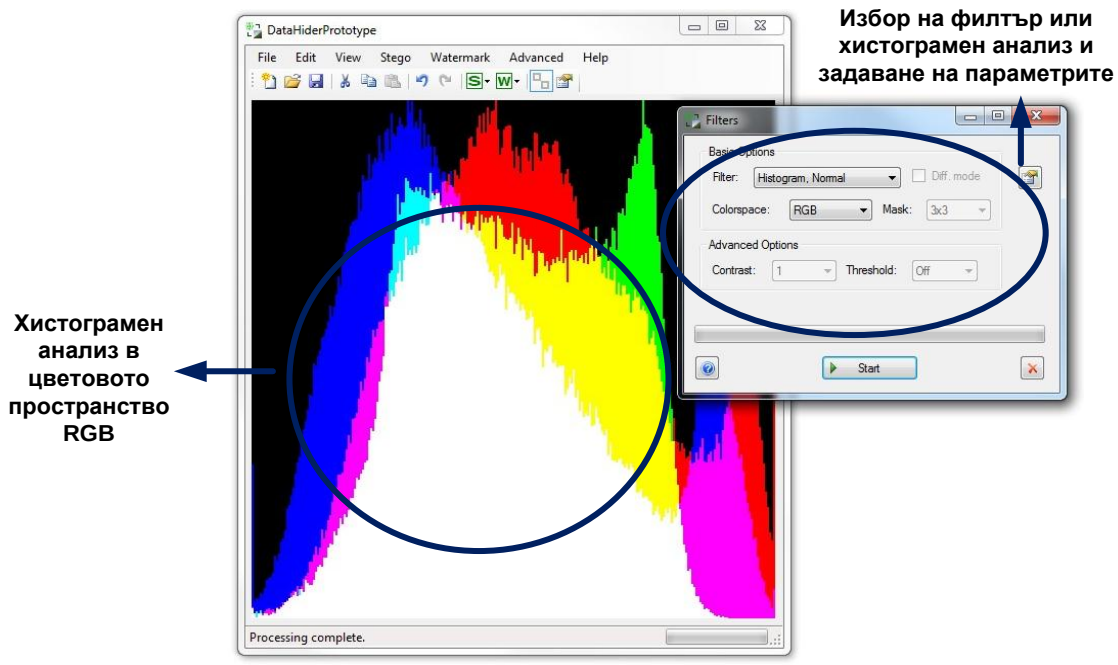
Фиг. 51: Резултат от прилагането на усредняващ филтър с маска 9 × 9

При сравнение на двойки от изображения за всяка двойка се пресмятат два основни статистически индикатора, които описват степента на подобие между изображенията от двойката: MSE (средна квадратична грешка, англ. mean squared error) и PSNR (отношение на пиковия сигнал към шума, англ. peak signal-to-noise ratio). Тези индикатори са описани подробно в раздел 5.2.2.

Стойностите на индикаторите се показват в панела с резултатите с изображения. След като всички намерени двойки се сравнят, средните стойности на MSE и PSNR се пресмятат и показват на екрана. Тези средни стойности описват средното ниво на подобие (или различие) между изображенията, създадени от предишна групова обработка, и оригиналните изображения, несъдържащи вградени данни. По този начин може да се получи информация за влиянието на модулните методи за криене на данни върху качеството на носещите изображения.

В дясната част на прозореца се намират няколко командни бутона, които контролират процесите на сравнение. Бутонът *Compare data* стартира двоично сравнение на файловото съдържание, бутонът *Compare images* стартира сравнение на файлове с изображения, а бутонът *Stop* прекратява възможно най-бързо вече стартиран процес на сравнение. Командният бутон *Save results* записва информацията, показвана в панела с резултати, под формата на текстов файл, чието местоположение се избира от потребителя. Статус панелът показва двойката файлове, които се сравняват в даден момент. В долната дясна част на прозореца се изобразява графично прогреса на сравнение на файловете.

4.6.3 Филтри и хистограмен анализ



Фиг. 52: Резултат от хистограмен анализ в цветовото пространство RGB

От меню *Advanced*, команда *Filters* се извиква прозорец, който дава достъп до редица филтри и средства за хистограмен анализ на изображението, използвани като инструменти при провеждането на стеганализ (вж. раздел 5.4).

Предлаганите филтри са усредняващ филтър, Гаусов филтър, медианен филтър, Винер-Колмогоров филтър и Лапласиан на Гаусиана (разгледани подробно в [46]). Поддържаните маски на филтрите варират от 3×3 до 11×11. Поддържаните цветови пространства са *RGB, Greyscale* (нива на сивото) и YC_bC_r . Последното намира приложение в JPEG стандарта (раздел 3.2.2). Поддържа се и създаването на т. нар. диференчни изображения (англ. difference images). Интензитетът на всеки пиксел на тези изображения се формира от абсолютната стойност на разликата между пиксела със същите координати в оригиналното изображение и пиксела със същите координати във филтрираното изображение.

Резултатът от прилагането на усредняващ филтър с маска 9×9 е показан на Фиг. 51. Резултатът от хистограмен анализ на същото изображение в цветовото пространство RGB е показан на Фиг. 52.

4.6.4 Приложен програмен интерфейс за мрежови услуги

Приложният програмен интерфейс за мрежови услуги е реализиран на базата на технологията ASP.NET на Майкрософт, която е част от платформата .NET. Той предоставя директен достъп до модулните методи за криене на данни, управлявани от координатора. Комуникацията между клиента (произволно Интернет-базирано приложение) и сървъра (съдържащ модулните методи за криене на данни) може да се осъществи чрез т.нар. SOAP протокол (версия 1.1 или 1.2) [124] или обикновени HTTP GET или POST заявки [125].

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <hideDCT xmlns="http://ilchev.net">
      <sourceImage>base64Binary</sourceImage>
      <destinationImageFormat>JPG or BMP</destinationImageFormat>
      <informationToHide>base64Binary</informationToHide>
      <errorCorrection>boolean</errorCorrection>
      <saveFileName>boolean</saveFileName>
      <dataStreamFileName>string</dataStreamFileName>
      <withstandJPEGCompressionRatio>unsignedByte
      </withstandJPEGCompressionRatio>
    </hideDCT>
  </soap:Body>
</soap:Envelope>
```

Фиг. 53: Примерна SOAP заявка

Примерна SOAP заявка за кодиране на данни посредством комбинацията от базовия модул *DCTHiderEngine* и стеганографския приложно-специфичен модул *StegoHider* е показана на Фиг. 53. Името на метода (операцията) на мрежовата услуга (англ. web service operation), който се стартира чрез заявката е *hideDCT*. Параметрите, които клиентът трябва да зададе в SOAP заявката, са както следва:

1. Оригиналното изображение, кодирано в *base64* формат [126] (*sourceImage*). Това кодиране е стандартизиран начин на представяне на произволен поток от байтове чрез малки и големи букви от латинската азбука, цифри и символите „+“,“/“ и „=“ (общо 65 символа, „=“ се използва като контролен символ).
2. Форматът, в който ще бъде върнато крайното изображение, след като кодирането на данните приключи - на този етап JPG или BMP (*destinationImageFormat*).
3. Двоичните данни, които трябва да се вградят в изображението - също кодирани в *base64* формат (*informationToHide*).
4. Булева стойност, която контролира незадължителната употреба на кодове за корекция на грешки: *true*, ако се използва корекция на грешки, *false*, ако корекцията на грешки не е необходима (*errorCorrection*).
5. Булева стойност, която контролира дали името на файла, който съдържа двоичните данни, ще се запомни като част от направляващото файлово описание (*saveFileName*).
6. Името на файла, който съдържа двоичните данни, като текстов низ (англ. string). Този параметър не се използва, ако *saveFileName* (параметър 5) има стойност *false* (*dataStreamFileName*).
7. Максималното ниво на JPEG компресия, което може да се приложи върху крайното изображение, върнато като резултат от заявката, без това да разруши вградените данни (*withstandJPEGCompressionRatio*).

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <hideDCTResponse xmlns="http://ilchev.net">
      <hideDCTResult>base64Binary</hideDCTResult>
    </hideDCTResponse>
  </soap:Body>
</soap:Envelope>
```

Фиг. 54: Примерен отговор на SOAP заявка

Примерен отговор на разгледаната по-горе SOAP заявка е показан на Фиг. 54. Отговорът на заявката предава изображението, съдържащо вградените данни, обратно на клиентското приложение (*hideDCTResult*). Това изображение е кодирано в *base64* формат.

Като алтернатива на SOAP заявката може да се използва HTTP POST заявка (Фиг. 55). За разлика от XML-базирания SOAP протокол, необходимите параметри се предават на интерфейса за мрежови услуги чрез обичайното за HTTP POST заявките разделяне с амперсанди в тялото на заявката. Тази полезна алтернатива опростява създаването на заявката и е удобна за използване, ако клиентското приложение не разполага с библиотеки, осигуряващи поддръжка на SOAP – например JavaScript-базирано приложение, изпълнявано в Интернет браузър.

```
POST /StegiWeb/StegiWeb.asmx/hideDCT HTTP/1.1
Host: xxx.xxx.xxx.xxx
Content-Type: application/x-www-form-urlencoded
Content-Length: length

sourceImage=string&destinationImageFormat=string&
informationToHide=string&informationToHide=string&
errorCorrection=string&saveFileName=string&
dataStreamFileName=string&withstandJPEGCompressionRatio=string
```

Фиг. 55: Примерна HTTP POST заявка

Примерен отговор на HTTP POST заявка е показан на Фиг. 56. XML елементът *base64Binary* съдържа крайното изображение с вградените данни (кодирано в *base64* формат).

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<base64Binary xmlns="http://ilchev.net">base64Binary
</base64Binary>
```

Фиг. 56: Примерен отговор на HTTP POST заявка

Подобни заявка и отговор се използват и в процеса на декодиране на данни, вградени в JPEG изображение (използваният метод на мрежовата услуга е наименуван *unHideDCT*). Също така са реализирани и двойката методи на мрежовата услуга *hideLSB* и *unHideLSB*, които предоставят достъп до стеганографския метод за криене на данни в некомпресирани изображения, съставен от приложно-специфичния модул *StegoHider* и базовия модул *LSBHiderEngine*.

Независимо от протокола, използван при комуникацията (SOAP или HTTP), заявките, идващи от клиентското приложение, първо се обработват от Интернет сървъра IIS и след това се препращат към координатора. В зависимост от конфигурацията на IIS сървъра, няколко клиентски заявки могат да бъдат обработвани едновременно посредством стартиране на няколко копия на кода за криене на данни в мултимедия.

4.7 Заключение

Прототипната реализация на програмната система е разработена с цел да позволи практическото изследване и оценяване на модулния подход и модулните методи за криене на данни, както и за да даде полезни насоки за тяхното подобрение. Като среда за бърза разработка на приложения (англ. rapid application development, RAD), платформата Microsoft .NET се доказа като отличен избор – най-вече заради отличните функции за откриване на грешки и многобройните вградени библиотеки. Едно от най-важните ѝ предимства е възможността не само за подробен оглед но и за промяна на програмния код по време на неговото изпълнение. По този начин могат да се открият причините за сложни за проследяване грешки или проблеми в производителността на кода. Решения на проблеми и възможни подобрения могат да се реализират и тестват на място чрез промяна на вече изпълняващия се код. Тъй като най-значимите и трудни за отстраняване проблеми се дължат на грешки при разработката на самите методи за криене на данни, а не на грешки при тяхното програмно изпълнение, това предимство е съществено.

Единственият недостатък на реализацията от високо ниво в .NET среда, е сравнително бавната скорост на изпълнение на кода в сравнение с езици от по-ниско ниво като C/C++ и асемблер. Следователно, при внедряване на програмното осигуряване транслирането на някои критични секции от кода (най-вече на базовите модули) към език от по-ниско ниво би донесло значително подобрение в скоростта на изпълнение, като първоначалните тестове показват фактор на повишаване на скоростта около две.

В процеса на програмната реализация на модулните методи, се открояват следните резултати с оригинален авторски принос:

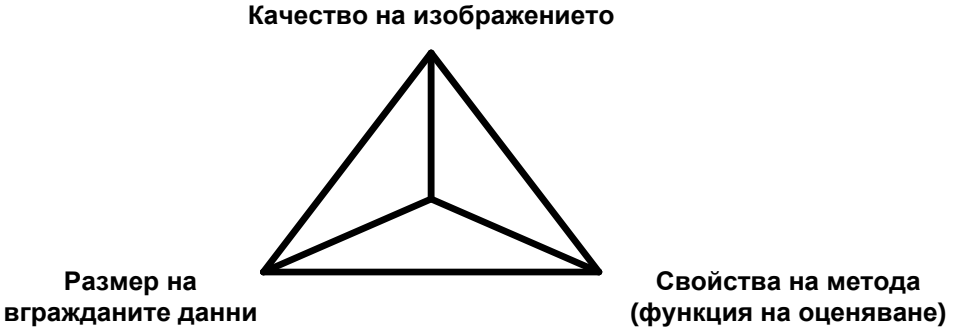
1. Създадена е многослойна архитектура на прототипна програмна система за целите на проверката, оценката и стеганализа на модулния подход за криене на данни и разработените модулни методи.
2. Реализирана е прототипна програмна система посредством програмните езици VB.NET, C# и C/C++. Тя се състои от следните слоеве: слой на данните, слой на базовите модули, слой на приложно-специфичните модули и интерфейсен слой.
3. Проектирани и реализирани са инструменти за групова обработка и анализ на изображенията, за прилагане на филтри и за построяване на хистограми в различни цветови пространства.

Архитектурата на програмната система и програмната реализация са представени в публикуваната в „Доклади на БАН“ авторска публикация [110], издадената от IEEE авторска публикация [111] и публикациите [114] и [115].

Глава 5

Оценка и стеганализ на разработените методи

В тази глава се прави оценка на модулните методи за криене на данни в мултимедия и се разглежда тяхната устойчивост срещу статистически стеганализ. Оценката на методите се извършва по следните три критерия (Фиг. 57):



Фиг. 57: Критерии за оценка на модулните методи

1. *Качество на изображението* – показва степента на подобие на изображенията преди и след процеса на кодиране на данните, измерена чрез средната квадратична грешка (MSE) и отношението на пиковия сигнал към шума (PSNR) [127]. По-малка степен на подобие съответства на по-ниско качество.
2. *Размер на вграждане данни* – дава представа за максималната големина на данните, които могат да се вградят в изображението. Този и предишният критерий често се измерват като средни величини, получени чрез обработка на множество различни файлове с изображения и данни.
3. *Свойства на метода (функция на оценяване)* – своеобразен вид оценъчна функция, която представя степента на значимост на свойствата на метода за дадено приложение. Тя взема под внимание модулността и адаптивността на методите, устойчивостта срещу JPEG трансформации, способността за вграждане на произволни, дефинирани от потребителя двоични данни и възможността за откриване на неправомерно променени области в изображението (вж. 1.2.3 и 2.1).

Даден метод не може да се оптимизира по трите критерия едновременно. Например, ако трябва да се вгради по-голямо количество данни, качеството на изображението ще се влоши и/или някое от свойствата на метода трябва да се пожертва. Добавянето

на допълнително свойство (като например откриването на неправомерно променени области в изображението) към метода за криене на данни води до намаляване на максималния размер на вгражданите данни и/или понижаване на качеството на изображението. Когато методите за криене на данни се оценяват, тези компромиси трябва да се вземат под внимание.

Също така различните сценарии на приложение имат различни изисквания. Това означава, че не съществува универсален метод за криене на данни, подходящ за всички възможни сценарии. За всяка област на приложение трябва да се направи подходящият компромис между гореописаните три критерия (например чрез съставяне на подходящата комбинация от базови и приложно-специфични модули) с цел да се максимизира ползата от метода за криене на данни в конкретното приложение. За някои приложения може да се наложи неизбежно извършване на промени в методите, напр. чрез създаване и използване на нови модули.

5.1 Експериментални множества от носещи изображения и двоични данни за вграждане

Оценката на модулните методи се осъществява чрез тяхното прилагане върху специално подбрани множества от изображения и файлове с двоични данни за вграждане. Те са избрани така, че да покриват широк спектър от възможни комбинации между различни видове изображения (цветни, черно-бели, снимки, комикси, и т.н.) и двоични данни (текстови, изпълними, компресирани файлове и др.). Експерименталното множество от изображения е едно и също при всички извършвани експерименти и е описано в раздел 5.1.1. Поради различното максимално количество на вгражданите данни при стеганографския приложно-специфичен модул и приложно-специфичния модул за цифрово маркиране се използват две различни експериментални множества от двоични данни. Те са описани в детайли в раздел 5.1.2.

5.1.1 Експериментално множество от изображения

Специално подбраното множество от *75* носещи изображения се използва при всички извършвани експерименти. То се състои от често използвани стандартни изображения (*Lena, Baboon,* и др.), снимки, направени с цифров фотоапарат, сканирани лого-изображения, комикси, сканирани снимки и сканирани текстови страници. Множеството съдържа цветни изображения, изображения с нива на сиво и черно-бели изображения в следните формати: BMP, JPEG, PNG, GIF и TIFF. Отделните изображения са подбрани така, че да представят богат набор от различни цветови преходи и текстури.

Примерни изображения от експерименталното множество, са представени на Фиг. 58. Модулните методи за криене на данни трябва да могат да обработят коректно всяко едно от тях. Богатото разнообразие на изображенията е индикатор за приложимостта

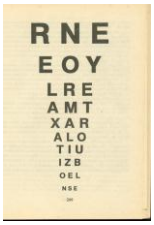
на методите върху често използвани видове изображения в различни области на приложение.

Лена

Цветна снимка

Сканирана снимка в нива на сивото

Цветна рисунка



Сканиран текст

Фиг. 58: Примерни изображения, принадлежащи на експерименталното множество

5.1.2 Експериментални множества от двоични данни

Експерименталните множества от двоични данни се различават за различните типове методи за криене на данни поради голямата разлика в максималния размер на данните, които могат да бъдат вградени. Стеганографският метод се отличава с по-голям капацитет за криене на данни и неговото експериментално множество се състои от двадесет различни текстови, изпълними, компресирани, графични и други двоични файлове с размери от няколко байта до няколко килобайта. Експерименталното множество, използвано при метода за цифрово маркиране, се състои от десет текстови водни знака с размери, вариращи от два до седемдесет байта.

Разнообразието на елементите от двете експериментални множества е индикатор за надеждното представяне на методите при работа с произволни двоични данни. В комбинация с експерименталното множество от изображения, описано в предния раздел, експериментите със стеганографския метод включват $75 \times 20 = 1500$ двойки от изображения и файлове с двоични данни, а експериментите с метода за цифрово маркиране включват $75 \times 10 = 750$ такива двойки.

5.2 Оценка на модулните методи

В този раздел се описва как се изпитват свойствата на модулните методи с различни изображения и двоични данни за вграждане и се измерва качеството на получените изображения. Основно свойство, което трябва да се изпита поради естеството на разработените алгоритми, е устойчивостта на вградените данни срещу JPEG компресия, декомпресия и рекомпресия. То се гарантира от базовия модул *DCTHiderEngine*. Качеството на крайните изображения се измерва с помощта на индикаторите MSE и PSNR (вж. раздел 5.2.2). Методът за стеганографски цели и за методът целите на цифровото маркиране се изпитват отделно поради разликата в максималния размер на използваните двоични данни.

Изпитанията се провеждат чрез описаната в Глава 4 програмна система и нейния графичен потребителски интерфейс за групова обработка и анализ, показан на Фиг. 48 и Фиг. 49. Всички възможни комбинации от два елемента – един елемент от експерименталното множество от изображения и един елемент от експерименталното множество от двоични данни, се обработват от избрания модулен метод с подходящо подбрани параметри на процесите на кодиране и декодиране. След приключване на груповата обработка, получените резултати се използват за оценяването на съответния модулен метод. По време на обработките на изображенията се използва библиотеката с отворен код IJG.

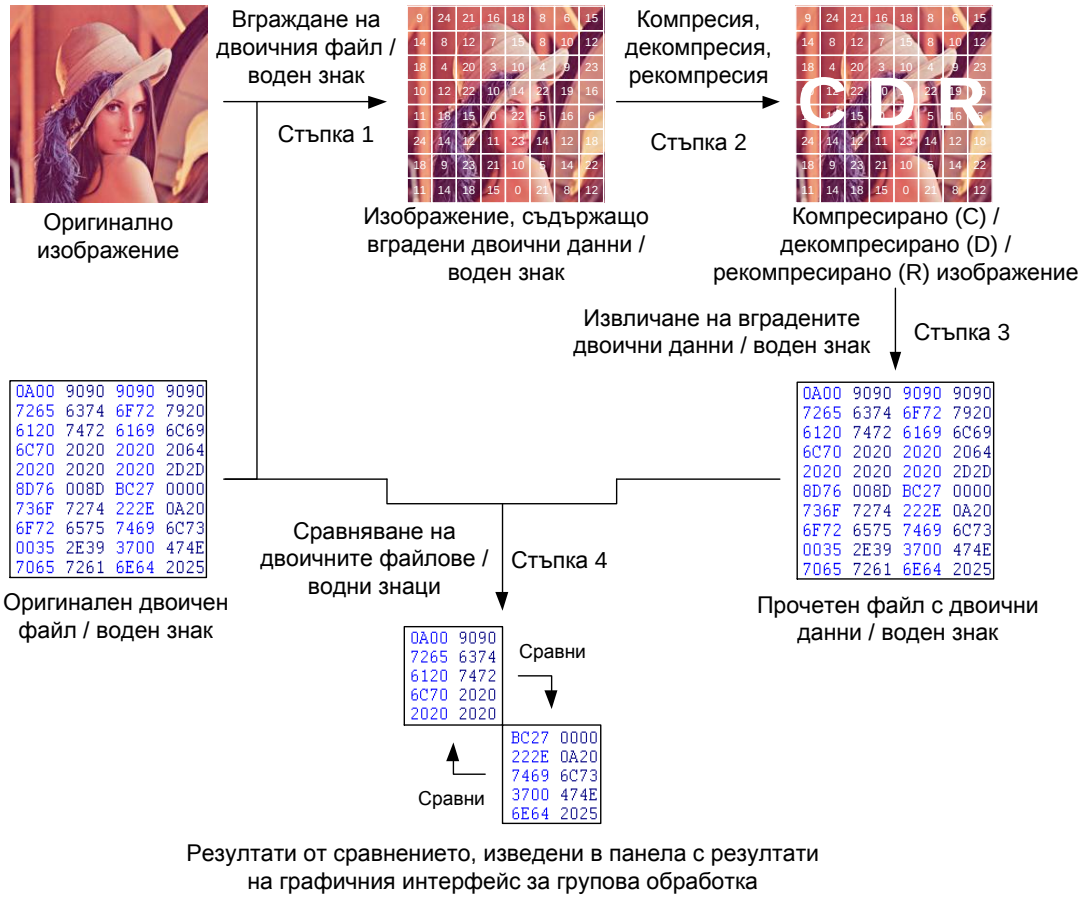
5.2.1 Проверка на устойчивостта срещу JPEG трансформации

Устойчивостта на вгражданите данни срещу JPEG трансформации е основно свойство на модулните методи, предоставяно от базовия модул *DCTHiderEngine*. За да покажем необходимостта от този модул, първо ще покажем съвсем накратко, че данните, вградени в JPEG изображения с фактор на компресия 70, не са устойчиви срещу JPEG декомпресия, без да е приложен методът, описан в раздел 3.2.4.

Нека разгледаме коефициентите $C_{k,l}|(k,l) \in Z_{data}$ и $C_{k,l}^{(1)}|(k,l) \in Z_{data}$, както са дефинирани съответно в стъпки 0 и 2 от алгоритмичното описание в раздел 3.2.4. След обработката на двойките данни от експерименталните множества, средно 99.75% от всички $C_{k,l}^{(1)}$ са идентични с $C_{k,l}$. Допускайки, че са в сила равномерни статистически разпределения на всички величини, то този процент съответства на средно 99.875% некоректно възстановени битове на вградените данни. Тук е важно да се отбележи, че промяната на един пиксел от изображението след декомпресия, може да промени всички DCT коефициенти на съответния блок. Ако това се случи, то тогава битовете данни, вградени в блока, се променят на случаен принцип. Вследствие на това няколко бита на вградените данни могат да се променят в тясна пространствена близост един до друг. Това прави използването на кодове за корекция на грешки ненадеждно. Установеният процент съответства на 1.25 променени бита на всеки 1000 бита вградени данни. Това не е достатъчно за точното извличане на вградените данни, особено ако се цели тяхната употреба като криптографски сигнатури за проверка на изображението. Методът, описан в раздел 3.2.4 има за цел да реши този проблем.

Неговата функционалност се проверява с помощта на три различни експеримента, състоящи се от поредица от четири стъпки. Стъпки от 1 до 3 се различават леко при всеки експеримент, като основната част от разликите се задават като параметри на груповата обработка през графичния потребителски интерфейс. Стъпка 4 е идентична при всички експерименти и се изпълнява от графичния потребителски интерфейс за статистически анализ. Експериментите са подобни за двата разработени приложно-специфични модула – *StegoHider* и *WatermarkHider* (базов модул на методите и в двата

случая е *DCTHiderEngine*), като единствената разлика се състои в използването на различно експериментално множество от данни. Обобщено представяне на стъпките на трите експеримента е дадено на Фиг. 59.



Фиг. 59: Проверка на устойчивостта срещу JPEG трансформации

Първият експеримент проверява устойчивостта на вградените данни срещу JPEG компресия като изпълнява следните стъпки за всяка експериментална двойка, състояща се от едно изображение и един двоичен файл – елементи на съответните експериментални множества:

1. Вграждане на двоичните данни или воден знак в изображението и записване на полученото изображение в BMP формат.
2. Компресиране на полученото изображение (съдържащо вградени данни) в JPEG формат посредством външно приложение за графична обработка, като се използва фактор на JPEG компресия $r \geq q$, където q е предварително зададеното (в програмно меню *Options* на графичния интерфейс) максимално ниво на JPEG компресия, срещу което базовият модул *DCTHiderEngine* гарантира устойчивост на вградените данни. Неравенството $r \geq q$ означава, че извършеното компресиране с фактор на

компресия r запазва повече детайли от оригиналното изображение (и следователно отговаря на по-ниска степен на компресиране) в сравнение с JPEG компресиране с фактор на компресия q.

3. Извличане на вградените двоични данни или воден знак от компресираното JPEG изображение. Двоичните данни или воден знак се записват като файл на твърдия диск.
4. Сравняване на оригиналния двоичен файл или воден знак с файла, прочетен от изображението и записан на твърдия диск при стъпка 3. Ако двата файла (оригиналният и прочетеният) не са идентични, броят на различните битове се пресмята и извежда в панела с резултатите на графичния интерфейс за групова обработка.

Вторият експеримент проверява устойчивостта на вградените данни срещу JPEG декомпресия, изпълнявайки следните стъпки за всяка експериментална двойка, състояща се от изображение и файл за вграждане, съдържащ двоични данни или воден знак:

1. Вграждане на двоичните данни или воден знак в изображението и записване на полученото изображение в JPEG формат.
2. Декомпресиране на полученото изображение (съдържащо вградени данни) посредством външно приложение за графична обработка и записване на резултата в BMP формат.
3. Извличане на вградените двоични данни или воден знак от декомпресираното BMP изображение. Двоичните данни или воден знак се записват като файл на твърдия диск.
4. Сравняване на оригиналния двоичен файл или воден знак с файла, прочетен от изображението и записан на твърдия диск при стъпка 3. Ако двата файла (оригиналният и прочетеният) не са идентични, броят на различните битове се пресмята и извежда в панела с резултатите на графичния интерфейс за групова обработка.

Третият експеримент проверява устойчивостта на вградените данни срещу JPEG рекомпресия, изпълнявайки следните стъпки за всяка експериментална двойка, състояща се от изображение и файл за вграждане, съдържащ двоични данни или воден знак:

1. Вграждане на двоичните данни или воден знак в изображението и записване на полученото изображение в JPEG формат.
2. Рекомпресия на полученото изображение (съдържащо вградени данни) в JPEG формат посредством външно приложение за графична обработка. Използва се фактор на JPEG компресия $r \geq q$, където q е предварително зададеното (в програмно меню *Options* на графичния интерфейс) максимално ниво на JPEG компресия, срещу което базовият модул *DCTHiderEngine* гарантира устойчивост на вгражданите данни.

3. Извличане на вградените двоични данни или воден знак от компресираното JPEG изображение. Двоичните данни или воден знак се записват като файл на твърдия диск.
4. Сравняване на оригиналния двоичен файл или воден знак с файла, прочетен от изображението и записан на твърдия диск при стъпка 3. Ако двата файла (оригиналният и прочетеният) не са идентични, броят на различните битове се пресмята и извежда в панела с резултатите на графичния интерфейс за групова обработка.

След завършване на проверката, всеки от гореописаните експерименти извежда в панела с резултатите на графичния потребителски интерфейс броя на прочетените файлове с данни или водни знаци, които са различни от техните оригинали. Този брой се разделя на общия брой обработени двойки и получената дроб, отразяваща процента на неуспешни вграждания, се използва при оценяването на модулните методи. Резултатът от проверката *R* се дефинира в проценти както следва:

$$R = \left(1 - \frac{n_d}{n_a}\right).100\% ,$$

където n_d е броят различни файлове, а n_a е общият брой обработени двойки.

В идеалния случай всички прочетени файлове с данни или водни знаци са идентични със своите оригинали, което води до нулев брой на разликите. Тогава резултатът от проверката е равен на *100%* и експериментът за проверка на устойчивостта срещу JPEG трансформации е преминат успешно. В случай на наличие на разлики, резултатът от проверката е по-малък от *100%*, което показва неуспешно преминат експеримент. В такъв случай, методите за криене на данни трябва да бъдат подобрени, така че да се елиминират причините за получените разлики.

5.2.2 Проверка на качеството на изображенията

Качеството на изображенията се измерва чрез степента на подобие между всяко едно от оригиналните изображения в експерименталното множество от изображения и съответното изображение, получено след вграждане на двоичен файл или воден знак от експерименталното множество с данни. Малки разлики между изображенията са индикатор за добро качество, докато големи разлики свидетелстват за нежелано голяма загуба на качество, която е забележима за крайния потребител.

Оценката на степента на подобие между изображенията може да се направи с помощта на следните статистически индикатора: средна квадратична грешка (MSE) и отношение на пиковия сигнал към шума (PSNR) [127]. И двата индикатора оценяват нивото на многобройните разлики между изображенията с помощта на една единствена числена стойност и са важни критерии при оптимизацията и оценяването на модулните методи за криене на данни.

Средната квадратична грешка MSE се изчислява с помощта на следната формула, която взема предвид числените стойности на RGB компонентите на пикселите в изображението:

$$e^{(MSE)} = \frac{1}{3d_x d_y}\sum_{i=0}^{d_y-1}\sum_{j=0}^{d_x-1}\left(R_{i,j}^{(1)} - R_{i,j}^{(2)}\right)^2 + \left(G_{i,j}^{(1)} - G_{i,j}^{(2)}\right)^2 + \left(B_{i,j}^{(1)} - B_{i,j}^{(2)}\right)^2.$$

В горната формула с $e^{(MSE)}$ се обозначава средната квадратична грешка, с d_x и d_y се обозначават съответно ширината и височината на изображението, а с $R_{i,j}^{(1)}$, $G_{i,j}^{(1)}$, $B_{i,j}^{(1)}$, $R_{i,j}^{(2)}$, $G_{i,j}^{(2)}$ и $B_{i,j}^{(2)}$ се обозначават съответно червената, зелената и синята компоненти на пиксела с координати *(i, j)* съответно от първото и второто изображение.

Изчисляването на средната квадратична грешка е стандартен статистически подход за обективно измерване на степента на различие между две комплексни величини (функции, сигнали, изображения и др.). Малка стойност на MSE означава, че средното ниво на разлика между двете изображения е малко. В специалния случай на две идентични изображения, MSE има стойност, равна на нула.

Отношението на пиковия сигнал към шума PSNR се базира на средната квадратична грешка и се изчислява за изображения по следната формула:

$$e^{(PSNR)} = 10.log_{10}\left(\frac{P_{max}^2}{e^{(MSE)}}\right) = 10.log_{10}\left(\frac{255^2}{e^{(MSE)}}\right) = 20.log_{10}\left(\frac{255}{\sqrt{e^{(MSE)}}}\right)$$

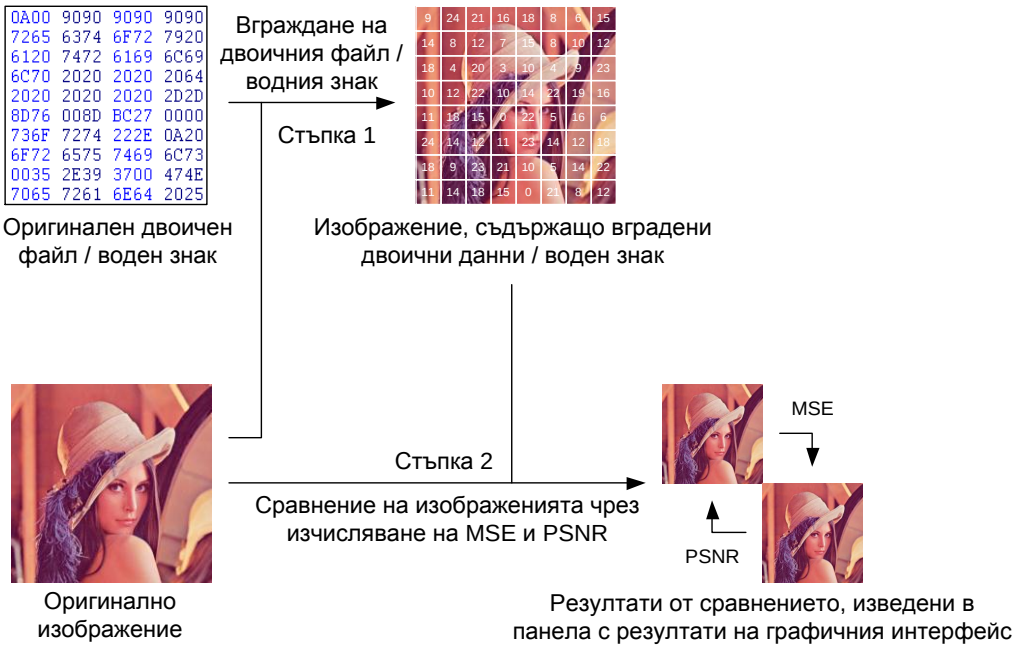
В горната формула с $e^{(PSNR)}$ се обозначава отношението на пиковия сигнал към шума, а с P_{max} се обозначава най-голямата възможна стойност на интензитета на една цветова компонента в пиксел от изображението. P_{max} има стойност *255* в обичайния случай на осембитово представяне на всяка цветова компонента на изображението.

Величината PSNR се измерва в децибели (dB) по логаритмична скала и оценява приблизително човешкото възприятие на средната квадратична грешка. За разлика от MSE, по-голяма стойност на PSNR означава по-добро качество на изображението. В специалния случай на две идентични изображения, PSNR има безкрайно голяма стойност (∞).

Основна цел на всички методи за криене на данни е минимизирането на стойността на MSE и съответно максимизирането на стойността на PSNR. Минималната достижима средна квадратична грешка е равна на нула, максималната възможна стойност на отношението на пиковия сигнал към шума е безкрайност. Тези крайни стойности обикновено са недостижими и съответстват на двойка напълно идентични изображения. В общия случай процесът на кодиране на данните внася промени в оригиналното изображение, което прави получаването на такава идентична двойка изображения изключително рядко явление.

Процедурата за проверка на качеството на получените изображения се извършва чрез графичния потребителски интерфейс за групова обработка и анализ на програмната система. Следващите стъпки се изпълняват за всяка експериментална двойка, състояща се от изображение и файл с двоични данни или воден знак (Фиг. 60):

1. Вграждане на двоичните данни или воден знак в изображението и записване на полученото изображение под форма на графичен файл.
2. Сравнение на оригиналното и полученото след процеса на вграждане на данни изображение чрез изчисляването на статистическите индикатори MSE и PSNR, които измерват разликата между двете изображения. Изчислените стойности се извеждат в панела с резултатите на графичния интерфейс за групова обработка.



Фиг. 60: Проверка на качеството на изображенията

След като всички експериментални двойки бъдат обработени, се изчисляват средни стойности за MSE и PSNR, даващи обща характеристика за работата на метода върху избраните експериментални множества от изображения и двоични данни или водни знаци:

$$e_{avg}^{(MSE)} = \frac{1}{K}.\sum_{i=1}^{K} e_i^{(MSE)} \quad \text{и} \quad e_{avg}^{(PSNR)} = \frac{1}{K}.\sum_{i=1}^{K} e_i^{(PSNR)} .$$

В горната формула K обозначава броя на обработените двойки (изображение-вграждан двоичен файл или воден знак), $e_i^{(MSE)}$ и $e_i^{(PSNR)}$ са изчислените стойности на двата статистически индикатора за експерименталната двойка с пореден номер i. Средните стойности $e_{avg}^{(MSE)}$ и $e_{avg}^{(PSNR)}$ характеризират средното качество на изображението, постигнато от модулните методи. Тези стойности се използват при оценката на всеки един метод и са важни обективни критерии за установяване на общата забележимост на промените, направени в оригиналното изображение по време на процеса на кодиране на данни.

5.2.3 Експериментални резултати на модулния стеганографски метод

Модулният стеганографски метод за криене на данни използва в тази своя експериментална конфигурация базовия модул *DCTHiderEngine*. Експериментите се извършват по процедурите, описани в раздели 5.2.1 и 5.2.2. Оценява се устойчивостта срещу различни видове JPEG трансформации, както и постигнатото средно качество на изображенията, съдържащи вградени данни. Резултатите от експериментите са обобщени в Табл. 3.

Факторът на JPEG компресия е параметър, дефиниран от крайния потребител или приложение (вж. раздел 4.6.1), който контролира максималната степен на компресия, срещу която вградените данни са устойчиви и остават кодирани в изображението. Промяната на този параметър води до отместване на баланса между свойството устойчивост срещу JPEG трансформации и средното качество на получените изображения (Табл. 3).

Експерименталните резултати от проверката на устойчивостта срещу JPEG компресия, декомпресия и рекомпресия – в рамките на степента, позволена от фактора на JPEG компресия – са отлични: *100%* устойчивост. Следователно, проверката на това свойство на модулния стеганографски метод е премината успешно. Получените резултати потвърждават коректната реализация на свойството като част от базовия модул *DCTHiderEngine*.

Табл. 3: Модулен стеганографски метод – постигнати резултати

Параметри на експерименталните процедури				Постигнати резултати				
Експериментални двойки	Фактор на JPEG компр.	Среден размер на изобр. [пиксели]	Среден размер на данните [байтове]	Устойчивост срещу JPEG трансформации			Средна стойност на MSE	Средна стойност на PSNR [dB]
				Компресия [%]	Декомпресия [%]	Рекомпресия [%]		
1500	*70*	*606x583*	*1004*	*100*	*100*	*100*	*26,5*	*38,9*
1500	*80*	*606x583*	*1004*	*100*	*100*	*100*	*15,5*	*40,0*
1500	*90*	*606x583*	*1004*	*100*	*100*	*100*	*7,5*	*42,2*

Средните стойности на средната квадратична грешка се намират в интервала *[7,5; 26,5]* (цветови нива / нива на сиво), а средните стойности на отношението на пиковия сигнал към шума са в интервала *[38,9; 42,2]* (децибели). Тези стойности показват добро качество на изображенията, тъй като средната абсолютна разлика между оригиналното изображение и изображението след вграждане на данните е относително малка, вземайки предвид характеристиките на човешкото око. Визуалната инспекция на резултатите потвърждава заключението от числовите данни. Средният размер на вгражданите данни (*1004* байта) е добра стойност, като се има предвид средния размер на изображенията – *606x583* пиксела – и отличната устойчивост срещу JPEG трансформа-

ции. Както може да се очаква, с намаляването на зададения от потребителя фактор на JPEG компресия, намалява и средното качество на изображенията след вграждане на двоичните данни. Това се вижда от намаляващите стойности на средната квадратична грешка. Причината за това е, че при по-ниска стойност на фактора на JPEG компресия вградените данни са устойчиви срещу по-високи нива на компресия. Това подобряване на устойчивостта се компенсира с постепенно влошаване на качеството на изображението. Независимо от значителното понижение на MSE, съответните средни стойности на PSNR са близки една до друга. Това показва, че възприеманата от човешкото око разлика в качеството на изображенията е малка.

5.2.4 Експериментални резултати на модулния метод за цифрово маркиране

Модулният метод за цифрово маркиране се изпитва в комбинация с базовия модул *DCTHiderEngine*. Аналогично на стеганографския метод, проверката на метода се извършва, следвайки процедурите, описани в раздели 5.2.1 и 5.2.2. Те оценяват устойчивостта на метода срещу JPEG трансформации и средното качество на изображенията, получени след вграждането на водния знак. Постигнатите резултати са показани в Табл. 4.

Проверката на устойчивостта срещу JPEG трансформации показва отличен резултат – устойчивост на вградените водни знаци в *100%* от експерименталните двойки. Този резултат още веднъж потвърждава коректността на новия алгоритъм, използван в *DCTHiderEngine*, както и неговата програмна реализация (Табл. 3 и Табл. 4).

Табл. 4: Модулен метод за цифрово маркиране – постигнати резултати

Параметри на експерименталните процедури				Постигнати резултати				
Експериментални двойки	Фактор на JPEG компр.	Среден размер на изобр. [пиксели]	Среден размер на данните [байтове]	Устойчивост срещу JPEG трансформации			Средна стойност на MSE	Средна стойност на PSNR [dB]
				Компресия [%]	Декомпресия [%]	Рекомпресия [%]		
750	*70*	*606x583*	*39*	*100*	*100*	*100*	*26,3*	*38,3*
750	*80*	*606x583*	*39*	*100*	*100*	*100*	*15,0*	*39,9*
750	*90*	*606x583*	*39*	*100*	*100*	*100*	*7,0*	*42,4*

Средните стойности на средната квадратична грешка се намират в интервала *[7,0; 26,3]* (цветови нива / нива на сиво), а средните стойности на отношението на пиковия сигнал към шума принадлежат на интервала *[38,3; 42,4]* (децибели). Те са много подобни на съответните стойности за стеганографския метод, което показва добра предсказуемост на поведението на базовия модул *DCTHiderEngine*. Получените стойности показват добро качество на изображението, тъй като разликата между изображенията

преди и след вграждането на водните знаци е сравнително малка. В сравнение с проверката на стеганографския метод, тук е използвано различно експериментално множество от водни знаци, чиято средната големина е значително по-малка: *39* байта. Това се налага поради различните свойства, предлагани от метода. Експерименталното множество от изображения е запазено идентично.

Аналогично на стеганографския метод, качеството на изображенията се увеличава с увеличаването на фактора на JPEG компресия (по-голяма стойност на този фактор съответства на по-ниско ниво на компресия). По този начин крайният потребител или приложение може да постигне подходящ компромис между устойчивостта срещу JPEG трансформации и качеството на полученото изображение.

5.2.5 Оценка на модулните методи

Този раздел представя оценката на модулните методи, направена на базата на резултатите от процедурите за проверка, дискутирани в предходните раздели. Оценката е направена според триадата от критерии: качество на изображението, размер на вгражданите данни и свойства на метода/функция на оценяване (Фиг. 57).

Качеството на изображението се измерва посредством стойностите на статистическите индикатори MSE и PSNR. Те се пресмятат за три различни предварително дефинирани фактора на JPEG компресия. Както е показано в раздели 5.2.3 и 5.2.4, стойностите на двата индикатора за двата изследвани метода (стеганографски метод и метод за цифрово маркиране) са приблизително еднакви. Стойностите на MSE варират в сравнително широкия интервал: *[7,0; 26,5]* за различните фактори на JPEG компресия. Стойностите на PSNR, които представят по-точно човешкото възприятие на разликите между изображенията, лежат в по-тесния интервал: *[38,9; 42,4]*. Тези стойности на PSNR са сравними с влошаването на качеството на изображението, предизвикано от самия процес на JPEG компресия. Типичните стойности на PSNR, сравнявайки компресирано в JPEG изображение без вградени данни с некомпресирания му оригнал, варират между *20* и *40 dB* в зависимост от фактора на JPEG компресия [127], [128]. В сравнение с качеството на изображенията, получени от други методи, продукти и Интернет-базирани услуги за криене на данни (вж. раздел 1.2, както и Табл. 5 и Табл. 6), качеството на изображенията, постигнато от модулните методи е отлично. Средните стойности на PSNR лежат в тесни граници около *40 dB*, което надминава много от съществуващите методи при използваните средни размери на изображението и вгражданите данни.

Размерът на вгражданите данни се измерва в байтове. Поради различните изискванията, налагани от областта на приложение на всеки стеганографски метод и метод за цифрово маркиране е много трудно да се извърши сравнение на количеството данни, които могат да бъдат вградени от всеки метод. Средните размери на данните, вграждани от модулните методи – *1004* байта за модулния стеганографски метод и *39* байта за

метода за цифрово маркиране – показват голямата степен на вариране на този критерий за оценка. Това е породено от желанието да се добавят допълнителни свойства на метода, което неизбежно води (в различна степен) до намаляване на максималния размер на вгражданите данни и/или влошаване на качеството на изображението. Както се вижда от експерименталните резултати в Табл. 3 и Табл. 4, качеството на изображението е приблизително еднакво за модулния стеганографски метод и модулния метод за цифрово маркиране. Това неизбежно води до рязко намаляване на максималния размер на данните, вграждани от метода за цифрово маркиране.

Табл. 5: Характеристики на съществуващите методи за криене на данни

Метод	Размер на вгражданите данни за изображения с размери *256x256* до *768x512* пиксела [байтове]	Средна стойност на PSNR [dB]
JSteg [50], [52]	*2224 - 2642*	*29,71 - 41,60*
Джао, Кох [54] , [55]	*380*	*Н.Р.*
О'Руанаид [56]	*55 - 512*	*Н.Р.*
Кокс [57]	*125*	*Н.Р.*
Ву, Лиу [58]	*135*	*Н.Р.*
Лин, Чанг [59], [60], [61]	*Н.Р.*	*32,95 – 40,70*
Провос [62], [63]	*1462*	*Н.Р.*
Вестфелд [64]	*192 - 1935*	*Н.Р.*
Чанг [69]	*6656*	*27,63 – 39,14*
Фридрих [73], [74], [75]	*130 - 1124*	*35,32 – 53,12*
Ли, Кокс [81]	*1536*	*28,00 – 49,00*
Изадиния [84]	*8192*	*43,11 – 43,12*

* Н.Р. = Не се разглежда (няма данни)

Сравнението на количеството вграждани данни с други методи, продукти и услуги за криене на данни (Табл. 5 и Табл. 6) не е тривиална задача. Всеки метод е разработен в отговор на специфични потребителски нужди и съществува голяма степен на вариране. Освен това за много методи количеството на вгражданите данни зависи не само от размерите на изображението, но и от неговото съдържание (стойностите на пикселите) и от различни параметри, зададени от потребителя на метода. Средното количество данни, които даден метод може да вгради, обикновено варира от няколко байта при методите за цифрово маркиране и стига до килобайтове при стеганографските методи. Като се имат предвид тези особености, размерът на данните, които могат да бъдат вградени от модулните методи, се движи в диапазона, характерен за класическите методи, продукти и услуги.

Третият критерий за оценка – *свойства на метода (функция на оценяване)* – трябва да представи върху числова скала комбинацията от свойства, предлагани от метода. Целта е да се позволи сравнение на метода с други избрани методи по този критерий. Възможен прост подход е да се изброят свойствата на метода, важни за дадено приложение, и техният брой да се нанесе върху числовата скала. Основен недостатък на този

подход е фактът, че свойствата на методи, създадени от различни изследователи и фирми, се различават в своята дефиниция и програмна реализация, което затруднява допълнително сравнението.

Табл. 6: Характеристики на съществуващите продукти и услуги за криене на данни

Продукт / Услуга	Размер на вграждaните данни за изображения с размери *256x256* до *768x512* пиксела [байтове]	Средна стойност на PSNR [dB]
Steganos Privacy Suite [79]	*Н.Р.*	*52,70*
JPHide [80]	*Н.Р.*	*56,40*
InvisibleSecrets [81]	*неограничен*	*Н.Р.*
Digimarc [90]	*3 - 4*	*35,10*
Photopatrol [92]	*Н.Р.*	*33,00*
SignMyImage [94]	*10*	*35,80*
Icemark [96]	*20*	*32,60*
Eikonamark [97]	*8*	*40,80*

* Н.Р. = Не се разглежда (няма данни)

В специалния случай на модулните методи, сравнението е по-лесно, тъй като методите не само имат един и същи създател, но и споделят базовия модул *DCTHiderEngine*, който осигурява следните важни свойства (вж. и раздел 2.1):

1. Модулност и адаптивност;
2. Пълна устойчивост срещу различни видове JPEG трансформации;
3. Възможност за работа с произволни изображения и вграждани данни.

Също така двата модулни метода споделят следните свойства:

1. Незадължителни кодове за корекция на грешки;
2. Псевдослучайно разпределение на данните в изображението.

Стеганографският приложно-специфичен модул не добавя нови свойства към цялостния стеганографски метод и се стреми да максимизира количеството вграждани данни. От друга страна, *приложно-специфичният модул за цифрово маркиране* добавя следните свойства към цялостния метод за цифрово маркиране, който се получава като комбинация от базовия и приложно-специфичния модул:

1. Откриване на области от изображението, променени (неправомерно) след вграждането на водния знак;
2. Възстановяване на вградения воден знак в случай на наличие на такива променени области.

Следователно, модулният метод за цифрово маркиране включва всички свойства, предоставени от *DCTHiderEngine* и характерни за стеганографския метод, като добавя две нови свойства за сметка на понижаването на максималното количество данни, които могат да се вградят в изображението. Ако се оценява броя на свойствата на всеки метод: *5* свойства, предоставени от стеганографския метод и *7* свойства, предоставени от метода за цифрово маркиране, то превъзходството на метода за цифрово маркиране по този критерий е оценено коректно.

Когато е наложително сравнение с други съществуващи методи за криене на данни, е необходимо прилагането на по-общ подход. Основен проблем е преобразуването на характеристиките и свойствата на методите в добре обосновани числени стойности, които да могат да се използват за сравнение. Това може да се постигне чрез специална функция на оценяване, която приема степента на реализация на всяко свойство на метода като аргумент и връща като резултат числена стойност, описваща ползата, която комбинацията от реализирани свойства носи на крайния клиент. Примерна функция на оценяване от този тип може да се формира като сума от произведението на теглата, асоциирани с всяко свойство на метода и зависещи от конкретния сценарий на приложение, със степента на реализация на съответното свойство:

$$f_{eval} = \sum_i w_i r_i \,,$$

където f_{eval} е функцията на оценяване, w_i е теглото, асоциирано със свойство i, а r_i е степента на реализация на това свойство.

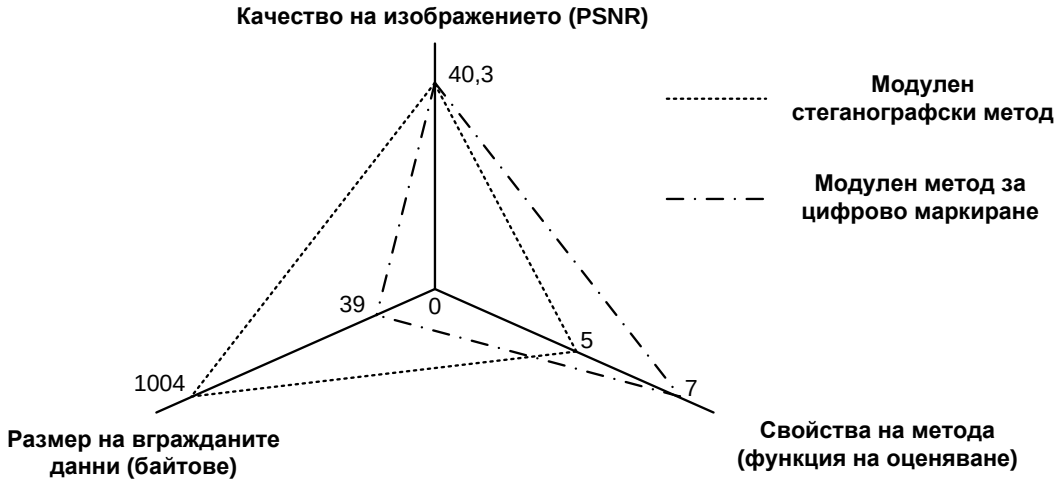
Нека разгледаме конкретен приложен сценарий, за който са важни *устойчивостта срещу JPEG трансформации* (свойство 1) и *възстановяването на данните в случай на промени в изображението* (свойство 2), като свойство 1 е особено важно за крайния клиент, а свойство 2 е желателно, но не задължително. След внимателни разговори с крайния клиент се определят тегла w_1 и w_2, такива че $w_1 > w_2$, напр. $w_1 = 1$ и $w_2 = 0.5$, които изразяват по подходящ начин относителната важност на всяко свойство за клиента. Нека сравним модулния метод за цифрово маркиране и напр. метода *JSteg* [50], [52]. За модулния метод за цифрово маркиране (означен чрез *Mdw*) имаме $r_1^{(Mdw)} = r_2^{(Mdw)} = 1$, което съответства на пълна степен на реализация на двете свойства. Методът *JSteg* вгражда данни, устойчиви само на JPEG компресия, като не разглежда JPEG декомпресия и рекомпресия. Тази различна реализация на свойство 1 затруднява оценката. След разговор с клиента, уточняващ важността на различните JPEG трансформации, се оценява $r_1^{(JSteg)} = 0.5$. Методът *JSteg* не реализира свойство 2, водейки до $r_2^{(JSteg)} = 0$. Крайните оценки за всеки метод са както следва:

$$f_{eval}^{(Mdw)} = w_1 r_1^{(Mdw)} + w_2 r_2^{(Mdw)} = 1.5 \text{ и } f_{eval}^{(JSteg)} = w_1 r_1^{(JSteg)} + w_2 r_2^{(JSteg)} = 0.5.$$

Тези стойности показват, че модулният метод за цифрово маркиране е значително по-подходящ за използване в конкретния приложен сценарий в сравнение с *JSteg*.

Както е показано в мотивацията за разработката и дефинирането на важните характеристики на модулните методи (вж. 1.1 и 2.1), класическите методи за криене на данни не притежават най-подходящата комбинация от свойства, необходими за приложение в Интернет-базирани сценарии. В частност, класическите методи за криене на данни не са лесно адаптируеми към конкретен сценарий на приложение, който се различава дори леко от сценария, за който са разработени. Освен това те обикновено не гарантират устойчивост срещу всички видове JPEG трансформации. Следователно, функцията на оценяване на техните свойства, не може да даде резултат, близък до този на модулните методи.

При промяна на контекста на употреба на методите се сменят и изискванията на потребителите. Тези промени се отразяват и на функцията на оценяване на свойствата, засягайки пряко стойностите на теглата w_i, което води до различни резултати от оценката.



Фиг. 61: Модулни методи за криене на данни – резултати от оценката

Резултатите от оценката на двата модулни метода са обобщени на Фиг. 61. Тя показва средната стойност на PSNR за двата метода като индикатор на качеството на изображението, както и размера на вградените данни (вж. Табл. 5 и Табл. 6). Функцията на оценяване е опростена до брой на свойствата, предоставяни от всеки метод. Подчертан е компромисът, който се прави между размера на вградените данни и двете нови свойства, предоставени от метода за цифрово маркиране. Компромиси от този вид променят площта и положението на триъгълника, който описва графично общата оценка на метода. Това води до промяна в практическите приложения, за които е подходящ съответният метод.

Чрез анализ на изискванията на потребителя, могат да се определят минимални стойности за всеки критерий на оценка – минимална стойност на PSNR, минимален размер на вградените данни и свойства на метода, необходими за съответния Интернет-базиран сценарий. По този начин за даден сценарий може да се определи мини-

мална площ на триъгълната област, както и ограничения в нейното положение, които трябва да бъдат изпълнени за всеки метод за криене на данни, за който се твърди, че удовлетворява потребителските изисквания.

Представената диаграма на Фиг. 61 е удобен начин за визуализация на:

- Характеристиките на методите за криене на данни;
- Изискванията към методите за криене на данни, наложени от даден сценарий на приложение.

Тя може да се използва като опростен и интуитивен графичен интерфейс, с помощта на който крайните потребители могат да сравнят визуално силните и слабите страни на различни методи и да изберат този метод, който е най-подходящ за нуждите им. При необходимост може да се създаде нов метод, предлагащ желания баланс чрез:

- Създаване на нов базов или приложно-специфичен модул;
- Промяна на вече съществуващ модул;
- Използване на различна комбинация от вече създадени модули.

След това новият метод може да се изпита, за да се оцени дали отговаря по-добре на потребителските изисквания за конкретното приложение. Ако оценката е положителна, новият метод (дефиниран чрез неговия базов и приложно-специфичен модул) може да се съхрани с оглед на бъдеща употреба в сходен сценарий на приложение.

Важно е да се отбележи, че диаграмата на Фиг. 61 може да се разшири чрез включване на допълнителни критерии за оценка на методите като например скоростта на изпълнение или необходимите изчислителни ресурси (изчислително време, памет, характеристики на процесора и др.). Те са от особено значение при реализация на методите за микроконтролери или малки преносими устройства (англ. embedded systems). По този начин се позволява кратко и гъвкаво визуално описание на методите за криене на данни и бърза оценка на тяхната приложимост в даден Интернет-базиран сценарий. След това модулният метод, предлагащ най-подходящата комбинация от критериите за оценка, може да бъде избран и внедрен.

5.3 Оценка на ефективността на съществуващите подходи и методи за статистически стеганализ

Едно от важните изисквания към практическата приложимост на методите за криене на данни е способността за вграждане на данните по такъв начин, че те да са невидими не само за реалните потребители, но и за съществуващите методи за стеганализ. Като част от провежданата научноизследователска дейност, експертите по криене на данни в мултимедия са разработили статистически методи със сравнително широк об-

сег на приложение, способни да откриват присъствие на вградени данни в JPEG изображения. Важно е да се отбележи, че тези методи дават статистически обоснован отговор в проценти на въпроса има ли вградени данни в дадено изображение (пасивен стеганализ). Ако има такива, може да се търси по-подробна информация за техните характеристики като дължина на данните, използван метод за криене и параметри на криенето (активен стеганализ). Основно ограничение на методите за стеганализ е, че като правило те не могат да декодират съдържанието на откритите данни. Информация за статистическия стеганализ и за някои от по-разпространените методи за стеганализ и за ефективността на някои стандартни стеганализ експерименти, проверяващи изображенията за наличие на вградени данни, е дадена в [6], [9], [36], [62], [63], [66], [67] и [129].

В този раздел се разглежда целесъобразността на формалното изследване и проверка на системите за стеганография и цифрово маркиране, обобщават се някои от разработените до момента по-популярни методи за стеганализ и след това се разглеждат резултатите от прилагането им върху изображения, съдържащи данни, вградени от модулните методи. Трябва предварително да се отбележи, че се счита за нерешима задача (поне за сега) създаването на универсален метод за стеганализ, способен да открива наличието на данни, вградени от произволен метод. Обикновено даден метод за стеганализ разпознава резултата от действието на един единствен метод за криене на данни или в най-добрия случай от действието на група методи, базирани на общ подход.

При разработката на методи за стеганализ обикновено се анализира в детайли предварително определения метод (или група от методи) за криене на данни, който трябва да бъде атакуван. Основната цел на този анализ е намирането на статистически (или други характерни) артефакти в изображението, които да служат като индикатори за извършена обработка от съответния метод. Това предполага детайлното познаване на метода за криене на данни. Ако липсва информация за принципа на действие и програмната реализация на метода за криене на данни, то задачата за създаване на ефективен стеганализ метод се усложнява многократно.

Методите за стеганализ могат да бъдат разделени на същите категории като методите за криене на данни (вж. раздел 2.2). Тук фокусът е основно върху методите за стеганализ, работещи в DCT-спектралната област. Причината за това се крие в основния базов модул, разработен в дисертацията, който осигурява устойчивост срещу JPEG трансформации. За втория базов модул, разработен за некомпресирани изображения или изображения, компресирани с беззагубна компресия, са от значение методи за стеганализ, разработени с цел откриване на данни, вградени в най-младшите битове на пикселите. От гледна точка на еволюцията и надграждането на методите за стеганализ, първо ще се дискутират някои от методите, работещи директно с пикселите на изображението и техните най-младши битове. След това ще се представят стеганализ методите, работещи в DCT-спектралната област.

След като се разгледа ефективността на съществуващите методи за стеганализ по отношение на изображения, генерирани от модулните методи, в следващите раздели се

разработват и изследват от автора няколко нови прототипни метода за стеганализ. Те имат за цел да проверят устойчивостта на модулните методи срещу няколко принципни подхода за стеганализ.

5.3.1 Целесъобразност на формалното изследване и проверка на системите за стеганография и цифрово маркиране

Преди да се разгледат в детайли някои от разпространените методи за стеганализ, ще бъде представен накратко общ информационно-теоретичен стеганографски модел, предложен от Кашин [130]. Той е разпространен представител на опитите за формално изследване и проверка на системите за стеганография и цифрово маркиране. Други информационно-теоретични модели са дискутирани от Цьолнер [131], Мителхолцер [132] и Съливан [133].

Информационно-теоретичният модел на Кашин използва понятията ентропия, взаимна информация (англ. mutual information) и относителна ентропия [134], [135] при описанието и анализа на стеганографските системи.

Ентропията на статистическото разпределение P_X върху дискретно множество Ψ, се дефинира по следния начин:

$$H(X) = -\sum_{x \in \Psi} P_X(x) log P_X(x),$$

където с X обикновено се обозначава случайна променлива със статистическо разпределение P_X. Условната ентропия (англ. conditional entropy) на случайна променлива X при дадена случайна променлива Y се дефинира като:

$$H(X|Y) = -\sum_{y \in \Omega} P_Y(y) H(X|Y = y),$$

където с $H(X|Y = y)$ се обозначава ентропията на условното статистическо разпределение $P_{X|Y=y}$. Ентропията на дадено статистическо разпределение винаги удовлетворява неравенствата: $0 \leq H(X) \leq log|\Psi|$.

Взаимната информация между X и Y се дефинира като намаляването на ентропията в резултат на допълнителното присъствие на Y: $I(X;Y) = H(X) - H(X|Y)$, като се наблюдава симетрия между X и Y: $I(X;Y) = I(Y;X)$.

Относителната ентропия между две статистически разпределения P_{X_1} и P_{X_2} се дефинира по следния начин:

$$D\left(P_{X_1} \parallel P_{X_2}\right) = -\sum_{x \in \Psi} P_{X_1}(x) log \frac{P_{X_1}(x)}{P_{X_2}(x)},$$

като по дефиниция: $0 log \frac{0}{0} = 0$ и $p log \frac{p}{0} = \infty$ за $p > 0$.

Относителната ентропия между две статистически разпределения е винаги неотрицателна. Тя е равна на нула тогава и само тогава, когато статистическите разпределения са равни. Трябва да се отбележи, че относителната ентропия не е истинска мярка за

близост в математическия смисъл, тъй като не е симетрична относно разпределенията и не изпълнява необходимото неравенство.

Нека с C се обозначават носители на информация, несъдържащи вградени данни, които се генерират спрямо статистическото разпределение P_C. Аналогично, нека с S се обозначават носители на информация, съдържащи вградени данни, които се генерират спрямо статистическото разпределение P_S. Вградените данни се обозначават с E. Те са генерирани спрямо статистическото разпределение P_E, и се вграждат в носителите чрез функция (алгоритъм) на вграждане F. Алгоритъмът за възстановяване на тези данни се обозначава с G, като двойката алгоритми (F, G) образуват т.нар. стегосистема.

Въвеждат се следните дефиниции, валидни за стегосистеми:

1. Една стегосистема се нарича *ε-сигурна* (англ. *ε-secure*), ако е валидно неравенството:

$$D(P_C \parallel P_S) \leq \varepsilon.$$

   Ако ε е равно на нула, то тогава стегосистемата се нарича *перфектно сигурна* (англ. *perfectly secure*). Тази дефиниция е валидна за стеганографски системи за предаване на едно единствено съобщение.
2. Ако се моделира стеганографска система за многократно предаване на съобщения, източниците/носителите на информация могат да се моделират като стохастични процеси. Ако броят на предаваните съобщения се обозначи с n и носителите на информация, несъдържащи вградени данни, се генерират от стационарен информационен източник, то горната дефиниция може да се промени както следва:

   Една стегосистема се нарича *средно ε-сигурна* (англ. *ε-secure on average*), ако е валидно неравенството:

$$\lim_{n\to\infty} \frac{1}{n} D(P_C \parallel P_S) \leq \varepsilon.$$

   Ако ε е равно на нула, то тогава стегосистемата се нарича *перфектно средно сигурна* (англ. *perfectly secure on average*).

Въз основа на тези дефиниции се правят вероятностни оценки на максималната ефективност на стеганализ-методите, разработени за дадена стегосистема. Основното ограничение на този модел следва от допускането, че всички носители на информация, както и вгражданите данни, се подчиняват на съществуващи и известни статистически разпределения, до които потребителите на стегосистемата имат достъп.

За разлика от други инженерни и информационни области, приложението на вероятностни модели в стеганографията и цифровото маркиране е силно затруднено поради специфичния начин на използване и обработка на вгражданите данни и техните носители. В стандартния случай не може да се създаде вероятностен модел на използваните носители на информация (напр. изображения) или вграждани данни (напр. двоични файлове или водни знаци), тъй като по дефиниция те са изцяло под контрола на отдел-

ните потребители, чиито нужди са различни и се променят във времето. Това е особено силно изразено при използване на методите за криене на информация в Интернет-базирани сценарии, където всеки потребител се нуждае от напасване на метода за вграждане на данни към конкретните нужди в даден момент. Поради тази причина, настоящите информационно-теоретични модели, описващи стеганографски системи, нямат това практическо значение, което би могло да се очаква, ако се направи директен паралел с класическата криптография.

5.3.2 Методи за стеганализ, работещи директно с пикселите на изображението

Този раздел представя някои от по-значимите методи за стеганализ, които работят директно с пикселите на изображението. Тези методи служат като необходимо въведение, улесняващо изложението на методите за стеганализ, работещи в спектралната област. Един от класическите и може би най-известни и добре изследвани подходи за стеганализ [6], [136], [137], [138], [139] визира откриването на информация, вградена посредством директно заменяне на най-младшия бит на всеки цветови канал на изображението с пореден бит от данните, които се вграждат (англ. *LSB flipping*). Методът за стеганализ разчита на следната статистическа зависимост, която се поражда като следствие от вграждането на данните:

$$h(2k) = \sum_{i,j} c_{i,j}^{2k} \approx \sum_{i,j} c_{i,j}^{2k+1} = h(2k+1),$$

където с h се обозначава хистограмата на изображението, а с i, j се обозначават координатите на даден пиксел в изображението. Стойността на $c_{i,j}^{n}$ е 1, ако интензитетът на пиксела $c_{i,j}$ (или неговото ниво на сиво) в дадено цветово пространство има стойност равна на n. Ако стойността на интензитета е различна от n, то стойността е: $c_{i,j}^{n} = 0$.

При криенето на произволно избрана или криптирана информация, след заменянето на най-младшия бит, вероятността пикселът да има четна стойност на интензитета ($2k$) е равна на вероятността пикселът да има нечетна стойност на интензитета ($2k+1$). Вследствие от това хистограмата h на изображението се състои от групирани по двойки съседни стълбове. Стълбовете от всяка двойка имат приблизително еднакви стойности. Този ефект е толкова по-силно изразен, колкото е по-голямо количеството на скритата информация.

Описаната по горе статистическа особеност на хистограмата може да се използва за откриването на наличие на данни, вградени чрез директно заменяне на най-младшия бит, с голяма степен на вероятност. Тази базова идея за стеганализ и подходът за изследване на хистограмата на изображенията (хистограмен анализ) с цел откриване на наличие на вградени данни са доразвити в редица следващи подходи и методи за стеганализ.

Общ подход, както и резултати от изследването на статистическите особености на хистограмата, включващ анализ на двойки стълбове на хистограмата и т.нар. Regular-Singular (RS) анализ, са представени от Кер [140]. Идеята за RS-анализа е предложена от изследователската група на Фридрих през 2001г. [141], [142], [143] и математически формулирана и изследвана в [144], [145] и [146]. Този подход включва разделянето на пикселите на изображението на групи с малък брой (напр. 4) пиксели и образуването на „дискриминираща" функция, чиято стойност нараства с нарастването на нивото на шум в пикселите на групата. Примерната оценъчна функция може да се свърже със степента на различие на стойностите (дисперсията) на пикселите в групата:

$$f(x_1, x_2, \dots, x_n) = \sum_{i=1}^{n-1} |x_{i+1} - x_i|$$

В допълнение се дефинират две пермутации F_1 и F_{-1}:

$$F_1: 0 \leftrightarrow 1, 2 \leftrightarrow 3, \dots, 254 \leftrightarrow 255$$

$$F_{-1}: -1 \leftrightarrow 0, 1 \leftrightarrow 2, \dots, 255 \leftrightarrow 256$$

С помощта на горните пермутации за дадена група пиксели $G = (x_1, x_2, \dots, x_n)$, която се състои от пикселите $x_1, x_2, \dots, x_n$ със стойности, принадлежащи на множеството $P = \{0, 1, \dots, 255\}$ (съответстващо на 8-битов цветови канал), могат да се дефинират следните видове групи:

Регулярни групи (*R*-групи): $G \in R \Leftrightarrow f\big(F(G)\big) > f(G)$

Сингулярни групи (*S*-групи): $G \in S \Leftrightarrow f\big(F(G)\big) < f(G)$

Като се има предвид изложеното по-горе, основната идея на RS-анализа се състои в изследването на броя на групите R_M, R_{-M}, S_M и S_{-M}, като за едно „нормално" изображение без скрита информация, очакваният брой на всяка една от групите е свързан със следните зависимости:

$$R_M \approx R_{-M} \text{ и } S_M \approx S_{-M}$$

За разлика от горните зависимости в изображения със скрита информация в най-младшите битове на пикселите се наблюдава следната зависимост:

$$R_M \approx S_M,$$

като стойностите на R_{-M} и S_{-M} се отличават значително от $R_M \approx S_M$.

Авторите твърдят и представят експериментални резултати в подкрепа на тезата, че техният подход е ефективен срещу методите за криене на данни, реализирани в различни продукти (комерсиални и с отворен код като *Steganos*, *S-Tools*, *Hide4PGP*), които използват директно заменяне на най-младшия бит за вграждане на данни в цветни изображения и изображения с нива на сивото.

Друг метод, базиран на т. нар. *диференчна хистограма на изображението* (англ. *difference image histogram*), е предложен от Джанг и Пинг [147], [148], [149]. Първата стъпка от метода е генерирането на т.нар. *диференчно изображение D* (англ. *difference image*) посредством следната дефиниция:

$$D_h(i, j) = I(i + 1, j) - I(i, j),$$

където D_h е диференчното изображение, а $I(i,j)$ е интензитетът на пиксела с координати (i, j).

След това се създават диференчните изображения D_f и D_g на изображенията, получени съответно чрез обръщане (англ. flip) на LSB-битовете на оригиналното изображение или замяната им с нулев бит. Авторите твърдят, че съществуват статистически зависимости между трите диференчни изображения, ако в оригиналното изображение няма вградени данни. При наличие на псевдослучайни вградени данни, тези статистически зависимости се променят. По този начин може да се открие използването на LSB-базирани методи за криене на информация и да се установи приблизително размера на вградените данни.

Разгледаните по-горе подходи за анализ на изображения с цел откриването на наличие на данни, вградени с помощта на LSB-базирани методи, не могат да бъдат използвани директно за откриването на данни, вградени посредством DCT-базирани методи или други методи, които вграждат данни в спектралната област (напр. DWT или Фурие базирани методи). От друга страна, те биха могли да се използват като отправна точка за създаването на нови методи за стеганализ на DCT-коефициентите на изображението. В този случай трябва да се вземат предвид различаващите се статистически особености на DCT-коефициентите, които са от важно значение за успешното функциониране и добри резултати, постигнати от стеганализа.

5.3.3 Методи за стеганализ, работещи в DCT спектралната област

Този раздел представя някои от значимите методи за стеганализ, които анализират DCT спектралната област, за да установят наличие на данни, устойчиви на загубна компресия. Тези данни са вградени най-често от методи, които целят постигането на устойчивост срещу JPEG компресия.

Въпреки различните области (пространствена и спектрална), в които се вгражда информация, може да се направи паралел между подходите за вграждане на данни на двете групи методи - методите, работещи директно с пикселите на изображението, и методите, работещи с DCT коефициентите. От една страна, извършваните от всички тези методи (в пикселите или в DCT коефициентите) промени имат пряко или непряко отражение върху младшите битове на съответните стойности, които могат да се анализират статистически. От друга страна, ефективността на директното пренасяне на даден подход за статистически стеганализ от пространствената област в спектралната област се намалява значително вследствие на различните статистически характеристики на групите от съседни пиксели в изображението и на групите от DCT коефициенти, принадлежащи на един DCT блок. Вследствие на това, директното приложение на подходите за LSB-стеганализ е неефективно при изображения, съдържащи вградени в спектралната област данни. Въпреки това, тези подходи могат да послужат като полез-

на отправна точка при разработването на нови методи за стеганализ, които отчитат статистическите особеностите на DCT коефициентите в спектралната област.

Един от важните общи подходи за изследване на изображения, в чиито DCT коефициенти е възможно да има вградени данни, е предложен от Фридрих и нейната изследователска група [150], [151], [152]. Подходът се базира на възможността за получаване на добро приближение на оригиналното изображение (изображението преди вграждане на данните) от изображението, което се анализира за наличие на вградени данни. Така полученото приближение на оригиналното изображение има „макроскопични" характеристики, които са подобни на тези на оригинала. Практическата реализация на този подход се състои в декомпресиране на анализираното JPEG изображение до матрица от пиксели и последващо намаляване на ширината и дължината му с *4* пиксела чрез премахване на първите *4* реда и първите *4* колони. Новополученото изображение има пиксели, зададени по следния начин:

$$I_{new}(i,j) = I_{old}(i+4, j+4),$$

където с $I_{new}(i,j)$ и $I_{old}(i,j)$ се обозначава интензитета на пиксела с координати *(i, j)* съответно в новото и първоначалното изображение.

Новото изображение се рекомпресира, като се използват параметрите на компресия (степен на компресия и квантизиращи таблици) на първоначалното изображение. По този начин се намалява значително статистическият и визуалният ефект от разделянето на пикселите в блокове с големина 8×8, характерни за JPEG компресията.

Следващата стъпка е определянето на статистическа характеристика *S*, специфична за даден метод за криене на данни. Тази характеристика обикновено зависи от дължината на вгражданите данни и може да се моделира като монотонно растяща функция *S(m),* където *m* представлява дължината на вгражданите данни. По дефиниция $S(0) = S_{original} \approx S_{new}$, а $S(m) = S_{old}$, където $S_{original}$ е стойността на статистическата характеристика за оригиналното изображение (т.нар. „baseline" стойност). По този начин с помощта на оценката на *S(0),* получена чрез предложения подход, може да се направи вероятностна оценка за наличие на вградени данни и тяхната предполагаема дължина.

Авторите на подхода дискутират различни статистически характеристики в [150] и [151]. Част от тях са статистически характеристики от първи ред. Те са базирани върху различни изследвания на хистограмата на DCT коефициентите с цел откриване на данни, вградени от алгоритъма *F5* на Вестфелд [64]. Други са статистически характеристики от втори ред. Те изследват корелациите между DCT блоковете и разликата в ентропията и степента на увеличението на нивото на шум в DCT коефициентите на изображението при повторно вграждане на псевдослучайни двоични данни с цел откриване на данни, вградени от алгоритъма *OutGuess* на Провос [53], [63]. Предлагат се и статистически стеганализ-класификатори за тези два алгоритъма. При разработването и прилагането на статистическата характеристика е важно да се вземат предвид и статистическите артефакти, предизвикани от двойна JPEG компресия (вграждане на данни в

изображение, чийто изходен файлов формат е JPEG), за да се постигне успех при практически експерименти.

На базата на гореописания подход някои автори предлагат и изследват други статистически характеристики. Сред тях са характеристики, базирани на дискретна уейвлитна трансформация (англ. *DWT*) [148], корелация на цветовете [153], или т.нар. Марков-базирани (англ. Markov based) характеристики, които използват разликите между DCT коефициентите в хоризонтална, вертикална и диагонална посока [152]. Изследват се различни алгоритми за криене на данни: *Steghide* [154], *F5* [64], *OutGuess* [53], [63], Хуанг [155], Пива [156], Съливан [133] и др.

Друг разпространен подход за анализ на изображенията е дискутиран от Хармсен и Пърлман [157]. Авторите предлагат представянето на входното изображение в *пространство на признаците* (англ. feature space). В идеалния случай то линеаризира зависимостите между дадена (статистическа) характеристика на изображението и наличието и размера на вградените данни. След това в пространството на признаците може да се приложи стандартен статистически анализ, като се предлага използването на т.нар. *линейна дискриминанта на Фишер* (англ. *Fisher Linear Discriminant*, *FLD*). Тя използва предварително класифицирани и обозначени обучаващи данни, разделени в два класа - такива, които съдържат вградени данни, и такива, които не съдържат вградени данни. Целта е да се намери едномерно подпространство, където проекциите на представителите на обучаващите данни върху подпространството да изпълняват следните условия:

1. Максимизиране на разстоянието между средните стойности за всеки клас;
2. Минимизиране на вариацията в рамките на всеки клас.

Авторите предлагат и нелинейно разширение на тази дискриминанта в пространството на признаците. Като статистическа характеристика на изображението се използва сравнително бърза за изчисление хистограма h_C на квантизираните DCT коефициенти (вж. раздел 3.2.2), дефинирана по следния начин:

$$h_C[d] = \sum_{k \in C} h_k[d],$$

където $h_k[d]$ е броят на квантизираните DCT коефициенти, които имат стойност d и честотен индекс k в зигзагообразната JPEG матрица ($k \in K = \{0, 1, \ldots, 63\}$), а $C \subseteq K$, за DCT трансформацията, използвана в JPEG, която има размер на блоковете 8×8.

С горната сума се цели групирането на отделните хистограми h_k в n на брой групи, така че:

$$\sum_{d \in S} h_{C_i}[d] \approx \sum_{d \in S} h_{C_j}[d] \text{, за } \forall i, j \in \{1, \ldots n\} \text{ и } \forall d,$$

където C_1, ..., C_n са подмножества на K, такива че $C_i \cap C_j = \emptyset$ за $\forall i, j \in \{1, \ldots n\}$ и $\bigcup_{i \in \{1, \ldots, n\}} C_i = K$ и $S = \{-p, \ldots, p\}$ е множеството на стойностите, които могат да бъдат

заемани от DCT коефициентите. С помощта на горното групиране може да се дефинира примерен характеристичен вектор за дадено изображение като:

$$\vec{x} = [h_{C_1}[-p], \dots, h_{C_1}[p], \dots, h_{C_n}[-p], \dots, h_{C_n}[p]]^T.$$

Авторите твърдят, че този вектор дава добри резултати при относително голям размер на вгражданите данни (напр. 75% точност на разпознаване при размер на данните, равен на 50% от максималния). Основната трудност при приложението на този подход е в намирането на достатъчно ефективна и бърза за изчисление статистическа характеристика на изображението, по която могат да се отличат изображенията, съдържащи вградени данни, от изображенията, които не съдържат вградени данни.

Друг подход, който има някои общи черти с описаните методи, е предложен от Ли [158]. Предлаганият метод използва линейната дискриминанта на Фишер в комбинация с друг характеристичен вектор, получен от DCT матрицата. Ши [159] използва като характеристична функция разликите в 2D JPEG матрици, които са моделирани като Марковски процес. След това се прилагат гранични нива (англ. threshold), които намаляват размерността на характеристичния вектор с цел намаляване на времето, необходимо за извършването на класификацията. Твърди се, че методът е ефективен при откриването на наличие на данни, вградени от *OutGuess* [53], *F5* [64] и *MB1* [68]. Подобрения на този подход са предложени от Чен и Ши [160], Лиу [161], [162] и Чо [163], [164].

Необходимо е да се отбележи, че част от подходите, разработени с цел откриване на наличие на вградени данни в DCT спектралната област, се базират на друг вид трансформация - дискретната уейвлитна трансформация (DWT), като се използват различни статистически характеристики на уейвлитните коефициенти [165], [166], [167]. Дискретната уейвлитна трансформация (DWT) би могла да има предимство при стеганализа с това, че тя комбинира информация както за самите пиксели на изображението (пространствена информация), така и за неговия честотен спектър. Неин основен недостатък е повишаването на сложността на разработване на стеганализ методите поради използването на друг вид коефициенти, различен от DCT-коефициентите, в които непосредствено е вградена информацията.

5.3.4 Ефективност на методите за стеганализ по отношение на модулните методи за криене на данни

Този раздел дискутира ефективността на някои по-популярни и разпространени алгоритми за стеганализ при откриването на данни, вградени от модулните методи. За тази цел е използвана програмата с отворен код *Stegdetect* [168], създадена от Нилс Провос. *Stegdetect* може да провежда четири различни вида статистически стеганализ, които откриват присъствие на данни, вграждани от класически програми за криене на данни като *JSteg, JPHide* и *OutGuess*. Чувствителността на статистическите експерименти, която влияе на допуснатите грешки при откриването на данни (вж. [6]), се контролира чрез специален команден параметър на програмата, наречен *Sensitivity*.

За да се оцени ефективността на стеганализ алгоритмите по отношение на откриването на данни, вградени от модулните методи, *Stegdetect* се стартира върху три множества, съставени от JPEG изображения.

Първото множество се състои от оригиналните експериментални изображения, които не съдържат никакви вградени данни, като те се преобразуват предварително в JPEG формат (вж. раздел 5.1).

Второто и третото множество са съставени от изображения, съдържащи данни, вградени съответно от стеганографския метод и метода за цифрово маркиране. Изображенията, принадлежащи на тези множества, са получени като резултат от процедурите за проверка, описани в раздели 5.2.1 и 5.2.2, използвайки фактора на компресия, избран по подразбиране: 70. Експерименталните данни, вградени във всяко от двете множества изображения, са различни.

За всяко изображение от дадено множество *Stegdetect* оценява вероятността то да съдържа вградени данни и в зависимост от прага на чувствителност дава отговор „*да*" (има вградени данни) или „*не*" (няма вградени данни). Оценката се базира на четири различни експеримента за стеганализ. След като всички изображения, от дадено множество, бъдат изследвани от *Stegdetect* за наличие на вградени данни, процентът P_c на изображенията, в които *Stegdetect* не е открил данни, се пресмята по следния начин:

$$P_c = \frac{n_c}{n_a} \times 100\% ,$$

където n_c е броят на изображенията, в които *Stegdetect* не е открил данни, а n_a е общият брой на изображенията в дадено множество.

Табл. 7: Резултати от анализа, проведен със *Stegdetect*

Stegdetect чувствителност	P_c за множеството от оригинални изображения [%]	P_c за множеството от изображения, съдържащи данни, вградени от стеганографския метод [%]	P_c за множеството от изображения, съдържащи данни, вградени от метода за цифрово маркиране [%]
1,0	91,5	97,7	93,0
2,0	73,2	91,7	88,0

* Стойността по подразбиране на чувствителността на *Stegdetect* е равна на 1,0

Табл. 7 показва стойностите на P_c за трите множества от изображения при два различни прага на чувствителност в *Stegdetect* (стойността по подразбиране на прага на чувствителност е равна на *1.0*).

Високите процентни стойности на P_c водят до извода, че съществуващите статистически методи за откриване на вградени данни не могат да откриват надеждно присъствието на данни, вградени от модулните методи. Нещо повече – процентните стойности на P_c за двете множества от изображения, съдържащи вградени данни, са дори по-високи от стойностите за множеството от оригинални изображения,. Това показва, че модулните методи се справят добре със задачата да наподобят естествени, необработвани изображения. В това отношение стеганографският метод се справя малко по-

добре от метода за цифрово маркиране, тъй като неговите P_c стойности са малко по-големи.

Най-вероятните причини за този на пръв поглед леко изненадващ резултат се крие във възможни статистически деформации, съществуващи в оригиналните изображения и предизвикани напр. от многократна компресия. Някои от тези деформации могат да бъдат погрешно интерпретирани от стеганализ алгоритмите, като индикатори за наличие на вградени данни. Модулните методи коригират голяма част от тези деформации и по този начин постигат по-голяма процентна стойност на P_c.

В заключение, резултатите, представени в този раздел, показват неефективността на някои от съществуващите и разпространени статистически алгоритми и подходи за стеганализ, когато се използват за откриване на данни, вградени от модулните методи. Това е важно предимство, свързано със сигурността, пред редица традиционни методи за криене на данни, придобили широка популярност, като *JSteg, JPHide, OutGuess* и други, базирани на тях.

5.4 Реализация и оценка на ефективността на два принципни подхода за стеганализ

Практическите резултати, представени в раздел 5.3.4, показват, че съществуващите методи за стеганализ не са ефективни при откриването на данни, вградени от модулните методи, разработени в дисертацията. Като се имат предвид тези резултати и обзора на някои от важните подходи и методи за стеганализ, представен в раздели 5.3.2 и 5.3.3, в този раздел се разглеждат няколко различни варианта на два принципни подхода при създаването на общи методи за стеганализ:

1. Първият принципен подход се базира на изследването на корелации в изображението. Използва се пространствената връзка между пикселите на изображението, която се изразява в това, че съседни пиксели често имат подобни стойности на интензитета. Поради тази причина се очаква корелацията в рамките на един блок, както и между блоковете, да е по-малка в изображения, съдържащи скрита информация, в сравнение с техните оригинали.
2. Вторият подход се базира на прилагане на филтър върху изображението и изследването на разликите между резултата от филтрирането и изображението, което се филтрира, чрез построяването на диференчно изображение. Тъй като вграждането на данни би следвало да увеличи нивата на шум в изображението, то се очаква диференчните изображения, получени от изображенията, съдържащи скрита информация, да се отличават с по-високи стойности на интензитета на пикселите в сравнение с диференчните изображения, получени от оригиналите.

Основната цел е да се оцени практическата приложимост на двата подхода по отношение на модулните методи. Както при разработката на методите за криене на информация, така и при стеганализа, се цели създаването на методи, които не изискват достъп до оригиналното изображение или статистическа информация, която го описва (слепи методи, англ. blind methods).

Трябва да се отбележи, че модулният подход за криене на данни утежнява разработката на методи за стеганализ поради факта, че всеки модулен метод е съставен от комбинация от модули, всеки от които има влияние върху статистическите свойства на мултимедийното съдържание, като базовите модули имат по-голямо влияние в сравнение с приложно-специфичните модули. Тъй като потребителят има свободата сам да определя използваната комбинация от модули в зависимост от нуждите си, е необходима разработката на множество методи за статистически стеганализ, всеки от които е способен да открива наличието на информация, вградена от комбинация от модули. Част от насоките за бъдеща работа е създаването на модулна концепция за стеганализ, която е огледално копие на модулния подход, разработен в дисертацията и която би позволила разделянето на методите за стеганализ на модули.

В следващите раздели се описва програмната реализация на няколко разновидности на двата гореописани подхода и се дискутират получените резултати.

5.4.1 Изследване на корелации в изображението

Реализирани са общо четири алгоритъма за изследване на корелации и други статистически характеристики в изображенията. Алгоритмите се базират на разделянето на изображението на блокове от пиксели. Нека допуснем, че изображението се състои от m блока по височина и n блока по ширина. Нека всеки блок $Z_{(c,d)}, c \in \{0,1,\dots,m-1\}, d \in \{0,1,\dots,n-1\}$ е с височина k елемента и ширина l елемента. Нека означим с $Z_{(c,d)}(a,b), a \in \{0,1,\dots,k-1\}, b \in \{0,1,\dots,l-1\}$ елемента с координати *(a,b)* в рамките на блок *(c,d)*. Елементите на блоковете могат да бъдат DCT коефициенти (JPEG) или пиксели, в зависимост от предполагаемия използван алгоритъм за вграждане на данни. Представените в този раздел резултати се отнасят единствено към елементи на блоковете, които са DCT коефициенти. Самите блокове отговарят на JPEG стандарта ([47]) и са с размер 8×8 елемента. Алгоритмите разглеждат обикновено t на брой коефициента на DCT трансформацията, като всеки коефициент z_e има координати в DCT блока $(a_e, b_e), a_e \in \{0,1,\dots,7\}, b_e \in \{0,1,\dots,7\}, \text{e} \in \{0,1,\dots,t-1\}$.

Първият алгоритъм, наречен *Interblock1*, функционира по следния начин: за всеки коефициент z_e се изпълняват поредица от стъпки. За всеки ред от блокове $Z_{(c_i,d)}, c_i \in [0;m)$ се образуват две поредици от блокове $Z_{(c_i,d_f)}$ и $Z_{(c_i,d_g)}$, всяка от които се състои от $\lfloor n/2 \rfloor$ блока, където $\lfloor n \rfloor$ обозначава премахването без закръгляване на дробната част на n, $d_f \in \{2s | s \in \left[0; \left\lfloor \frac{n}{2} \right\rfloor\right)$ и $d_g \in \{2s+1 | s \in \left[0; \left\lfloor \frac{n}{2} \right\rfloor\right)$. От двете поредици блокове се

образуват съответните поредици от коефициенти $Z_{(c_i,d_f)}(a_e,b_e)$ и $Z_{(c_i,d_g)}(a_e,b_e)$, за които се изчислява корелацията на Пиърсън (англ. Pearson). За две случайни променливи $X \equiv Z_{(c_i,d_f)}(a_e,b_e)$ и $Y \equiv Z_{(c_i,d_g)}(a_e,b_e)$ корелацията на Пиърсън се изчислява по следната формула [169], [170], [171]:

$$\rho_{X,Y} = \frac{cov(X,Y)}{\sigma_x \sigma_y} = \frac{E[(X - E(X))(Y - E(Y))]}{\sigma_x \sigma_y},$$

като в дискретния случай формулата приема вида:

$$r_{X,Y} = \frac{\sum_{i=0}^{n-1}(X_i - E(X))(Y_i - E(Y))}{\sqrt{\sum_{i=0}^{n-1}(X_i - E(X))^2}\sqrt{\sum_{i=0}^{n-1}(Y_i - E(Y))^2}},$$

където $E(X)$ и $E(Y)$ са съответно средните стойности (математическите очаквания) на случайните променливи *X* и *Y*.

Коефициентите на корелация R_{c_i,z_e} се пресмятат за всички редове $c_i \in [0;m)$ за даден коефициент z_e на DCT трансформацията. Ако се разглеждат *t* на брой коефициенти на DCT трансформацията, то тогава общият брой на коефициентите на корелация е $m \times t$. За тях се изчислява квадратен корен от квадратичната средна стойност (англ. root mean square (RMS) value) посредством следната формула:

$$RMS_{row} = \sqrt{\frac{1}{tm}\sum_{e=0}^{t-1}\sum_{i=0}^{m-1} R_{c_i,z_e}^2}.$$

Аналогично се изчислява стойността RMS_{col} за колоните от блокове на изображението (n на брой) и коефициентите z_e на DCT трансформацията (t на брой). От двете стойности RMS_{row} и RMS_{col} се изчислява RMS_{img} за изображението:

$$RMS_{img} = \sqrt{\frac{RMS_{row}^2 + RMS_{col}^2}{2}}.$$

Така получената стойност RMS_{img} се използва за оценка на корелацията между блоковете на DCT трансформацията. Поради наличието на пространствена връзка между пикселите на изображението (сходство в интензитета на съседни пиксели) се очаква корелацията да е по-малка при изображенията, съдържащи вградени данни, отколкото при техните оригинали.

Вторият алгоритъм, наречен *Interblock2*, функционира по начин, подобен на *Interblock1*. Разликата между двата алгоритъма се състои в начина на формиране на поредиците от блокове, за които се изчислява корелацията на Пиърсън. По подобие на *Interblock1*, за всеки ред от блокове (без последния ред) $Z_{(c_i,d)}, c_i \in [0;m-1)$ на изображението се изчислява по един коефициент на корелация. Двете поредици от блокове $Z_{(c_i,d)}$ и $Z_{(c_{i+1},d)}$, които участват в пресмятането на коефициента на корелация се образуват съответно от текущия ред от блокове c_i и съседния следващ ред c_{i+1}, като всяка

поредица се състои от n блока. От двете поредици блокове се образуват съответните поредици от коефициенти $Z_{(c_i,d)}(a_e, b_e)$ и $Z_{(c_{i+1},d)}(a_e, b_e)$, за които се пресмята корелация на Пиърсън по подобие на алгоритъма *Interblock1*. След това се изчисляват стойностите RMS_{row}, RMS_{col} и RMS_{img}, като RMS_{img} се използва за оценка на корелацията между блоковете на DCT трансформацията.

Алгоритъмът *Interblock2* пресмята и усреднява квадратично коефициентите на корелация, съответстващи на всяка двойка съседни редове (и колони) от блокове на DCT трансформацията, което води до леки разлики в пресметнатите стойности на корелация в сравнение с алгоритъма *Interblock1*.

Третият алгоритъм, наречен *Intrablock*, също има общи характеристики с *Interblock1* и *Interblock2*. За всеки ред от блокове $Z_{(c_i,d)}, c_i \in [0; m)$ се образуват две поредици от блокове, които участват в пресмятането на коефициента на корелация. Първоначално те са идентични: $Z_{(c_i,d)}$ и $Z_{(c_i,d)}$ и се състоят от n блока. От тях се образуват две различни поредици от стойности $Z_{(c_i,d)}(a_e, b_e)_0$ и $Z_{(c_i,d)}(a_e, b_e)_1$. $Z_{(c_i,d)}(a_e, b_e)_0$ съответства на стойността на най-младшия бит (бит с позиция *0*) в коефициента z_e на съответния блок на DCT трансформацията. Аналогично, $Z_{(c_i,d)}(a_e, b_e)_1$ съответства на стойността на бита с позиция *1* в коефициента z_e. Така получените две поредици от двоични стойности, съответстващи на битовете на даден коефициент z_e за блоковете на DCT трансформацията, които се намират на ред c_i, се подлагат на корелация на Пиърсън по подобие на алгоритъма *Interblock1*. След това се изчисляват стойностите RMS_{row}, RMS_{col} и RMS_{img}, като RMS_{img} се използва за оценка на корелацията между блоковете на DCT трансформацията. Алгоритъмът *Intrablock* пресмята и усреднява квадратично коефициентите на корелация, съответстващи на битове в рамките на едни и същи блокове на DCT трансформацията. Така се създава мярка за вътрешната корелация в рамките на тези блокове.

Четвъртият алгоритъм, наименуван *0-Analysis*, се различава от предходните три алгоритъма по това, че не пресмята директно корелация, а вместо това изчислява процента на нулеви коефициенти $Perc_{zero}$ в блоковете на DCT трансформацията по следния начин:

$$Perc_{zero} = \frac{1}{tmn} \sum_{e=0}^{t-1} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} f_{e,i,j},$$

където $f_{e,i,j} = \begin{cases} 1, & \text{ако } Z_{(c_i,d_j)}(a_e, b_e) = 0 \\ 0, & \text{в противен случай} \end{cases}$.

Процентът на нулеви коефициенти също като корелацията индиректно представлява форма на зависимост между DCT блоковете. Висок процент на нулеви коефициенти съответства на еднородно по цвят изображение без шум, а нисък процент на нулеви коефициенти съответства на изображение с текстури и високо ниво на шум. От това следва, че стойността на $Perc_{zero}$ би могла да се използва за оценка на нивото на шум

и степента на текстуриране на изображението. За изображения, съдържащи вградени данни, би следвало нивото на шум да се промени и вследствие на това процентът на нулеви коефициенти $Perc_{zero}$ да бъде различен.

Описаните четири алгоритъма са програмно реализирани в рамките на програмната система, представена в Глава 4. Фокусът на изследванията в този раздел е насочен към представянето на базовия модул (вж. Глава 3), който е устойчив срещу JPEG трансформации и до голяма степен обуславя статистическите характеристики на JPEG изображенията, съдържащи вградени данни.

Избрана е комбинацията на този модул с приложно-специфичния модул за стеганографски цели, който е разработен за вграждане на възможно най-голямо количество произволни двоични данни. Той позволява постигането на най-голяма степен на промяна на оригиналните изображения и фин контрол върху количеството вградени битове чрез промяна на входните файлове с двоични данни. Следователно, тази комбинация от модули е добра отправна точка за изследванията. Обобщени резултати, получени от всеки един от алгоритмите за стеганализ за 75-те различни експериментални изображения и 20-те различни експериментални файла с данни, дискутирани в раздел 5.1, са дадени съответно в Табл. 1, Табл. 2, Табл. 3 и Табл. 4 от Приложение 1

Първата и втората колона в таблиците съдържат името на файла с изображението и неговия размер. Колоната „Оригинал (о)" съдържа стойността RMS_{img}, пресметната от съответния алгоритъм за оригиналното изображение, което не съдържа вградени данни. Колоните „JPEG 70", „JPEG 80" и „JPEG 90" съдържат статистически параметри, описващи стойностите RMS_{img}, пресметнати от алгоритъма за изображения, съдържащи вградени данни, устойчиви на JPEG компресия с фактор съответно 70, 80 или 90. За всяко едно изображение стойностите RMS_{img}, получени от вграждането на 20-те експериментални файла с данни, се описват групово чрез математическото очакване μ и стандартното отклонение σ. Колоните „(μ-о)/σ" съдържат отклонението (представено в пъти сигма) на стойността, пресметната за оригиналното изображение, от математическото очакване.

След анализ на получените резултати от различните алгоритми, правят впечатление следните важни особености:

1. Големи разлики в стойностите на коефициентите на корелация, изчислени за оригиналните изображения. Това обуславя от своя страна големите разлики в стойностите на математическото очакване μ и затруднява значително еднозначното идентифициране на изображенията, съдържащи вградени данни, ако оригиналното изображение не е достъпно.
2. Сравнително ниските стойности на стандартното отклонение σ, които би следвало да улеснят стеганализа и повишат неговата надеждност при благоприятни стойности на другите статистически характеристики („μ" и „(μ-о)/σ"). От друга страна,

ниски стойности на σ могат да бъдат и индиректна индикация за доброто представяне на методите за криене на информация. Те показват, че JPEG изображенията с различни вградени данни са много близки помежду си и близки до оригиналното изображение (което е в произволен графичен формат), ако и то се подложи на JPEG компресия със същия фактор.

3. Силно променлив знак и сравнително ниски абсолютни стойности на величините в колоните „(μ-o)/σ". Тези величини дават сравнително пряка оценка на надеждността на стеганализа. Оптималният случай, характеризиращ успешен стеганализ, се отличава с постоянен знак на величините и сравнително високи абсолютни стойности (≥ 3), обусловени от правилото „68-95-99,7" при нормално Гаусово разпределение [172]. Това правило гласи, че около 68% от експерименталните стойности попадат в рамките на интервала $\mu \pm \sigma$, около 95% от тях попадат в рамките на интервала $\mu \pm 2\sigma$ и около 99,7% от тях попадат в рамките на интервала $\mu \pm 3\sigma$.

   Случаят, в който това се наблюдава най-ясно изразено, са коефициентите на корелация на изображенията с вградени данни, устойчиви на JPEG компресия с фактор 70, в Табл. 3, като са налице и значителен брой изключения. За другите таблици и фактори на компресия 80 и 90 (по-качествени изображения с по-ниска степен на компресия), силно променливият знак и променливите и сравнително ниски абсолютни стойности (често ненадвишаващи *2*) затрудняват значително стеганализа.

Горепосочените особености на коефициентите на корелация показват, че стеганализът без наличие на оригиналното изображение е ненадежден. В частност, големите разлики в стойностите на коефициентите на корелация, изчислени за оригиналните изображения и силно променливият знак и ниските абсолютни стойности на величините в колоните „(μ-o)/σ" са решаващи фактори, определящи до голяма степен неуспеха на корелационните алгоритми. Подобрение на резултатите би могло да се получи при евентуално установяване на допълнителни зависимости между коефициентите на корелация и други характеристики, напр. размерите и текстурите на изображението, с цел намаляване на разликите в стойностите на коефициентите на корелация. От получените таблици такива зависимости не могат да бъдат непосредствено извлечени.

5.4.2 Прилагане на филтри върху анализираното изображение

За целите на стеганализа е възможно прилагането на различни филтри върху изображението - усредняващ филтър, Гаусов филтър, медианен филтър, Винер-Колмогоров филтър, Лапласиан на Гаусиана, описани подробно в [46]. Независимо от конкретния вид на филтъра, процедурата на прилагане и получаване на числова метрика е една и съща. Нека се използват означенията и основните положения от раздел 5.4.1, като фокусът продължава да е насочен към изследването на JPEG изображения [47]. За всеки коефициент z_e на DCT трансформацията се съставя матрица $A_{orig}^{z_e}$ с размерност $m \times n$

с елементи $Z_{(c,d)}(a_e, b_e)$, $c \in [0; m), d \in [0; n)$. Върху $A_{orig}^{z_e}$ се прилага филтър f_s от някой от гореспоменатите видове и се получава филтрираната матрица $A_{filt}^{z_e} = f_s(A_{orig}^{z_e})$. Пресмята се разликата $A_{diff}^{z_e}$ на матриците $A_{orig}^{z_e}$ и $A_{filt}^{z_e}$: $A_{diff}^{z_e} = A_{orig}^{z_e} - A_{filt}^{z_e}$. Матрицата $A_{diff}^{z_e}$ се използва като индикатор на нивото на шум в изображението и служи като база за пресмятането на числова метрика - квадратен корен от квадратичната средна стойност („root mean square (RMS) value“) посредством формулата:

$$RMS_{A_{diff}^{z_e}} = \sqrt{\frac{1}{mn}\sum_{c=0}^{m-1}\sum_{d=0}^{n-1} A_{diff}^{z_e}(c,d)^2}.$$

Подобно на анализа в раздел 5.4.1, обикновено се разглеждат *t* на брой коефициента z_e на DCT трансформацията, e ∈ $[0; t)$. За всеки коефициент се пресмята метриката $RMS_{A_{diff}^{z_e}}$ и след това от тях се образува крайната метрика RMS_{img}, описваща изображението:

$$RMS_{img} = \sqrt{\frac{1}{t}\sum_{e=0}^{t-1} {RMS_{A_{diff}^{z_e}}}^2}.$$

Тази метрика е единична реална стойност, която в идеалния случай характеризира числено нивото на шум в изображението. Гореописаният алгоритъм за изчисление на RMS_{img} е програмно реализиран за гореспоменатите пет филтъра в рамките на програмната система, описана в Глава 4. Както и в раздел 5.4.1, фокусът на изследванията е върху представянето на базовия модул, устойчив срещу JPEG трансформации, който до голяма степен обуславя статистическите характеристики на изображенията с вградени данни (вж. Глава 3). Избрана е същата комбинация на този модул с приложно-специфичния модул за стеганографски цели и същите 75 различни експериментални изображения и 20 различни експериментални файла с данни, дискутирани в раздел 5.1. Резултатите за всеки един от петте филтъра са представени съответно в Табл. 1, Табл. 2, Табл. 3, Табл. 4 и Табл. 5 от Приложение 2. Значенията на колоните в таблиците са същите като в раздел 5.4.1.

Получените резултати за различните филтри показват сходство с резултатите, получени в раздел 5.4.1:

1. Големи разлики в стойностите на метриките, изчислени за оригиналните изображения, което обуславя големите разлики в стойностите на математическото очакване μ и затруднява идентифицирането на изображенията с вградени данни.
2. Сравнително ниски стойности на стандартното отклонение σ, които в този случай са индикатор, че JPEG изображенията, които съдържат различни вградени данни,

са близки помежду си и близки до оригиналното изображение (което е в произволен графичен формат), ако и то се подложи на JPEG компресия със същия фактор.

3. Силно променлив знак и променливи абсолютни стойности на величините в колоните „(μ-o)/σ", което затруднява стеганализа. По този критерий резултатите от прилагането на филтри върху изображението са по-добри в сравнение с раздел 5.4.1, като подобрението се изразява в относително по-високите абсолютни стойности на величините. За съжаление, поради големите разлики в стойностите на метриките, изчислени за оригиналните изображения, и разликите в стойностите на математическото очакване μ, това подобрение трудно може да се използва пълноценно за ефективен стеганализ.

Горепосочените особености на метриките показват, че стеганализът чрез прилагане на филтри върху изображението е ненадежден, ако няма достъп до оригиналното изображение. Както и в раздел 5.4.1, големите разлики в стойностите на метриките, изчислени за оригиналните изображения и силно променливият знак и абсолютни стойности на величините в колоните „(μ-o)/σ" са решаващи фактори, определящи до голяма степен неуспеха на стеганализа. Подобрение на резултатите би могло да се получи, ако бъдат намерени допълнителни зависимости между метриките и други характеристики, напр. размерите и текстурите на изображението, с цел намаляване на разликите на стойностите. Получените таблици не дават индикации за съществуването на такива зависимости.

5.5 Заключение

Експериментите и оценката на модулните методи за криене на данни предоставят емпирично доказателство за качеството на тяхното проектиране и програмна реализация. Методите успешно реализират свойствата, необходими за приложение в Интернет-базирана среда, и постигат целта и задачите на дисертацията, дефинирани в раздел 1.4.

В сравнение с класическите монолитни методи за криене на данни, модулните методи предлагат отлично качество на изображението и вграждат добро количество данни. Тяхното основно предимство се състои в уникалната комбинация от свойства, които другите методи не предлагат. Възможността за лесно адаптиране към изискванията на потребителите е иновативна и особено важна характеристика. Тя позволява лесна промяна на методите (в сравнение с класическите монолитни методи) и ускореното им внедряване в различни приложни сценарии в Интернет-базирана среда.

Описаните и изследвани в тази глава подходи и методи за стеганализ не са ефективни по отношение на модулните методи за криене на информация. Както е видно от раздел 5.3, методите за стеганализ обикновено се разработват за изследване на строго определен метод (понякога клас от методи) за криене на информация. Това означава,

че разработката на стеганализ методи за анализ на наличието на данни, вградени от модулните методи, налага всъщност създаването на група от методи за стеганализ или модулна концепция за стеганализ, подобна на модулния подход за криене на информация.

Раздели 5.3.4, 5.4.1 и 5.4.2 показват резултатите, получени съответно от някои популярни методи за стеганализ и някои общи подходи. Заключението е, че разгледаните методи и подходи *не могат* успешно да открият наличието на данни, вградени от модулните методи и по-специално от базовия модул, устойчив на JPEG трансформации, върху който са съсредоточени изследванията. Това следва да даде известна увереност на потребителите на модулните методи, че присъствието на техните вградени данни в изображения е с голяма доза вероятност невидимо за неоторизирани трети лица.

В резултат на проведените изследвания и експерименти са постигнати следните приноси:

1. Оценена е реализацията на модулните методи чрез изследване на устойчивостта на вградените данни срещу JPEG трансформации и проверка на качеството на изображенията.
2. Изследвана и оценена е ефективността на някои популярни и разпространени алгоритми за стеганализ по отношение на модулните методи за криене на данни. Освен това е изследвана и оценена ефективността на два принципни подхода за стеганализ: подход, базиран на изследване на корелациите в изображението, и подход, базиран на прилагане на филтри.

Оценката и стеганализа на разработените методи се дискутират в издадената от IEEE авторска публикация [111], отличения с награда авторски доклад [112] и публикациите [113], [114] и [173].

Глава 6

Приложение на методите в Интернет-базирани сценарии

В тази глава се дискутира практическото приложение на разработените методи в няколко конкретни Интернет-базирани сценария:

1. Предотвратяване на фишинг (англ. phishing) при използване на портали за Интернет-банкиране;
2. Защита на мултимедийната интелектуална собственост на информационни агенции;
3. Подобряване на законосъобразното използване на мултимедийно съдържание в Интернет-базирани общества, например социални мрежи и форуми като Facebook и DevianArt.

Описани са допълнителните предимства на модулните методи за криене на данни за компаниите и клиентите им, включени в тези сценарии. Представена е примерна прототипна реализация, съобразена с нуждите на сценариите. Концепцията и програмната реализация на Интернет-базирана услуга за сертифициране на мултимедийно съдържание посредством методи за криене на данни е подробно разгледана и нейната употреба в сценариите е представена в следващите раздели.

6.1 Предотвратяване на фишинг на банкови портали

Този сценарий разглежда как сигурността на порталите за Интернет-банкиране може да се подобри чрез използването на методи за криене на данни в допълнение към традиционните технологии за сигурност, които банките използват. Сценарият е от значение и за други корпорации с Интернет портали, които дават достъп на крайни клиенти до част от корпоративната инфраструктура – мобилни оператори като М-Тел, големи Интернет-базирани търговски платформи като Amazon и eBay или услуги за плащане като PayPal.

6.1.1 Фишинг – описание на проблема

Фишингът е подход, използван от криминални лица за неправомерно придобиване на лични и финансови клиентски данни като например персонални идентификационни номера (англ. personal identification number, PIN), потвърдителни номера на транзакции (англ. transaction authentication number, TAN), номера на банкови сметки, номера и данни на кредитни карти или потребителски имена и пароли.

Обикновено жертвата на фишинг получава подправено електронното съобщение, което привидно е изпратено от известна банка или друга институция, с която жертвата има търговски взаимоотношения. В електронното съобщение се споменава за сериозен проблем с данните, транзакциите или акаунта на жертвата – например превод на голяма сума пари, поръчка на голямо количество стока, или предсрочно закриване на акаунта. Съобщението има за цел да шокира жертвата и да я убеди, че са необходими незабавни мерки за предотвратяване на сериозни материални загуби.

Втората част от електронното съобщение се състои от кратки и ясни насоки за отстраняване на проблема. Жертвата трябва да отвори Интернет портала на банката или институцията, като използва ясно видима електронна връзка (англ. hyperlink), обозначена в съобщението. В действителност, електронна връзка отваря Интернет страница, имитираща реалния портал, която е под контрола на криминалния подател на електронното съобщение. Вече отворила фалшивата Интернет страница, жертвата е подканена да въведе своите лични или финансови данни (PIN, TAN, номер на банкова сметка, номер на кредитна карта и т.н.), за да отстрани въображаемия проблем. Веднага щом жертвата въведе исканите данни, те могат да се използват от подателя на фишинг-съобщението за нелегални цели – обикновено свързани със значителни финансови загуби.

Друг по-изтънчен вариант на фишинг използва свойствата на т.нар. система за имената на областите (англ. Domain Name System, DNS). Преди да може да отвори даден Интернет портал, браузърът на клиента трябва да замени името на портала, въведено от потребителя (напр. *www.example.com*) със съответстващия му IP адрес (напр. *208.77.188.166*), който играе ролята на уникален идентификатор на всеки сървър в глобалната мрежа. Ако даден хакер успее да повлияе на процеса на превръщане на името на сървъра в неговия IP адрес (англ. DNS resolution process), той може да промени IP адреса, който съответства на името на портала за Интернет-банкиране. По този начин, дори при коректно въведено име на сървъра, браузърът на жертвата ще бъде препратен към фалшивия банков портал, контролиран от хакера.

Фишингът представлява сериозна опасност поради факта, че ако жертвата не е внимателна, тя няма да разбере своевременно за кражбата на изключително важни лични и финансови данни и ще открие проблема едва тогава, когато с откраднатите данни е извършена злоупотреба и е късно щетите да бъдат поправени.

6.1.2 Недостатъци на традиционните технологии

Традиционните технологии за повишаване на сигурността при работа с Интернет портали разчитат на т.нар. криптографски сертификати (Фиг. 62), които са базирани на криптография с публични и частни ключове и използват криптиращи алгоритми подобни на RSA – алгоритъм на Rivest-Shamir-Adleman [20]. Криптографските сертификати могат да бъдат самоподписани (англ. self-signed) или да бъдат издадени от специална оторизираща компания (англ. certification authority, CA), която проверява и гарантира идентичността на собственика на Интернет портала.

Фиг. 62: Информация за криптографски сертификат на Интернет портал във Firefox

Криптографските сертификати имат за задача постигането на следните две главни цели:

1. Осигуряване на криптирана връзка по целия път от клиента до сървъра, така че да се предотврати нежелано подслушване на предаваните данни;
2. Предоставяне на механизъм за проверка на идентичността на собственика на Интернет портала, което спомага за откриването на опити за фишинг. Механизмът се базира на доверието, което крайните потребители имат в съответната CA – тя е гарант, че Интернет порталът се поддържа и предоставя на разположение от компанията, посочена в криптографския сертификат.

Недостатъкът на използването на криптографски сертификати в Интернет среда се крие в нарастващата сложност за крайните потребители. За да се осигури надеждна защита, потребителят трябва при посещение на Интернет портала да сравнява името на собственика, посочено в криптографския сертификат, с реалното име на банката или компанията (Фиг. 62). Освен това е необходимо потребителят да знае името на CA, чийто услуги собственикът на Интернет портала използва, и да проверява (пак при всяко посещение) дали криптографският сертификат е подписан точно от тази CA. Потребителят може да се довери на Интернет портала, само и единствено ако името на собственика на портала и на изписаната CA съвпадат с очакваните.

Повечето потребители не обръщат особено внимание на СА, която е издала сертификата. Това позволява на криминалните лица да регистрират автентично изглеждащ криптографски сертификат с друга СА, която не е толкова строга в процедурите за проверка на идентичността. Също така потребителите обикновено не обръщат достатъчно внимание на точното име на собственика на портала, показвано като част от информацията за криптираната връзка (Фиг. 62). Ако порталът използва сертификат, издаден на името на компания с име, подобно на реалното име на банката или институцията, повечето потребители не биха забелязали разликата – особено ако реалното име е сравнително дълго и трудно за разчитане.

Човешкият фактор улеснява значително заобикалянето на традиционните технологии за сигурност. Необходимо е изследването на нови алтернативи за подобряване на нивото на защита и предоставяне на по-сигурни Интернет-базирани банкови и бизнес портали за крайни потребители.

6.1.3 Предотвратяване на фишинг чрез криене на данни

Традиционните технологии за сигурност разчитат на криптографски подходи и техни основни цели са проверката на идентичността на собственика на портала, защитата на Интернет сървъра, който обслужва портала, от хакерски атаки и криптирането на връзката между портала и крайните клиенти. Важна компонента, която често не се взема предвид при цялостното проектиране на системите и технологиите за сигурност, е самото съдържание на портала. Традиционните схеми за сигурност третират съдържанието като пасивен обект, регулирайки единствено нивото на достъп до него. То не се използва за подобряване на сигурността и служи единствено на крайните потребители.

Технологиите за криене на данни позволяват активната употреба на съдържанието на Интернет портала за подобряване на цялостната сигурност и предотвратяване на различните вариации на фишинг. Така самото съдържание става допълнително средство за проверка на автентичността на портала и информацията, която той предоставя. Съдържанието на повечето модерни портали се състои от комбинация от различни видове мултимедия: текст, изображения, видео и т.н. Методите за криене на данни могат да вграждат информация в мултимедийното съдържание, която съдържа детайли за авторските права, данни за съответстващия му Интернет портал, традиционен криптографски сертификат или комбинация от трите. По този начин мултимедийното съдържание може неразделимо да се свърже със своето легитимно местоположение в определен Интернет портал. Освен това вграждането на данни в новосъздадените мултимедийни файлове може лесно да се автоматизира при публикуването им в даден Интернет портал.

Нека предположим, че дадена банка или институция използва методи за криене на данни за подобряване на сигурността на своя Интернет портал. За да получи достъп до личните и финансови данни на клиенти на банката или институцията чрез фишинг, по-

тенциалният хакер трябва да използва мултимедийното съдържание на оригиналния Интернет портал (което може лесно да се свали от портала и да се копира на друг сървър) или да създаде негова имитация. В първия случай, информацията, вградена в мултимедията, няма да съответства на новия фалшив Интернет портал. Във втория случай няма да присъства вградена информация. Двата случая са лесни за разпознаване от клиент (или клиентско приложение), който очаква да намери правилната идентифицираща информация в мултимедийното съдържание на Интернет портала.

От страната на крайния клиент, проверката на мултимедийното съдържание на портала на банката или институцията се извършва в идеалния случай автоматично чрез разширение (англ. extension, add-on или plug-in) на традиционните Интернет браузъри като напр. Mozilla Firefox. Разширението на браузъра извлича и сравнява информацията, вградена в мултимедийните файлове, с информацията за портала, от който те са заредени в браузъра. Ако се открие несъответствие, крайният потребител се уведомява за потенциален опит за фишинг.

Методите за криене на данни позволяват на всеки мултимедиен файл да съдържа своя собствена малка сигнатура (вж. дефиницията в раздел 1.1.2), която може да се сравни с различни характеристики на Интернет портала (напр. URL или IP адрес), от който се зарежда файла. Тъй като сигнатурата става неделима част от мултимедийния файл, потенциалните хакери по-трудно могат да ѝ противодействат в сравнение с разпространените криптографски подходи. Също така тази защита е напълно съвместима с клиенти, които не могат или не желаят да работят с допълнителните сигнатури, вградени в мултимедийните файлове.

6.1.4 Мрежова услуга за сертифициране

Модулните методи за криене на данни могат се предложат под формата на мрежова услуга чрез използването на подходящ интерфейс за мрежови услуги (вж. раздел 4.6.4). По този начин те могат да се включат като част от различни технологични инфраструктури в почти всеки съвременен Интернет-базиран сценарий. Тази мрежова услуга е по същество сертифицираща услуга, даваща възможност за надеждна проверка на автентичността на мултимедийни файлове в корпоративен интранет или в глобалната мрежа. Услугата помага за откриването на нелегални дейности, свързани със защитената мултимедия.

Сертифициращата услуга за криене на данни трябва да поддържа най-малко следните два основни случая на употреба (англ. use case):

1. Подписване на произволна мултимедия със сигнатури, определени от легитимните притежатели на авторските права;

2. Проверка на автентичността и сигнатурите на мултимедийните файлове и сравнението им с данните, идентифициращи Интернет порталите, които използват файловете (вж. раздел 6.1.5).

Услугата може се интегрира сравнително лесно като част от вече съществуващо цялостно решение за сигурност. Нейно типично приложение, което в този разгледан случай подобрява защитата на банкови портали срещу фишинг и разчита на гореспоменатия интерфейс за мрежови услуги, е представено на Фиг. 63.

Фиг. 63: Приложение на модулните методи като мрежова услуга за сертифициране

В общия случай банковите служители генерират и добавят ново мултимедийно съдържание към Интернет портала на банката, което след това се разглежда от крайните потребители (стъпки *1* и *4*). Добавянето на сертифициращата услуга за криене на данни в мултимедия води до необходимостта от следните допълнителни стъпки:

1. Новото мултимедийно съдържание се добавя в банковия портал от служител на банката.
2. Изображението се препраща от Интернет сървъра на банката към сертифициращата услуга, за да бъде снабдено с вградена сигнатура. Това препращане може да се задейства автоматично чрез разширение на Интернет сървъра – например нов модул, ако използваният Интернет сървър е Apache. Алтернатива е ръчното задействане чрез специално подготвен скрипт (написан на PHP, ASP, Perl и т.н.) от банковите служители, ако е желателна по-голяма степен на контрол върху процеса на вграждане на сигнатурите (вж. раздел 6.1.7).

3. Мултимедийният файл с вградена сигнатура се предава обратно към Интернет портала на банката. След това порталът прави файла достъпен за крайни потребители.
4. Крайните потребители свалят и преглеждат снабденото със сигнатура мултимедийно съдържание от банковия Интернет портал. Браузърите на крайните потребители разполагат със специално разширение, чиято цел е да позволи проверката на мултимедия посредством сертифициращата услуга за криене на данни (вж. раздел 6.1.8). Разширението анализира съдържанието на банковия портал и съставя списък на мултимедийните файлове.
5. Браузърното разширение препраща откритите мултимедийни файлове към съответния интерфейс на сертифициращата услуга за криене на данни. Препращането на мултимедията може да става автоматично за всички Интернет портали или да се задейства ръчно от крайния потребител чрез подаване на подходяща команда при посещението на даден портал (натискане на бутон, избиране на меню и т.н.).
6. При получаването на резултатите от проверката на сигнатурите, браузърното разширение има за задача да уведоми потребителя по подходящ начин, съответстващ на сериозността на проблема (ако е открит такъв).

Сертифициращата услуга за криене на данни работи на отделен сървър за приложения (англ. application server), съдържащ хардуер, оптимизиран за бързи изчисления. Той приема мултимедийното съдържание, изпратено от Интернет сървъра на банката чрез интерфейс за мрежови услуги и вгражда съответните сигнатури за проверка на автентичността. Сигнатурите в програмния прототип се състоят от XML файл, който съдържа всички данни, свързани със сигурността и проверката на автентичността. Ако това не е желателно, напр. поради малкия максимален размер на данните, които могат да бъдат вградени, е възможно алтернативно решение – вграждане на кратък идентификационен номер, който служи за връзка между мултимедийния файл и описващата го сигнатура, която се съхранява отделно от него (вж. раздел 6.1.6).

Сертифициращата услуга за криене на данни изпълнява проверката на мултимедийните файлове, като анализира вградените в тях сигнатури. За даден файл са възможни три различни случая:

1. Наличие на вградена сигнатура, която съответства на Интернет портала, който съдържа мултимедията. В този случай всичко е наред, крайният потребител се уведомява за успеха на проверката и може да продължи необезпокоявано работата си с Интернет портала.
2. Наличие на вградена сигнатура, която не съответства на Интернет портала, съдържащ мултимедията. В този случай сертифициращата услуга уведомява легитимния притежател на авторските права, идентифициран посредством сигнатурата, за възможно нарушение на авторските права и опит за фишинг. Едновременно с това крайният потребител се предупреждава за опасността.

3. Отсъствие на вградена сигнатура, което означава, че собственикът на посетения Интернет портал не използва сертифициращата услуга за криене на данни. Крайният потребител се уведомява за този факт. Ако той знае, че банката използва сертифициращата услуга, това означава, че посетеният Интернет портал е фалшива имитация и представлява опит за фишинг.

В случаите, в които браузърът не поддържа проверката на мултимедийни сигнатури – например поради липсващо разширение – вградените сигнатури не се проверяват и гореописаният механизъм за подобрена защита срещу фишинг не се използва. Въпреки това, мултимедийното съдържание се показва без проблем в браузъра на потребителя. Интернет порталът функционира, както се очаква, и функционалността на традиционните технологии за сигурност, базирани на криптографски сертификати, не е накърнена. Вграждането на мултимедийните сигнатури като част от самото мултимедийно съдържание на файловете гарантира пълна съвместимост на подобрената технология, базирана на сертифициращата услуга, с всички традиционни решения за сигурност.

Прототипната реализация на сертифициращата услуга за криене на данни използва Microsoft Internet Information Server (IIS), инсталиран на отделен сървър, който предоставя необходимата изчислителна мощност за изпълнение на модулните методи за криене на данни. IIS има за задача да обработи заявките за вграждане и проверка на сигнатурите, получени през специално разработен за тази цел интерфейс за мрежови услуги, използван за комуникация със сертифициращата услуга (вж. раздел 6.1.5). Този интерфейс прави функционалността за вграждане и извличане на данни, реализирана от модулните методи, използваема в контекста на гореописания сценарий за предотвратяване на опити за фишинг. Той прави възможно взаимодействието на модулните методи с другите компоненти от решението в сценария – сървърния модул, който е част от банковия Интернет портал, и разширението за проверка на сигнатурите, което е част от клиентските браузъри. По този начин модулната технология за криене на данни може да се интегрира сравнително лесно с ИТ-инфраструктурите, използвани от банката или институцията и нейните крайни клиенти.

В следващите раздели се описват комуникационният интерфейс, разработен за сертифициращата услуга за криене на данни, и информацията, която се вгражда като сигнатура в мултимедийното съдържание. Също така е представена практическа реализация на интеграцията на сертифициращата услуга с банковия Интернет портал и браузърите на крайните потребители.

6.1.5 Мрежов интерфейс на услугата за сертифициране

Аналогично на интерфейса за мрежови услуги, описан в раздел 4.6.4, е създаден нов интерфейс специално за сертифициращата услуга, който е насочен към проверката на автентичността на мултимедия. Сертифициращата услуга за криене на данни трябва

да е способна да обработи както вграждането на сигнатурите, така и проверката и анализа на вече вградени сигнатури. Вграждането на сигнатурите се осъществява посредством метода (операцията) на мрежовата услуга (англ. web service operation) *hideCopyrightInformationByImage*. Примерна SOAP заявка и отговор са показани съответно на Фиг. 64 и Фиг. 65. SOAP заявката, показана на Фиг. 64 предава следните параметри към мрежовата услуга:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <hideCopyrightInformationByImage xmlns="http://ilchev.net">
      <sourceImage>base64Binary</sourceImage>
      <destinationImageFormat>JPG or BMP</destinationImageFormat>
      <errorCorrection>boolean</errorCorrection>
      <withstandJPEGCompressionRatio>unsignedByte
      </withstandJPEGCompressionRatio>
      <copyrightHolder>string</copyrightHolder>
      <creationYear>short</creationYear>
      <URL>string</URL>
    </hideCopyrightInformationByImage>
  </soap:Body>
</soap:Envelope>
```

Фиг. 64: Примерна SOAP заявка за вграждане на сигнатура в изображение

1. Изображението, в което трябва да се вгради сигнатурата, кодирано в *base64* формат [126] (вж. и раздел 4.6.4) (*sourceImage*).
2. Форматът на изображението, получено след вграждането на сигнатурата - на този етап JPG или BMP (*destinationImageFormat*).
3. Булева стойност, която контролира опционалната употреба на кодове за корекция на грешки: *true*, ако се използва корекция на грешки, *false*, ако корекцията на грешки не е необходима (*errorCorrection*).
4. Максималното ниво на JPEG компресия, което може да се приложи върху крайното изображение, върнато като резултат от заявката, без това да разруши вградената сигнатура (*withstandJPEGCompressionRatio*).
5. Името на легитимния притежател на авторските права върху изображението (*copyrightHolder*).
6. Годината на създаване на изображението (*creationYear*).
7. Стандартизиран идентификатор на ресурси (англ. uniform resource locator, URL), който посочва точния адрес на изображението в Интернет портала, където то ще бъде легитимно качено и направено достъпно за крайни клиенти (*URL*).

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <hideCopyrightInformationByImageResponse
xmlns="http://ilchev.net">
      <hideCopyrightInformationByImageResult>base64Binary
      </hideCopyrightInformationByImageResult>
    </hideCopyrightInformationByImageResponse>
  </soap:Body>
</soap:Envelope>
```

Фиг. 65: Примерен SOAP отговор на заявката от Фиг. 64

Описаните параметри могат да бъдат разделени в две общи категории – параметри, определящи начина на изпълнение на самите модулни методи за криене на данни (параметри *1* до *4*), и параметри, свързани с проверката на автентичността на изображението и авторските права (параметри *5* до *7*). Втората група параметри може лесно да се допълни с друга съществена информация като например списък от IP адреси на Интернет сървърите, където изображението може легално да се използва, или криптографски сертификат (вж. раздел 6.1.6).

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <unHideCopyrightInformationByURL xmlns="http://ilchev.net">
      <imageURL>string</imageURL>
    </unHideCopyrightInformationByURL>
  </soap:Body>
</soap:Envelope>
```

Фиг. 66: Примерна SOAP заявка за проверка на сигнатура, вградена в изображение

Примерен отговор на разгледаната SOAP заявка е показан на Фиг. 65. Той съдържа полученото изображение с вградена сигнатура, кодирано в *base64* формат. Проверката на вградените сигнатури се извършва от метода (операцията) на мрежовата услуга *unHideCopyrightInformationByURL*. Примерна SOAP заявка и отговора са показани съответно на Фиг. 66 и Фиг. 67.

Единственият използван параметър в заявката е *imageURL* – URL на изображението, чиято сигнатура трябва да се провери. SOAP отговорът връща като резултат XML документ, съдържащ извлечената сигнатура (ако такава бъде открита в изображение-

то). XML документът има представената в раздел 6.1.6 структура, която лесно може да се разшири, за да включва допълнителна информация, която се отнася до процеса на проверка на автентичността – например дата и час на вграждане на сигнатурата.

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <unHideCopyrightInformationByURLResponse
xmlns="http://ilchev.net">
      <unHideCopyrightInformationByURLResult>xml
      </unHideCopyrightInformationByURLResult>
    </unHideCopyrightInformationByURLResponse>
  </soap:Body>
</soap:Envelope>
```

Фиг. 67: Примерен SOAP отговор на заявката от Фиг. 66

Следващият раздел описва структурата и съдържанието на информацията за проверка на автентичността, която се използва като сигнатура в представения прототип. Тя се съставя от сертифициращата услуга за криене на данни на базата на входящите параметри и след това се вгражда в мултимедийното съдържание.

6.1.6 Вграждана информация за проверка на автентичността

Този раздел дискутира информацията за проверка на автентичността, вграждана като сигнатура от сертифициращата услуга за криене на данни. Съществуват общо два основни подхода, които могат да се реализират. Те постигат различен компромис между размера на вградената информация и лекотата и гъвкавостта на реализацията.

Първият подход вгражда цялата информация в мултимедийното съдържание под формата на XML файл. XML файлът има относително голям размер (обикновено около няколкостотин байта), но кодира с висока степен на сигурност всички необходими данни за проверка на автентичността в мултимедийното съдържание. По време на проверка на сигнатурата не е необходимо да се свалят други допълнителни данни. Мултимедията сама по себе си е достатъчна за извършване на проверката. Този подход е лесен за използване и предлага голяма степен на гъвкавост, тъй като структурата на XML файла може бързо да се промени, за да се реагира на промени в изискванията на потребителите. Основният недостатък е сравнително големият размер на вградената като сигнатура информация. Структурата на XML-базираната информация за проверка на автентичността се дефинира чрез стандартизиран от W3C формат за задаване и проверка на структурата на XML документи (англ. XML Schema Definition, XSD) [174].

Основният елемент в началото на XSD схемата (англ. root element) е именуван *ImageInformation* (Фиг. 68). Той може да съдържа:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" attributeFormDefault="unqualified">

<xs:element name="ImageInformation"> <xs:complexType><xs:sequence>
        <xs:element name="Copyright" type="TCopyright" minOccurs="0"/>
        <xs:element name="Location" type="TLocation" minOccurs="0"
                maxOccurs="unbounded"/>
        <xs:element name="Certificate" type="xs:base64Binary"
                minOccurs="0" maxOccurs="unbounded"/>
</xs:sequence> </xs:complexType> </xs:element>

<xs:complexType name="TCopyright"> <xs:sequence>
        <xs:element name="Year" type="xs:decimal" minOccurs="0"/>
        <xs:element name="Author" type="xs:string" minOccurs="0"
                maxOccurs="unbounded"/>
</xs:sequence> </xs:complexType>

<xs:complexType name="TLocation"> <xs:sequence>
        <xs:element name="Name" type="xs:string" minOccurs="0"/>
        <xs:element name="URL" type="xs:string" minOccurs="0"/>
        <xs:element name="ServerIP" type="xs:string" minOccurs="0"/>
</xs:sequence> </xs:complexType>

</xs:schema>
```

Фиг. 68: Формат на вградената информация за проверка на автентичността - XSD схема

1. Нула или един елемент, описващ легитимния притежател на авторските права (елемент *Copyright*);
2. Нула, един или повече елементи, идентифициращи Интернет порталите, където изображението може легално да се използва и показва на крайни потребители (елементи *Location*);
3. Нула, един или повече криптографски сертификати, имащи отношение към изображението, могат да бъдат съхранени в *base64* формат (елементи *Certificate*).

Елементът *Copyright* може да съдържат от своя страна два типа под-елементи:

1. Първият тип под-елементи (нула или един брой) съхранява годината на създаване на изображението (елемент *Year*).

2. Вторият тип под-елементи (нула, един или повече броя) съхранява списък на притежателите на авторските права върху изображението (елементи *Author*).

Всеки елемент *Location* може да съдържа три незадължителни под-елемента, които взети заедно идентифицират еднозначно даден Интернет портал:

1. Първият под-елемент задава името на Интернет портала (елемент *Name*).
2. Вторият под-елемент съхранява URL адреса на Интернет портала (елемент *URL*).
3. Последният под-елемент съдържа основния IP адрес на сървъра, който обслужва Интернет портала (елемент *ServerIP*).

Примерен XML документ, следващ гореописаната XSD схема, е показан на Фиг. 69. Той съдържа информация за автора, годината на създаване, както и за Интернет портала, на който JPEG изображението ще се публикува в Интернет. След като този документ се вгради в JPEG изображението, той позволява проверката на автора на изображението и на адреса на Интернет портала, от който то би трябвало да е свалено, без да са необходими допълнителни данни.

```
<?xml version="1.0" encoding="UTF-8"?>
<ImageInformation xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <Copyright>
    <Year>2007</Year>
    <Author>Svetozar Ilchev</Author>
  </Copyright>
  <Location>
    <URL>ilchev.net/StegiWeb/DSC00014.JPG</URL>
  </Location>
</ImageInformation>
```

Фиг. 69: Информация за проверка на автентичността - примерен XML документ

Вторият подход за реализирането на проверка на автентичността на мултимедийното съдържание се базира на вграждането на сравнително къс уникален идентификационен номер. Този номер е различен за всеки подписан мултимедиен файл и представлява удобно средство за идентификация на отделните мултимедийни копия. Проверката на мултимедийното съдържание, подписано чрез идентификационен номер, се нуждае от външна база данни, която съдържа същинската информация за проверка на автентичността като например имената на авторите, годината на създаване, местоположенията (адресите) на публикуване на файла в Интернет и други данни за Интернет порталите, предоставящи достъп до файла на крайни потребители. Тази информация трябва да се съхранява в базата данни, като мултимедийният идентификационен номер се използва като ключ за достъп (англ. (primary) key).

По време на процеса на проверка на сигнатурата, сертифициращата услуга за криене на данни първо извлича идентификационния номер, вграден в мултимедийното съдържание. След това тя се обръща към външната база данни, като използва идентификационния номер, за да прочете информацията за проверка на автентичността, принадлежаща към мултимедията. Прочетената информация се форматира по подходящ начин и след анализ резултатите се предоставят на крайния потребител.

Основното предимство на този подход се състои в сравнително малката дължина на вграждания мултимедиен идентификационен номер – обикновено в рамките на около 100 бита. Основният недостатък е използването на външна база данни, което е свързано с някои важни съображения. Базата данни е допълнителен обект, различен и независим от мултимедийното съдържание. Тя образува т.нар. единична точка на повреда (англ. single point of failure). За сравнение първият подход с вграждане на XML документ се нуждае единствено от споделеното в Интернет мултимедийно съдържание.

От първостепенна важност при втория подход е обезопасяването на базата данни по надежден начин. Ако потенциален хакер успее да получи контрол над нея, той може да фалшифицира информацията за проверка на автентичността, като по този начин компрометира резултатите от проверката. Също така трябва да се гарантира защита срещу неочаквана загуба на информацията вследствие на повреден хардуер, пожар и т.н. Освен това достъп до базата данни е необходим при обработката на всяка нова мултимедия, предадена към услугата за сертифициране за вграждане или проверка на сигнатурата. Ако базата данни е недостъпна по една или друга причина, мултимедията няма да може да се обработи.

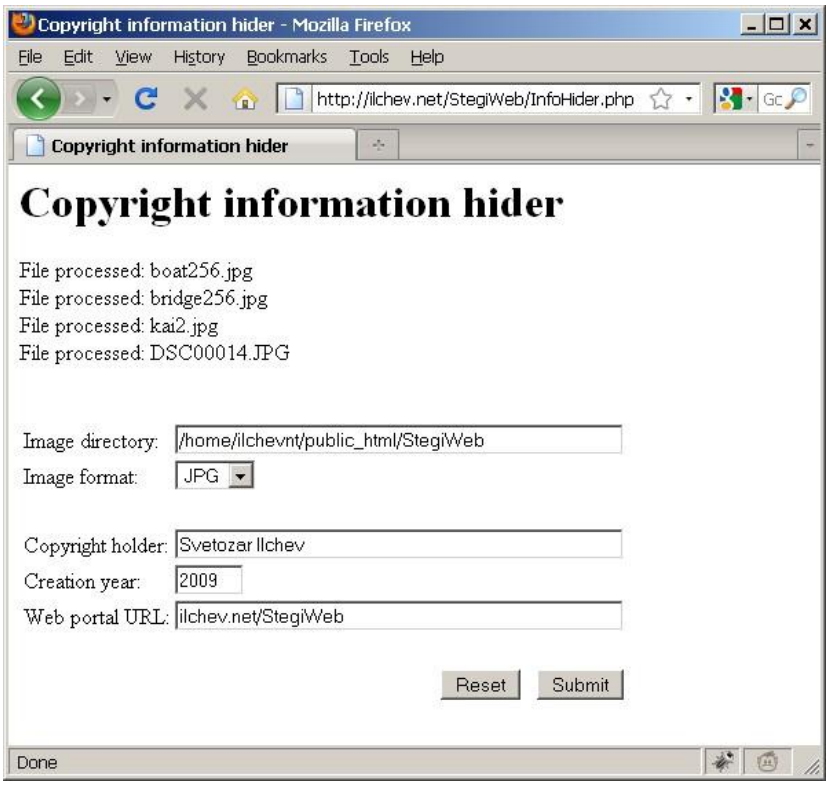
И двата разгледани подхода предлагат осъществими алтернативи за реализиране на вграждането и прегледа на информация за проверка на автентичността и са приложими в разглеждания сценарий за предотвратяване на опити за фишинг. На този етап прототипът, реализиран като част от дисертацията, използва подхода с вграждане на XML документ поради по-добрата гъвкавост и възможност за внасяне на бързи изменения.

6.1.7 Интеграция на услугата с банковия Интернет портал

Интеграцията на сертифициращата услуга за криене на данни с банковия Интернет портал може лесно да се осъществи с помощта на интерфейса за мрежови услуги, описан в раздел 6.1.5. За тази цел е необходимо Интернет сървърът на банката да препраща всички качени неподписани мултимедийни файлове към сертифициращата услуга, преди те да са направени достъпни за крайните потребители. Препращането на мултимедийните файлове може да се осъществи *автоматично* посредством създаването на ново разширение на Интернет сървъра – например Apache модул, който наблюдава всички мултимедийни файлове, качени на Интернет портала. Когато бъде качен нов мултимедиен файл (в този случай изображение), достъпът на крайни клиенти до него временно се забранява. След това файлът се препраща към метода (операцията) на

мрежовата услуга *hideCopyrightInformationByImage* (описана в раздел 6.1.5), която вгражда подходяща сигнатура в мултимедийното съдържание и го предава обратно на сървърното разширение (модул). Оригиналният файл се заменя с файла, получен от сертифициращата услуга, и след това достъпът на крайни клиенти до него се разрешава.

Друго алтернативно решение е използването на *команден скрипт* (англ. shell script), който работи във фонов режим (англ. daemon) или се стартира на определени интервали (англ. cron job). Скриптът проверява предварително дефинирани директории на файловата система за нови мултимедийни файлове и, ако бъдат открити такива, те се препращат към сертифициращата услуга за вграждане на подходящи сигнатури. Когато процесът на вграждане на сигнатурите бъде приключен и файловете бъдат изпратени обратно, командният скрипт ги копира в предварително зададени директории на файловата система, които са достъпни за крайни потребители.



Фиг. 70: Ръчно вграждане на сигнатури в изображения - GUI

Трета възможност е *ръчното стартиране* на вграждането на сигнатури в качените мултимедийни файлове. По този начин служител на банката може да вземе решение кои файлове трябва да бъдат подписани и кои не. За тази цел е необходим подходящ графичен потребителски интерфейс (англ. Graphical User Interface, GUI), посредством който се предоставя достъп до качените мултимедийни файлове и възможност за управление на процеса на вграждане на сигнатурите. Ползвателят на интерфейса трябва да има възможност да дефинира някои важни данни за мултимедийните файлове, из-

ползвани в сигнатурата като например името на притежателя на авторските права, годината на създаване на файла или местоположението (URL адрес) на файла в Интернет портала на банката, където той ще бъде легитимно качен и направен достъпен за крайни потребители.

Прототипен графичен потребителски интерфейс, който позволява ръчното стартиране на процеса на вграждане на сигнатури в изображения, е показан на Фиг. 70. Илюстрирано е успешното вграждане на сигнатури в *4* изображения от директория, указана от текстовото поле *Image directory*. Файловете са кодирани в JPEG формат, както се вижда от падащото меню *Image format*. Трите текстови полета в долния край на графичния потребителски интерфейс задават имената на притежателя на авторските права (текстово поле *Copyright holder*), годината на създаване на файловете с изображения (текстово поле *Creation year*) и легитимното местоположение (URL адрес) на файловете в Интернет (текстово поле *Web portal URL*).

```
$ws = new SoapClient(WSDL_URL, array('trace' => 1, 'encoding' =>
'UTF-8', 'soap_version' => SOAP_1_1));

$params = array('sourceImage' => $file_data, 'destinationImageFormat'
=> $imgFormat, 'errorCorrection' => true,
'withstandJPEGCompressionRatio' => JPEG_Q, 'copyrightHolder' =>
$copyHolder, 'creationYear' => $copyYear, 'URL' => $urlFile);

$new_file_data = $ws->__soapCall('hideCopyrightInformationByImage',
array('parameters' => $params), array(), null, $outputHeaders);
```

Фиг. 71: PHP скрипт – стартиране на метода *hideCopyrightInformationByImage*

Графичният потребителски интерфейс е Интернет-базиран: HTML формуляр, който стартира изпълнението на предварително написан PHP скрипт. Този скрипт е част от Интернет портала и осъществява комуникацията със сертифициращата услуга за криене на данни: избор и препращане на неподписаните изображения, задаване на параметри на сигнатурата и запомняне на подписаните изображения в подходяща директория. Най-важната част от кода на PHP скрипта е показана на Фиг. 71. Тази част извършва същинското стартиране на метода (операцията) на мрежовата услуга *hideCopyrightInformationByImage* посредством SOAP разширението в PHP (класа *SoapClient*). Най-напред се създава SOAP клиентски обект, достъпен чрез променливата *$ws*. При създаването на обекта се определя използваната версия на SOAP протокола (SOAP 1.1) и вида на кодирането на текста (UTF-8). Следващата стъпка е създаването на асоциативния масив (англ. associative array) *$params*, който съдържа всички параметри, които трябва да бъдат предадени на интерфейса за мрежови услуги (за пълно описание, вж. раздел 6.1.5). Последните два реда на Фиг. 71 показват стартитането на метода *hideCopyrightInformationByImage* с параметрите от масива *$params*. Полученото

изображение, съдържащо вградената сигнатура, е достъпно чрез променливата *$new_file_data*.

С помощта на интерфейса за мрежови услуги и PHP кода, показани съответно на Фиг. 70 и Фиг. 71, се извършва сравнително лесна интеграция на сертифициращата услуга за криене на данни дори при използването на споделен Интернет сървър (англ. shared hosting), където обикновено не се разрешава пълен достъп до сървъра или промяна на глобалната конфигурация чрез добавяне на нови сървърни разширения или модули.

Всеки начин на интегриране на сертифициращата услуга има определени предимства и недостатъци. Интегрирането посредством сървърно разширение или команден скрипт, работещ във фонов режим като демон, е добро решение за автоматично обработване на повечето качени мултимедийни файлове и вграждане на стандартно-зададени сигнатури. За някои файлове може да се наложи промяна на стандартните параметри на сигнатурата, което лесно може да се осъществи посредством Интернет базирания графичен потребителски интерфейс. Във всички случаи интерфейсът за мрежови услуги на сертифициращата услуга за криене на данни осигурява лесна интеграция с почти всички модерни Интернет-базирани решения.

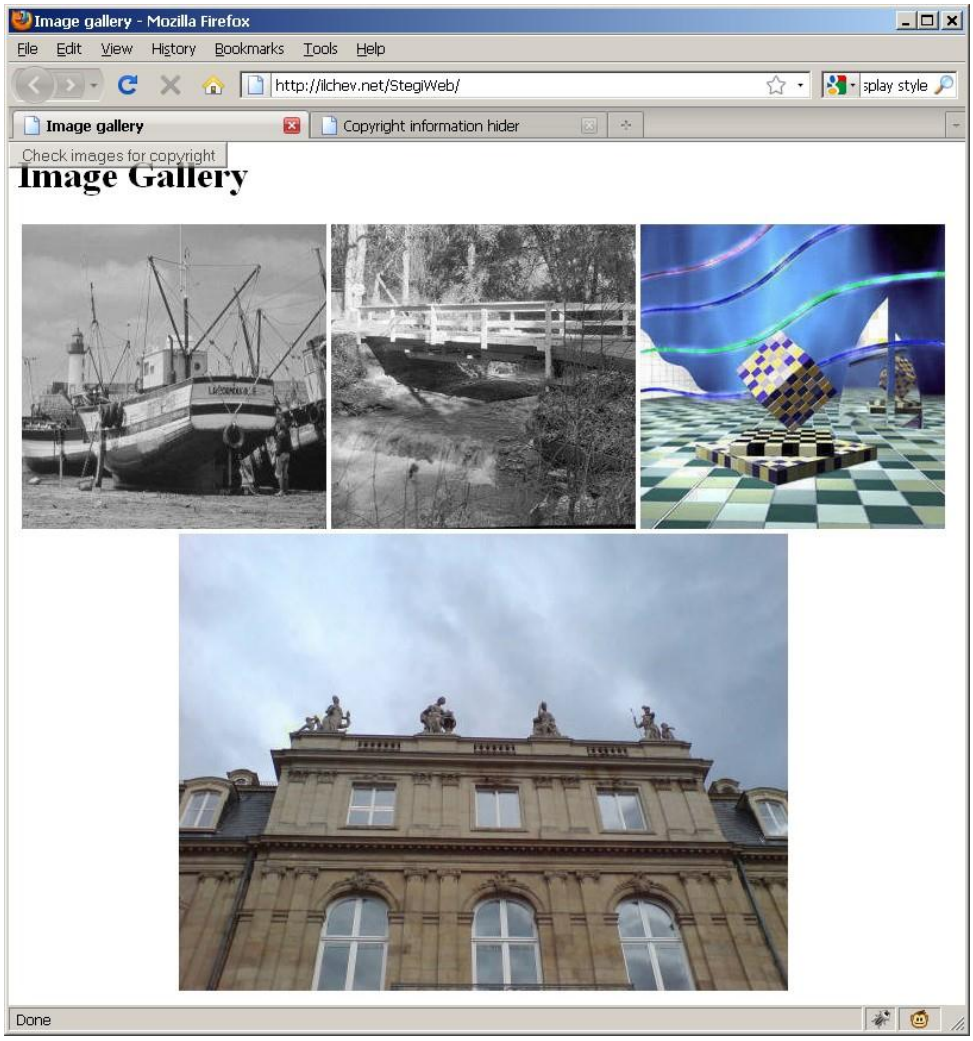
6.1.8 Интеграция на услугата с клиентски Интернет браузъри

Интеграцията на сертифициращата услуга за криене на данни с Интернет браузърите на крайните потребители зависи до голяма степен от разширяемостта на различните браузъри, съществуващи на пазара, с нова функционалност. Повечето модерни браузъри поддържат (в една или друга форма) браузърни разширения и потребителски скриптове (англ. user scripts), използвани за добавяне на нови функции в браузъра по желание на потребителите. Браузърните разширения работят като част от самия браузър. Те могат да променят поведението и изгледа на браузъра и имат достъп до повечето от неговите вътрешни библиотеки чрез специално разработен за целта програмен интерфейс (англ. Application Programming Interface, API). Браузърните разширения могат да бъдат написани на традиционен език за програмиране на десктоп приложения като C/C++, на скриптов език, разработен за целите на Интернет-базирани приложения като JavaScript или с помощта на комбинация от тях.

Потребителските скриптове работят като част от Интернет страницата, която е заредена в даден момент в браузъра. Те най-често са написани на JavaScript и наподобяват до голяма степен нормалния JavaScript код, използван от разработчиците на самите Интернет страници. Потребителските скриптове имат достъп до обектния модел на Интернет страницата (англ. Document Object Model, DOM) и могат да го променят. Разликата между тях и стандартно използвания JavaScript код се състои в това, че потребителските скриптове не се зареждат от самата Интернет страница, а се задават

предварително в браузъра и се активират за определени Интернет портали с предварителното съгласие на крайния потребител.

Както браузърните разширения, така и потребителските скриптове могат да бъдат използвани за автоматизирана интеграция на сертифициращата услуга за криене на данни с браузърите на крайните потребители. Браузърните разширения предоставят повече гъвкавост при реализирането на интеграцията и дават възможност да се направи по-мощен (графичен) потребителски интерфейс, но са значително по-сложни за създаване в сравнение с потребителските скриптове.



Фиг. 72: Графичен интерфейс на потребителски скрипт преди проверка на сигнатурите

При автоматизираната интеграция сигнатурите на всички използвани мултимедийни файлове ще бъдат автоматично препратени за проверка на вградените сигнатури към сертифициращата услуга. Проверката се извършва като част от процеса на зареждане на страницата, без да са необходими специални действия от страна на крайния потребител. Той ще бъде уведомен за подобрената сигурност и за резултатите от про-

верката, най-вече ако е открито несъответствие на сигнатурите на мултимедията с Интернет портала, което е индикация за вероятен опит за фишинг.

Друго алтернативно решение е да се предложи на крайния потребител прост графичен потребителски интерфейс, който той може да използва, за да се допита до сертифициращата услуга дали мултимедията в конкретен Интернет портал съдържа вградени сигнатури и, ако такива са налични, дали те съответстват на заредения портал. По този начин потребителят може да избира интерактивно за кои Интернет портали да се извърши проверка на мултимедийното съдържание, а графичният интерфейс има за задача да анализира и покаже резултатите, получени от сертифициращата услуга, в подходяща форма.

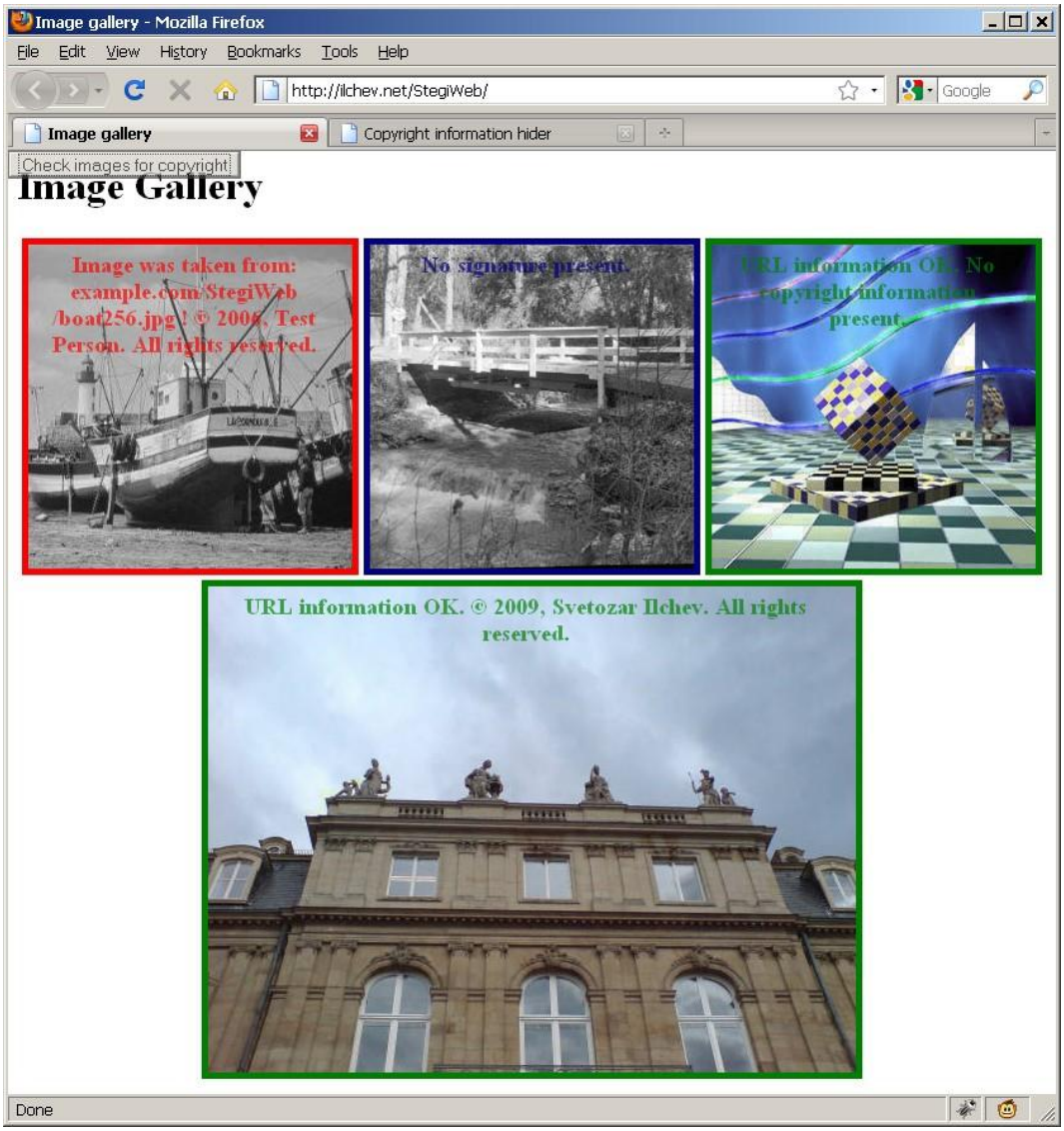
И двата вида решения – автоматизирано или ръчно стартиране на проверката на мултимедийни сигнатури – са осъществими на практика. Кое от тях трябва да се предпочете, зависи от конкретния случай на приложение, както и от ресурсите, необходими за изпълнението на сертифициращата услуга. Те от своя страна дават пряко отражение върху модела за плащане на услугата, дефиниран в договора, сключен между доставчика на услугата и потребителите.

В прототипната реализация в тази глава е избран втория тип решение – ръчно стартиране на процеса за проверка на мултимедийните сигнатури от крайния потребител. Програмната реализация се базира на потребителски скрипт, написан на JavaScript, който работи с широко разпространения Интернет браузър Mozilla Firefox. Тъй като самият Firefox не поддържа потребителски скриптове, е използвано браузърното разширение GreaseMonkey [175], което предоставя поддръжка на потребителски скриптове във Firefox. Други широко разпространени браузъри, които поддържат потребителски скриптове са Internet Explorer, Opera и Safari [176]. Графичният потребителски интерфейс, реализиран от потребителския скрипт, е показан на Фиг. 72 и Фиг. 73.

Фиг. 72 показва Интернет страница, съдържаща галерия от изображения. Графичният интерфейс на потребителския скрипт се състои от полупрозрачния бутон *“Check images for copyright”*, който се намира в горната лява част на прозореца на браузъра. Чрез този бутон потребителят може да стартира проверка на сигнатурите на изображенията. Когато бутонът се задейства, потребителският скрипт първо анализира HTML кода на Интернет страницата, заредена в браузъра в този момент. Скриптът идентифицира всички JPEG изображения, използвани в Интернет страницата и ги предава към сертифициращата услуга за криене на данни, която извършва същинската проверка на вградените сигнатури. След като проверката на сигнатурата приключи за дадено изображение, потребителският скрипт получава като отговор от сертифициращата услуга сигнатурата, извлечена от изображението. Ако такава сигнатура отсъства, се получава подходящо статус съобщение.

Фиг. 73 показва Интернет страницата след завършването на проверката на вградените сигнатури. Първото изображение на горния ред е маркирано с червена рамка. Тя показва, че изображението съдържа сигнатура, която не съответства на Интернет пор-

тала, от който изображението е заредено. В границите на червената рамка потребителският скрипт изписва прочетената от сигнатурата информация, която има отношение към авторските права и оригиналното местоположение на изображението. По този начин се предоставят на потребителя повече подробности относно потенциалния опит за фишинг или кражба на мултимедия.



Фиг. 73: Графичен интерфейс на потребителски скрипт след проверка на сигнатурите

Второто изображение на първия ред е маркирано със синя рамка. Тя показва, че не е намерена вградена сигнатура или че сертифициращата услуга е недостъпна. Съответното съобщение е изписано в границите на рамката.

Третото изображение на първия ред е маркирано със зелена рамка. Тя отбелязва присъствието на сигнатура, която съответства на Интернет портала, от който изображението е заредено. Сигнатурата съдържа данни относно легитимното местоположение на изображението (съответстващо на текущия Интернет портал). Съгласно инфор-

мацията, изведена в зелената рамка, тя не съдържа данни за авторските права. Тази частична сигнатура се възползва от гъвкавостта, предоставена от XML формата на информацията за проверка на автентичността.

Единственото изображение на втория ред също е маркирано със зелена рамка. То съдържа сигнатура, която съответства на заредения Интернет портал. Също така тази сигнатура съдържа информация за авторските права. Тя се показва на потребителя в границите на зелената рамка, обкръжаваща изображението.

```
GM_xmlhttpRequest({

method: 'POST',
url:
        'http://xxx.xxx.xxx.xxx:8888/StegiWeb/StegiWeb.asmx/unHideCopyrig
htInformationByURL',

headers: {
        'User-agent': 'Mozilla/4.0 (compatible) Greasemonkey/0.3',
        'Accept': 'application/xml,text/xml',
        'Content-type': 'application/x-www-form-urlencoded'
},
data: 'imageURL=' + escape(url),

onerror: function(responseDetails) {
        tagImage("Service offline", "navy", imgEl);
},

onload: function(responseDetails) {
        var parser = new DOMParser();
        try{

                var dom =
parser.parseFromString(responseDetails.responseText,"application/xml");
                checkForParsingErrors(dom);
                checkSignature(dom, url, imgEl);
        }catch(e){
                tagImage("No signature present", "navy", imgEl);
        }
}
});
```

Фиг. 74: Потребителски скрипт – стартиране на мрежов метод

Една от основните цели при разработването на графичния потребителски интерфейс чрез потребителския скрипт е минимизирането на промените, правени динамично в изгледа на Интернет портала, което повишава съвместимостта с различните банкови

портали. Графичният интерфейс е направен възможно най-лесен и интуитивен за употреба от крайни потребители. Едно натискане на полупрозрачния бутон *"Check images for copyright"* стартира проверката на сигнатурите. След нейното завършване, се използват три основни цвята, които представят резултатите по интуитивен начин на крайния потребител. Червена рамка на изображението съобщава за потенциален проблем със сигурността, синя рамка означава, че проверката не може да се извърши за дадено изображение поради липсваща сигнатура или недостъпна сертифицираща услуга, зелена рамка показва, че изследваното изображение съдържа сигнатура, която съответства на Интернет портала.

Частта от кода на потребителския скрипт, която изпраща същинската заявка за проверка на сигнатурата към сертифициращата услуга за криене на данни е показана на Фиг. 74. Методът (операцията) на мрежовата услуга, която се използва в този случай е *unHideCopyrightInformationByURL* (вж. раздел 6.1.5). Потребителският скрипт използва възможността, предоставена от платформата .NET, за стартиране на методи (операции) на мрежови услуги посредством HTTP POST заявки. По този начин се избягва необходимостта от създаване на класическите, но и по-сложни като формат SOAP. Това улеснява програмната реализация на потребителския скрипт (JavaScript код, зареждан автоматично от браузъра като част от текущата Интернет страница).

HTTP POST заявката използва т.нар. *Ajax* механизъм за асинхронна обмяна на данни във фонов режим – възможност, предоставяна от почти всички съвременни Интернет браузъри [177]. В конкретния случай е използвана функцията *GM_xmlhttpRequest*, предоставена от браузърното разширение GreaseMonkey. Главната цел на тази функция е да предостави на потребителските скриптове унифициран начин за извършване на HTTP *Ajax* заявки в различни браузърни среди, които често съдържат отклонения от W3C стандарта [178] при реализацията на *Ajax* механизма. Параметрите, предавани на функцията *GM_xmlhttpRequest* са както следва:

1. Вид на HTTP заявката, който е равен на *POST* в този случай (*method*);
2. Местоположението (URL-адресът) на метода (операцията) на мрежовата услуга *unHideCopyrightInformationByURL* (*url*);
3. Допълнителна информация за клиента, изпращана под формата на направляващо описание като част от HTTP заявката (*headers*);
4. Параметрите, изпращани към сертифициращата услуга за криене на данни (*data*). В случая на метода (операцията) *unHideCopyrightInformationByURL*, единственият необходим параметър е URL-адресът на JPEG изображението, което трябва да бъде проверено от сертифициращата услуга (е *imageURL*).
5. Идентификатор (адрес) на функция, която се стартира, ако не може да се установи връзка със сертифициращата услуга за криене на данни (*onerror*). Помощната функция *tagImage* създава дебелата зелена, червена или синя рамка около всяко обра-

ботено изображение и извежда на екрана текста, показващ статуса на проверката (“Service offline”), в същия цвят като рамката.

6. Идентификатор (адрес) на функция, която се стартира, след като сертифициращата услуга за криене на данни върне успешно резултат от проверката на сигнатурата (*onload*). Функцията анализира получения резултат и използва гореспоменатата функция *tagImage*, за да уведоми крайния потребител по подходящ графичен начин.

Потребителският скрипт представлява последния необходим компонент за интегрирането на сертифициращата услуга за криене на данни в разгледания сценарий за предотвратяване на фишинг. Когато крайният потребител посети легитимния портал на банката, потребителският скрипт ще маркира всички изображения със зелени рамки. Изображенията са подписани със сигнатури, вградени от сертифициращата услуга за криене на данни, както е описано в раздел 6.1.7. Сигнатурите съответстват на банковия портал.

Ако потребителят посети фалшив банков портал вследствие на опит за фишинг, потребителският скрипт ще маркира изображенията на портала с червени или сини рамки – по подобие на първите две изображения във Фиг. 73. Изображенията, които са взети от оригиналния банков портал съдържат вградени сигнатури. Посредством сертифициращата услуга за криене на данни ще се констатира, че те не съответстват на фалшивия портал. Потребителският скрипт ще уведоми крайния потребител за това нарушение на авторските права с помощта на червена рамка около съответните изображения. В случай на реално приложение, сертифициращата услуга също така ще уведоми банката за откритията проблем. В останалите изображения, използвани от фалшивия банков портал, няма да бъдат намерени вградени сигнатури и потребителският скрипт ще ги маркира със сини рамки и съобщение за липсваща сигнатура. Отсъствието на вградени сигнатури също е сигнал за потенциален опит за фишинг, тъй като легитимният банков портал задейства автоматично вграждането на сигнатури във всички качени файлове.

Различията в цветовете на рамките (зелена – всичко е наред, червена или синя – потенциален опит за фишинг) образуват интуитивен графичен потребителски интерфейс, чрез който крайният потребител може лесно да разграничи фалшивия банков портал от истинския.

Потребителският скрипт (или браузърно разширение) може да предприеме и други действия освен добавянето на подходящо оцветени рамки към изображенията. Той може да забрани достъпа до Интернет портали, чиито изображения не са снабдени с правилните сигнатури. Ако подписани изображения бъдат намерени на фалшив Интернет портал, потребителският скрипт може да използва информацията от сигнатурите, за да пренасочи Интернет браузъра на потребителя към адреса на истинския банков портал. Също така той може да уведоми съответните органи на властта за откритите нарушения. Крайната цел е да се осигури проста за използване и в същото време ефективна

защита срещу опити за фишинг, която да облагодетелства крайните потребители и да не изисква от тях специализирани познания в сферата на информационните технологии.

6.2 Защита на мултимедийната интелектуална собственост на информационни агенции

Този сценарий разглежда предимствата на модулните методи за криене на данни и сертифициращата услуга за криене на данни, представена в предишния раздел, по отношение на защитата на мултимедийно съдържание, публикувано в Интернет. Това е класически случай на приложение на цифровото маркиране. Основни целеви групи на подобрената защита са новинарски агенции, фотожурналисти, филмови продуценти - потребителски групи, предоставящи онлайн достъп до мултимедия в глобалната мрежа.

6.2.1 Недостатъци на традиционните подходи

Защитата на правата на интелектуална собственост за мултимедийно съдържание, което е достъпно за произволни крайни потребители в Интернет, е трудна задача. Новинарските агенции обикновено желаят да предоставят на Интернет потребителите единствено правото да разглеждат публикуваното мултимедийно съдържание, като копирането му и повторното му разпространяване в Интернет са строго забранени. На този етап не съществуват надеждни технически решения и механизми, които могат да предотвратят нелегалното разпространение на мултимедийно съдържание. Единствената възможност за новинарските агенции е те да маркират мултимедийното съдържание като своя собственост и след това да водят борба срещу нелегалното копиране и разпространение на мултимедията не с технически, а с правни средства.

Съществуват два традиционни подхода, използвани от новинарските агенции за деклариране на авторските права върху мултимедийно съдържание и по специално върху JPEG изображения. Първият подход разчита на мета-данните (описание на данните), които обикновено се съхраняват в направляващото описание на мултимедийния файл. Те съдържат информация за файла, която би могла да бъде от значение за програмното приложение, което го обработва, или крайния потребител. В случая на JPEG (и TIFF) изображения, от значение са т.нар. EXIF (англ. Exchangeable Image File Format) мета-данни, записвани в направляващото описание на файла [179]. Те съхраняват различни параметри и свойства на изображението като дата и час на създаването му, настройки на фотоапарата, географска ширина и дължина и др. Част от EXIF данните може да бъде и текстов низ, съдържащ информация за притежателя на авторските права. Използвайки този текстов низ, новинарските агенции могат да декларират пра-

вото си на собственост върху всяко JPEG (или TIFF) изображение, което публикуват в Интернет.

Основният недостатък на този подход е, че EXIF данните не могат да изпълняват функцията на защитен механизъм. Тяхната основна цел е да предоставят прост за използване начин за прикачване на полезна информация към изображенията. Поради това EXIF данните могат много лесно да бъдат открити, прочетени, променени или премахнати от почти всяка програма за разглеждане и обработка на изображения. Освен това, ако форматът на изображението бъде променен във формат, който не поддържа EXIF или друга подобна структура от мета-данни, информацията, която EXIF съхранява, ще бъде загубена.

Фиг. 75: Идентификация на притежателя на авторските права чрез видим текст

Вторият подход за деклариране на авторските права върху изображение или видео файл е поставянето на лого или текст, видими за крайния потребител, в мултимедийното съдържание (Фиг. 75). Логото или текстът идентифицират притежателя на авторските права. Те могат да бъдат полупрозрачни, черно-бели или цветни, и да се намират близо до някой от краищата или в средата на изображението. Този подход има два ясно изразени недостатъка:

1. Лошото естетическо впечатление, което видимите текст и лого правят на крайния потребител. Те най-често се добавят автоматично към мултимедийното съдържание и има вероятност да закрият важна част от изображението или видео-кадъра, като по този начин се намалява значително неговата артистична или информативна стойност.
2. Лесното откриване и премахване на текста/логото чрез специализирано програмно осигуряване за обработка на изображения или видео файлове като програмния па-

кет *Creative Suite* на компанията *Adobe*. Текстът и/или логото могат да бъдат филтрирани или просто изрязани, ако се намират близо до краищата на мултимедийното съдържание.

Традиционните подходи за маркиране на мултимедийното съдържание като интелектуална собственост на новинарските агенции не са надеждни и могат лесно да бъдат заобиколени дори от начинаещи Интернет потребители. За да се гарантира легалната употреба на мултимедийното съдържание, което е достъпно за крайни потребители в Интернет, са необходими нови подходи, които осигуряват по-надеждна защита.

6.2.2 Подобряване на защитата чрез криене на данни

Методите за криене на данни имат потенциала да преодолеят недостатъците, изброени в предходния раздел. Те могат да вградят невидима за крайния потребител информация за притежателя на авторските права директно в мултимедийното съдържание, което води до няколко ясно изразени предимства в сравнение с традиционните подходи. За разлика от използването на EXIF данни, информацията, вградена от методите за криене на данни не може да лесно да се открие, промени или изтрие от стандартните програми за разглеждане и обработка на изображения. В случай на промяна на файловия формат не съществуват мета-данни, запомнени във направляващото описание на файла, които могат да бъдат загубени. По този начин методите за криене на данни предлагат по-голяма надеждност и устойчивост на съхранената информация за авторските права.

За разлика от маркирането на изображението с видимо лого или текст, информацията, вградена от методите за криене на данни, не се вижда с невъоръжено око от крайния потребител. Следователно, тя не намалява артистичната или информативната стойност на мултимедията. Освен това програмното осигуряване за обработка на изображения или видео не може надеждно да филтрира или изреже вградените сигнатури. Те са разпръснати в цялото мултимедийно съдържание и често съществуват множество вградени копия на сигнатурата.

Като се вземе предвид отсъствието (до този момент) на правно регулиране на технологиите за криене на данни, техните предимства пред традиционните подходи за защита на интелектуалната собственост в Интернет са добре обосновани и мотивират възприемането им от агенциите, публикуващи мултимедийно съдържание в Интернет.

6.2.3 Откриване на нарушения на авторските права

Сертифициращата услуга за криене на данни, описана в разделите от 6.1.4 до 6.1.8, може да се използва също и за подобряване на защитата на мултимедийно съдържание, публикувано в Интернет от новинарски агенции, като спомага за своевременното откриване на нарушения на авторските права. Интегрирането на услугата със съществу-

ващата ИТ-инфраструктура на новинарска агенция може да се извърши по начина, описан в сценария за предотвратяване на опити за фишинг (раздел 6.1). Концепцията, илюстрирана на Фиг. 63, остава валидна и за този сценарии.

Фиг. 76: Откриване на нарушения на авторските права

Основна разлика в този сценарий е недостатъчната мотивация на потребителите за използването на сертифициращата услуга при посещение на новинарски Интернет портали, които не оперират с лични или финансови данни. Това налага въвеждането на допълнителен компонент - самостоятелен автоматизиран Интернет агент за търсене (англ. web crawler), чиято задача е регулярното сканиране на Интернет за случаи на нарушение на авторските права (Фиг. 76). Автоматизираният Интернет агент за търсене има за задача да сканира периодично Интернет порталите на конкурентни информационни агенции за изображения и видео файлове, които е вероятно да са били нелегално заимствани от Интернет портала на новинарската агенция (стъпка *1*). Интернет агентът предава откритите изображения и видео файлове на сертифициращата услуга за криене на данни, която ги проверява за вградени сигнатури (стъпка *2*). Възможни са два резултата от тази проверка:

1. Ако мултимедийното съдържание е било заимствано от Интернет портала на информационната агенция, то ще съдържа сигнатура, която е била вградена от сертифициращата услуга за криене на данни (вж. Фиг. 63). По време на проверката тази сигнатура ще бъде прочетена от сертифициращата услуга и информацията, която тя носи ще бъде сравнена с данните за Интернет портала, от който мултимедията е заредена. Тъй като това е конкурентен Интернет портал, сигнатурата няма да съответства на него. Интернет агентът ще бъде уведомен за несъответствието, което сигнализира за нарушение на авторските права (стъпка *3*) и новинарската агенция може да предприеме подходящи правни и съдебни мерки.

2. Ако съдържанието не е било заимствано от Интернет портала на новинарската агенция, сертифициращата услуга за криене на данни няма да открие сигнатурата на агенцията в мултимедийните файлове по време на проверката. В този случай нарушение на авторските права няма да бъде открито.

С помощта на сертифициращата услуга за криене на данни, новинарските агенции могат да проверяват по автоматизиран начин дали техните конкуренти заимстват мултимедийно съдържание без предварително одобрение. Услугата не може да предотврати самото нарушение на авторските права, но тя помага за неговото бързо откриване. Резултатите могат да се използват като доказателство при воденето на правни спорове с вече идентифицираните нарушители на авторските права.

6.3 Подобряване на законосъобразното използване на мултимедийно съдържание в Интернет-базирани общества

Предложените методи са подходящи за използване в рамките на Интернет-базирани общества – социални мрежи като Facebook и мултимедийни форуми като DevianArt, които се характеризират с формално нерегулирано разпространение и създаване на връзки (англ. linking) към мултимедийно авторско съдържание. Това поражда *два* принципни проблема:

1. Интернет потребителите трудно могат да определят правния статус на дадена мултимедийна творба и да установят контакт с нейния автор, за да получат разрешение за ползване, ако това е необходимо. При липсата на бърза и ефективна възможност за установяване на лицензна на ползване на мултимедийната творба, потребителите често допускат, че тя е свободна за ползване, тъй като вече се намира в Интернет. Ако това не е така, те поемат риска от съдебен процес, но считат вероятността това да се случи за незначителна.
2. Някои Интернет потребители целенасочено нарушават авторското право и претендират за авторство върху разпространявани в цифров вид творби, които не са създадени от тях. В случай,че реалните автори потърсят правата си, те трудно могат да докажат кой е истинският автор. Тъй като при разпространението на творбите между членовете на съответното Интернет-общество често не съществува и е нежелан надежден механизъм за централизирано наблюдение на комуникациите, доказването на авторство върху определена творба често е невъзможно.

Модулните методи за криене на данни, и по-специално приложно-специфичните модули за цифрово маркиране, имат потенциала да помогнат за разрешаването на тези два проблема.

6.3.1 Използване на методи за цифрово маркиране

За ефективното решение на гореспоменатите два проблема и за постигане на по-добро прилагане на законовите регулации е нужен нов технически подход, който би повлиял върху съществуващата Интернет култура и би довел до промяна в навиците на Интернет потребителите.Подходящо решение предлагат модулните методи за цифрово маркиране, които вграждат описващи творбата метаданни като неразделна част от съдържанието ѝ. Метаданните могат да съдържат детайлна информация за автора, както и URL-връзка към лицензa, описващ разрешените от автора начини на използване. Важно предимство на цифровото маркиране пред други подходи е невъзможността за разделяне на вградените метаданни от мултимедийната творба по време на нейното разпространение. Например, запомнянето на метаданните като част от направляващото описание на даден мултимедиен формат е неефективно в случай на смяна на формата с такъв, който има различна структура на направляващото описание. Видимата с просто око информация за авторските права може лесно да бъде отстранена при изрязване на част от изображението или филтриране. Методите за цифрово маркиране запазват информацията, ако мултимедийното съдържание остане в по-голямата си част непроменено.

Приложението на методите в рамките на Интернет-базираните общества има предимството, че предоставя на потребителите бърз и надежден начин за проверка на правния статус на мултимедийните творби. Даден заинтересован потребител или автоматизиран скенер могат да прочетат и филтрират вградената правна информация и да идентифицират и филтрират авторите на творбите и лицензионните условия. По този начин може да се реализира надеждно търсене на творби с даден правен статус или цена за единичен лиценз. Ако има нужда, може да се установи бърз контакт с притежателя на авторски права и бързо да се договори гъвкав и законосъобразен лиценз, постигащ компромис между нуждите на потребителя и автора.

Също така се предоставя ефективен механизъм за вътрешна саморегулация в рамките на дадено Интернет-базирано общество. Посредством подходящ потребителски интерфейс и разширение за браузър или автоматизиран скенер, вградените метаданни, съдържащи детайли за автора и лиценза на дадена творба, могат да бъдат автоматично прочетени и анализирани от браузъра или скенера и показани на потребителя. Ако има нарушение на авторското право, то ще се открие бързо и в резултат репутацията на извършителя в Интернет-базираното общество ще бъде сериозно накърнена.

Разглеждайки описаните проблеми от икономическа гледна точка, е важно да се спомене, че ако потребителят е поставен пред избор, той предпочита алтернативата, която максимизира неговата печалба. Под печалба се разбират не само материални облаги, но и нематериални понятия като време, репутация, степен на влияние, власт и др. Цифровото маркиране повишава „цената” на нарушението на авторското право посредством накърнена репутация и бързо публично разкриване на нарушенията. Еднов-

ременно с това тази технология намалява усилията и времето, отделени от потребителите (понижава „цената") за легализирането на употребата на чуждо мултимедийно съдържание съгласно лицензионните условия, определени от носителите на авторските права. Крайната цел е да се понижи печалбата от нарушаването на авторското право, така че то да стане неефективно за средностатистическия потребител в сравнение със законосъобразния път на получаване на съответния лиценз. Постигането на тази цел разчита на комбинацията от прилагането на методи за цифрово маркиране и ефекта на саморегулация в Интернет-базираните общества.

6.3.2 Приложни сценарии

Нека опишем накратко два важни сценария, илюстриращи значението на горепосочените фактори и подчертаващи ползата от цифровото маркиране в децентрализирани Интернет-базирани общества. Правилата, които регулират разпространението и използването на интелектуална собственост, се определят от притежателите на авторските права. Най-често се преследват две главни цели: постигане на популярност и получаване на материална компенсация, съответстваща на (икономическата) полза, която други потребители извличат от творбата. С цел да се разграничат различните типове потребители и да се постигне по-лесно популяризиране на творбите им, много автори избират да предоставят своите произведения с безплатен лиценз за ползване за некомерсиални цели - често разновидност на Creative Commons [180]. Комерсиалната употреба се заплаща според различни ценови модели. Цифровото маркиране може да улесни процеса на плащане в децентрализирани Интернет-базирани общества, където творбите не са въведени в централна система, съдържаща информация за съответния ценови модел, а се разпространяват свободно в Интернет-базираното общество.

Сценарий за заплащането на мултимедийни творби (микроплащане, англ. micropayment) е показан на Фиг. 77:

1. В първата стъпка авторът (притежателят на авторските права) използва компютърна програма или Интернет-базирана услуга за цифрово маркиране, за да вгради информация, идентифицираща лиценза за ползване и ценовия модел, в мултимедийната творба. Процесът може да се автоматизира, използвайки решенията, описани в раздел 6.1.7, така че във всяка мултимедийна творба, която се споделя от даден потребител с Интернет-обществото, се вграждат автоматично необходимите данни в момента на споделяне.
2. След това творбата се разпространява в Интернет-базираното общество с лицензионното ограничение, че свободната употреба и разпространение са разрешени само с некомерсиална цел.

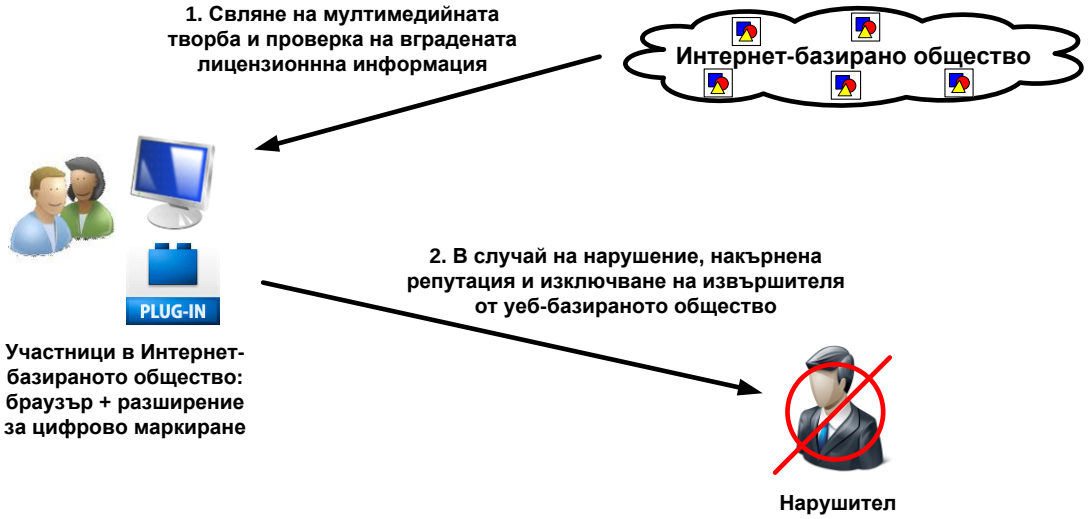
3. Ако даден бизнес потребител иска да използва творбата с комерсиална цел, той/тя може да я свали посредством нормален Интернет браузър и да провери вградения лиценз и ценови модел чрез разширението за цифрово маркиране.
4. Според ценовия модел и нуждите на потребителя, необходимото плащане може да се извърши, като браузърът на потребителя се препрати към подходяща услуга за обработка на плащания, напр. PayPal или Moneybookers. След извършване на плащането бизнес потребителят получава персонализирано копие на творбата. То съдържа информация, вградена от компютърната програма или Интернет-базираната услуга за цифрово маркиране, описваща купувача и разрешената комерсиална употреба, която може да бъде ограничена по време или начин на използване.

Фиг. 77: Микроплащане на мултимедийни творби

Прилагането на цифрово маркиране елиминира нуждата от централен каталог, свързващ мултимедийните творби с техните лицензионни споразумения, ценови модели и начин на заплащане. Съхраняването на тези данни като неделима част от защитената творба представлява значително предимство в Интернет-базираните общества и социални мрежи, които не се регулират централизирано. Илюстрираният сценарий описва първата и най-важна роля на цифровото маркиране в обществото – да подобри и улесни легалната употреба на интелектуалната собственост. Като показва на всеки потребител, инсталирал разширението за цифрово маркиране в браузъра си, детайлна информация за лиценза и начина на плащане, технологията дава на участниците в Интернет-базираното общество лесна и бърза възможност да използват мултимедийните творби по начина, предвиден от техния автор, в случай, че те желаят да направят това. Потребителите, които все още не са инсталирали разширение за цифрово маркиране в

браузъра си, ще могат да разглеждат и използват творбите по същия начин, по който са го правили преди въвеждането на технологията, което гарантира ненарушена степен на интерактивност в обществото.

Въпреки всички технологични улеснения, при липса на стриктни механизми за налагане на законосъобразно поведение, някои потребители съзнателно и целенасочено избират да нарушат авторското право - напр. да не заплатят за комерсиалната употреба на творбите, без значение, че имат необходимата информация за бързо и лесно извършване на плащането.Също така те биха могли да опитат да си присвоят свободно разпространяваната творба и да се афишират като нейни автори. Цифровото маркиране трудно може да се използва като защитен механизъм за предотвратяване на извършването на нарушения на авторското право, но то може да улесни тяхното бързо откриване от участниците в Интернет-обществото.



Фиг. 78: Откриване на нарушения на авторското право

Сценарий, описващ откриването на нарушения на авторското право, е показан на Фиг. 78. В ежедневния процес на комуникация, участниците в Интернет-обществото свалят и разглеждат мултимедийно съдържание. Ако те имат инсталирано разширение за цифрово маркиране в техния браузър, те ще видят автоматично името на автора и лицензa на съответната творба. Свободно разпространяваните копия за некомерсиална употреба се показват като такива. Те нямат индивидуален комерсиален лиценз. Копията за бизнес потребители са персонализирани и съдържат вградена информация, описваща бизнес потребителя и разрешения начин за употреба на творбата. Липсата на тази информация може лесно да се забележи от потребителите, които я разглеждат. В този случай репутацията на бизнес потребителя, който използва творбата за комерсиална цел без съответното разрешение от автора, ще бъде накърнена. Ако подобни нарушения продължат да бъдат извършвани, нарушителят може да бъде изключен от Интер-

нет-базираното общество изцяло. Подобни последствия заплашват и потребителите, които се опитват неправомерно да си присвоят авторските права върху свободно разпространявани творби на други автори.

Този сценарий описва как Интернет-базираните общества, снабдени с подходящите технологии, могат да се саморегулират. Цената на нарушенията на авторските права, измерени чрез накърнената репутация и евентуалното изключване от Интернет-базираното общество става достатъчно висока, така че ефективно да предотврати такива опити. Саморегулацията в Интернет-базираните общества и социалните мрежи би могла да бъде високо ефективна, подобно на малките населени места в реалния свят. Ако информация за неморално поведение - в този случай неспазването на авторското право - стане публично достояние, то тогава е неизгодно за всеки, които иска да остане част от обществото, да извършва нарушение, дори и при липса на формални наказателни механизми. Цифровото маркиране предоставя техническите средства за публично оповестяване на нарушения на авторското право в децентрализирани Интернет-базирани общества. Тази роля на технологията допълва първия сценарий, дискутиращ улесняването на легалната употреба. По този начин цифровото маркиране подобрява придържането към правилата на поведение в Интернет-базираните общества чрез едновременното намаляване на усилията за добро поведение и увеличаване на значимостта на последствията от нарушения.

6.4 Заключение

Описаните три сценария на приложение показват, че методите за криене на данни могат да се използват за предотвратяване на фишинг, защита на мултимедийната интелектуална собственост, както и като средство за определяне на лицензa и заплащането при използването на авторски творби с търговска цел. Това, което обединява тези доста различни приложни сценарии, е използването на самото мултимедийно съдържание за подобряването на различни аспекти на сигурността и защитата на участниците. Тъй като вградените сигнатури могат да реализират криптографски подходи на защита, то методите за криене на данни не заменят, а по-скоро допълват и дават нова свобода за ползване на традиционната криптография.

Сертифициращата услуга за криене на данни, която се използва в приложните сценарии, е достъпна чрез собствен сървър с инсталиран Майкрософт IIS сървър и платформа .NET. Веднъж инсталирана и конфигурирана, тя може да се използва по сравнително прост и гъвкав начин чрез стандартни комуникационни протоколи като SOAP и HTTP. Благодарение на модулността на разработените методи за криене на данни и гъвкавия интерфейс за мрежови услуги, тя може лесно да се адаптира към променящи се изисквания, което е важно за приложенията ѝ в Интернет. Във всички сценарии се цели високо ниво на автоматизация и съвместимост с потребители, които все още не използват услугата по различни причини.

В резултат от разгледаните в тази глава решения за подобряване на сигурността на Интернет-базирани сценарии са получени следните приложни приноси:

1. Изследван е начинът на приложение и са определени предимствата на модулния подход и модулните методи за криене на данни в три сценария: сценарий за предотвратяване на фишинг на банкови портали, сценарий за защита на мултимедийната интелектуална собственост на информационни агенции и сценарий за подобряване на законосъобразното използване на мултимедийно съдържание в Интернет-базирани общества.
2. Разработена е програмна реализация на сертифицираща услуга за криене на данни, която може да се интегрира в различни Интернет-базирани приложни сценарии и предоставя достъп до функционалността на модулните методи за криене на данни.

Приложенията на модулните методи и реализацията на сертифицираща услуга за криене на данни се дискутират в издадената от IEEE авторска публикация [111] и авторската публикация [115].

Глава 7

Заключение – основни резултати

Тази глава завършва дисертацията и обобщава основните акценти на научноизследователската работа. Изреждат се авторските публикации, получените резултати и приноси, направени в областта и се дават някои подходящи насоки за бъдеща работа.

7.1 Списък с авторски публикации по дисертацията

[1] S. Ilchev, "Accurate Data Embedding in JPEG Images for Image Authentication," *Comptes rendus de l'Acad'emie bulgare des Sciences*, vol. 66, no. 9, pp. 1247-1254, Sep. 2013, ISSN 1310-1331.

[2] S. Ilchev and R. Andreev, "Steganalysis Evaluation of Modular Data Hiding Methods," in *Proceedings of the International Conference "Automatics and Informatics'12"*, Sofia, Bulgaria, 2012, pp. 290-293, CD ISSN 1313-1869.

[3] S. Ilchev and Z. Ilcheva, "Modular Data Hiding as an Alternative of Classic Data Hiding for Web-based Applications," *Information Technologies and Control*, no. 1/2012, pp. 9-15, Jan. 2012, ISSN 1312-2622.

[4] S. Ilchev, "Modular Digital Watermarking Method for Image Tampering Detection," in *Proceedings of the International Conference "Automatics and Informatics'11"*, Sofia, Bulgaria, 2011, pp. B221-B224, ISSN 1313-1850, CD ISSN 1313-1869.

[5] S. Ilchev and Z. Ilcheva, "Protection of Intellectual Property in Web Communities by Modular Digital Watermarking," in *IEEE Signature Conference on Computers, Software and Applications (COMPSAC 2011), 35th IEEE Annual Computer Software and Applications Conference Workshops*, Munich, Germany, 2011, pp. 374-379, E-ISBN 978-0-7695-4459-5, Print ISBN 978-1-4577-0980-7, DOI 10.1109/COMPSACW.2011.69, INSPEC 12288790.

[6] S. Ilchev and Z. Ilcheva, "Modular Data Hiding Approach For Web Based Applications," in *Proceedings of the International Conference "Automatics and Informatics'10"*, Sofia, Bulgaria, 2010, pp. I253-I256, ISSN 1313-1850. **Best presented paper award.**

[7] S. Ilchev and Z. Ilcheva, "Modular data hiding for digital image authentication," in *Proceedings of the IADIS European Conference on Data Mining*, Freiburg, Germany, 2010, pp. 122-127, ISBN 978-972-8939-23-6.

Една публикация е публикувана в списание с импакт фактор „*Доклади на БАН*“, една публикация е публикувана в специализирано списание „*Information Technologies and Control*“, две публикации са докладвани на специализирани международни конференции на IEEE и IADIS. Три публикации, една от които е отличена с награда, са докладвани на специализирани национални научни конференции с международно участие. Резултати от дисертацията са представени и на научен семинар на ИИКТ..

7.2 Участие в проекти

Докторантът е взел участие в следните проекти:

1. „Concerto Premium”, финансиран като тендър по инициативата „Кончерто“ на Европейската комисия, Седма Рамкова Програма, Научна Тема: „Енергия“, договор № eu:15620-2011, референтен № ENER/C2/59-1/2010, координатор на проекта Steinbeis-Europa-Zentrum.
2. „Създаване на офис за технологичен трансфер „Информационни и комуникационни технологии за енергийна ефективност (ИКТЕЕ)“, финансиран по оперативна програма „Развитие на конкурентоспособността на българската икономика“ 2007-2013, договор № BG161PO003-1.2.02-0001-C0001, бенефициент: ИИКТ – БАН.

7.3 Постигнати резултати

Модулният подход за криене на данни в мултимедия за целите на мрежови приложения, разработен в дисертацията, е иновативна концепция по отношение на научноизследователската и приложната област на криенето на данни в мултимедия. Подходът позволява създаването на разширяеми и адаптируеми методи за криене на данни, които са насочени специално към приложения в глобалната мрежа. Поради възможността за повторно използване на модулите на методите и лесното напасване към промени в изискванията на сценария, модулните методи за криене на данни могат да бъдат създавани бързо и на достъпна цена. Комуникационният интерфейс за мрежови услуги и характеристиките на методите, разработени специално за употреба в Интернет-базирани сценарии, помагат за внедряването на технологията за криене на данни в мултимедия в сценарии от непосредствено значение за Интернет потребителите.

7.3.1 Модулност и адаптивност

Определянето и реализирането на подходящо разделяне на методите за криене на данни на сравнително самостоятелни модули е една от най-важните иновации в дисертацията. Традиционните методи за криене на данни са монолитни решения, разработени във връзка с конкретен проблем и конкретно приложение, което е от значение за да-

ден краен потребител – най-често голяма корпорация или правителствена агенция. За разлика от тях модулните методи за криене на данни, разработени в дисертацията, са изградени от самостоятелни базови и приложно-специфични модули. След като са били веднъж разработени, тези модули могат да бъдат използвани повторно в различни комбинации за съставянето на нови методи за криене на данни, които имат точните характеристики, необходими на крайните потребители в даден приложен сценарий. По този начин се намалява значително времето, финансовите ресурси и експертните познания, необходими за разработването на нови методи за криене на данни, когато сценарият или потребителската целева група бъдат променени.

Трябва да се отбележи, че преходът от монолитна към модулна структура на методите за криене на данни не е лесна задача. Монолитните методи за криене на данни могат да бъдат отлично оптимизирани към изискванията на потребителите в конкретния сценарий, за който те са разработени. Модулните методи се състоят от градивни модули, които комуникират помежду си посредством точно дефинирани интерфейси. Това позволява оптимизиране на функционалността в рамките на отделните модули, но намалява съществено потенциала за постигане на добра цялостна оптимизация на конкретен метод, съставен от конкретна комбинация от модули. Подходът, представен в тази дисертация, дефинира разделянето на методите на модули по начин, който дава същевременно възможност и оптимизация на съставните методи. Както показва оценката, направена в Глава 5, резултатите са много добри.

7.3.2 Подобрена устойчивост срещу JPEG трансформации и стеганализ

Устойчивостта на вградените данни е особено важно свойство на методите за криене на данни и е от важно значение за приложението им в Интернет-базирани сценарии.

В дисертацията е разработен нов метод за постигане на устойчивост срещу JPEG компресия, декомпресия и рекомпресия. Този метод е реализиран под формата на отделен базов модул (*DCTHiderEngine*) и може да се използва в комбинация с различни приложно-специфични модули като *StegoHider* и *WatermarkHider*.

Основният фокус при разработката на модула е идеалното (с точност до бит) възстановяване на данните, вградени в JPEG изображения, дори след прилагането на JPEG-базирани трансформации от широко разпространени програми за обработка на изображения (*GIMP, Photoshop, IrfanView* и др.). По този начин се гарантира максимална надеждност на вградените данни. Крайните потребители могат да изпълняват обичайни JPEG-базирани трансформации върху изображенията, без това да води до загуба на битове от скритата информация.

Приложно-специфичните съставни блокове на модулните методи, вграждащи данни в JPEG изображения, могат да използват вече реализираната функционалност на модула, без да се налага да се съобразяват с ограничения, особености и детайли от ниско ниво, налагани от JPEG формата. Устойчивостта срещу JPEG трансформации е реа-

лизирана изцяло в базовия модул, като осигурява надеждна база за вграждане на данни за приложно-специфичните модули.

Устойчивостта срещу откриването на вградените данни от разпространени методи за стеганализ също се числи към значимите приноси на разработваните в дисертацията модулни методи. Съществуващите методи за стеганализ не са ефективни, а анализът на представените принципни подходи за стеганализ води до извода, че създаването на нов метод за стеганализ, откриващ данните, вградени от модулните методи, не е лека задача, което води до отлична степен на сигурност на модулните методи срещу автоматизиран анализ на изображенията.

7.3.3 Услуга за сертифициране

Друг важен принос на дисертацията е разработката на услугата за сертифициране, представена в Глава 6, която е създадена, за да подобри сигурността в различни видове Интернет-базирани сценарии. Тя цели лесна интеграция на модулните методи за криене на данни с традиционната ИТ-инфраструктура, която се използва в глобалната мрежа: Интернет сървъри, приложни сървъри, Интернет браузъри, автоматизирани агенти за търсене и др. Посредством използването на интерфейс за мрежови услуги, услугата за сертифициране позволява на крайни клиенти и клиентски приложения да вграждат сигнатури по техен избор в мултимедийното съдържание преди публикуване на мултимедийните файлове в Интернет. Тези вградени сигнатури могат да бъдат проверявани от услугата за сертифициране и използвани за различни цели - напр. предотвратяване на опити за фишинг или откриване на нарушения на авторските права.

Разработката на традиционните методи за криене на данни най-често се фокусира върху един конкретен сценарий на приложение, който е от интерес за компания от частния сектор или правителствена агенция. Разработката е скъпа и получените методи трудно могат да бъдат адаптирани към нови условия. За разлика от традиционните методи, услугата за сертифициране използва модулните методи, като нейна изрична цел е да предостави гъвкава връзка между функционалността на методите за криене на данни и нуждите на реалните Интернет-базирани сценарии. Услугата за сертифициране се предоставя на разположение на крайните клиенти чрез стандартизирани технологии и протоколи за комуникация, което улеснява нейната употреба.

По този начин модулните методи за криене на данни могат да предоставят своята функционалност под формата на мрежова услуга (англ. Software-as-a-Service, SaaS). Така крайните клиенти: от малки и средни фирми до големи корпорации, които се интересуват от услугата, могат бързо да я интегрират в своите системи, като оптимизират необходимото за това време и разходи.

7.4 Основни приноси

Този раздел представя основните приноси на дисертацията.

7.4.1 Научно-приложни приноси

1. Разработен е нов модулен подход за криене на данни и е дефиниран наборът от методи за комуникация между модулите.
2. Създаден е собствен метод за постигане на устойчивост срещу JPEG компресия, декомпресия и рекомпресия, който работи с произволни изображения и данни за вграждане, като гарантира възстановяването на вградените данни с точност до бит. Методът е реализиран в контекстно-независим базов модул.
3. Проектиран е контекстно-независим базов модул за работа с изображения с беззагубна компресия, използващ модулация на индекса на квантизация. Той работи с произволни изображения и данни за вграждане и гарантира възстановяването на скритата информация с точност до бит.
4. Създаден е стеганографски метод за криене на данни, включващ стеганографски приложно-специфичен модул, направляващо описание на вгражданите файлове, кодове за корекция на грешки и разбъркване на данните.
5. Създаден е метод за цифрово маркиране, включващ приложно-специфичен модул за цифрово маркиране, направляващо описание на вгражданите водни знаци и възможности за откриване на променени блокове в изображението и възстановяване на вградения воден знак в случай на промени.
6. Създадена е многослойна архитектура на прототипна програмна система за целите на проверката, оценката и стеганализа на модулния подход за криене на данни и разработените модулни методи.
7. Изследвана и оценена е ефективността на някои популярни и разпространени алгоритми за стеганализ по отношение на модулните методи за криене на данни. Освен това е изследвана и оценена ефективността на два принципни подхода за стеганализ: подход, базиран на изследване на корелациите в изображението, и подход, базиран на прилагане на филтри.

7.4.2 Приложни приноси

1. Реализирана е прототипна програмна система посредством програмните езици VB.NET, C# и C/C++. Тя се състои от следните слоеве: слой на данните, слой на базовите модули, слой на приложно-специфичните модули и интерфейсен слой.
2. Проектирани и реализирани са инструменти за групова обработка и анализ на изображенията, за прилагане на филтри и за построяване на хистограми в различни цветови пространства.

3. Оценена е реализацията на модулните методи чрез изследване на устойчивостта на вградените данни срещу JPEG трансформации и проверка на качеството на изображенията.
4. Изследван е начинът на приложение и са определени предимствата на модулния подход и модулните методи за криене на данни в три сценария: сценарий за предотвратяване на фишинг на банкови портали, сценарий за защита на мултимедийната интелектуална собственост на информационни агенции и сценарий за подобряване на законосъобразното използване на мултимедийно съдържание в Интернет-базирани общества.
5. Разработена е програмна реализация на сертифицираща услуга за криене на данни, която може да се интегрира в различни Интернет-базирани приложни сценарии и предоставя достъп до функционалността на модулните методи за криене на данни.

7.5 Насоки за бъдеща работа

Основните предимства на модулния подход за криене на данни се състоят във възможността за адаптиране на модулните методи към:

- Променящи се потребителски изисквания;
- Нови мултимедийни формати;
- Нови приложни сценарии;
- Нови криптографски подходи, методи и приложения;
- Различни програмни системи под формата на мрежови услуги.

За да се максимизира потенциала за адаптиране на модулните методи към нови условия е необходима библиотека от предварително създадени стандартни базови и приложно-специфични модули. От особено значение са базовите модули, които предоставят устойчивост срещу различни видове компресиращи формати, които са свързани със загуба на информация (напр. форматът GIF, ограничен до палитра от 256 цвята), или устойчивост срещу някои основни видове трансформации като отрязване на част от мултимедийното съдържание, скалиране и др. Други модули биха могли да предоставят по-сложна функционалност, свързана със стеганографията и цифровото маркиране като например способността да се възстановят променени области от мултимедийния файл. Създаването на достатъчно голям набор от модули, които предлагат поддръжка на разпространени и често използвани графични формати, трансформации, криптографски методи и приложения и др. е важна предпоставка за успешното практическо приложение на модулните методи за криене на данни. Допълнителна част от насоките за бъдеща работа е създаването на модулна концепция за стеганализ, която е огледално копие на модулния подход, разработен в дисертацията. Тя ще позволи разделянето на

методите за стеганализ на модули, които могат да се разработват паралелно с модулите на методите за криене на данни.

Друга важна предпоставка за практическия успех на методите е свързана със създаването на единен набор от добре обмислени стандартизирани интерфейси за мрежови услуги, които гарантират удобно внедряване на услугата за сертифициране в различни видове Интернет-базирани сценарии. Разгледаният първи прототип на услугата предоставя на разположение няколко важни генерализирани мрежови метода за вграждане и проверка на мултимедийни сигнатури. Подобна функционалност трябва да се дефинира и за други случаи на приложение на криенето на данни в мултимедия (вж. раздел 1.2). Така създаденият интерфейс за мрежови услуги трябва да се организира според добре обмислена обща схема, която може да се разшири с нови методи и която на по-късен етап би могла да достигне до международно стандартизиране.

Мултимедийното съдържание вече е основна съставна част на Интернет. Методите за криене на данни предоставят ефективен начин за защита на това съдържание. В близко бъдеще те ще станат неотменима част от механизмите за сигурност, които осигуряват безопасността на потребителите.

Благодарности

На първо място бих искал да благодаря на моя научен ръководител, доц. д-р Румен Андреев за неговите ценни напътствия и споделен опит, които подобриха в значителна степен качеството на моята научна работа и нейното писмено изложение под формата на тази дисертация.

Бих искал да благодаря също така на всички колеги от ИИКТ-БАН за подкрепата и стимулиращата творческа атмосфера през времето на моята докторантура.

Декларация за оригиналност на резултатите

Декларирам, че настоящата дисертация съдържа оригинални резултати, получени при проведени от мен научни изследвания с подкрепата и съдействието на научния ми ръководител. Резултатите, които са получени, описани и/или публикувани от други учени, са надлежно и подробно цитирани в библиографията.

Настоящата дисертация не е прилагана за придобиване на научна степен в друго висше училище, университет или научен институт.

Подпис:

Библиография

[1] Herodotus, *Histories, Book 5*., 440 B.C.

[2] J. Peterson. (1997) Steganographia (Secret Writing), by Johannes Trithemius. [Online]. URL: http://www.esotericarchives.com/tritheim/stegano.htm (accessed August 9, 2012).

[3] Deutsche Bundesbank. Bundesbank - Currency - Euro-Banknotes - Security features. [Online]. URL: http://www.bundesbank.de/bargeld/bargeld_banknoten_sicherheitsmerkmale.en.php (accessed May 14, 2011).

[4] Science Kids. Create invisible ink with lemon juice. [Online]. URL: http://www.sciencekids.co.nz/experiments/invisibleink.html (accessed May 18, 2011).

[5] H. Trimm, *Forensics the easy way*, 1st ed.: Barron's educational series, 2005.

[6] I. J. Cox, M. Miller, J. Bloom, J. Fridrich, and T. Kalker, *Digital Watermarking and Steganography*, 2nd ed.: Morgan Kaufmann Publishers, 2008.

[7] E. Lin and J. Delp, "A Review of Data Hiding in Digital Images," in *Proceedings of the Image Processing, Image Quality, Image Capture Systems Conference (PICS '99)*, Savannah, Georgia, 1999, pp. 274-278.

[8] K. Curran and K. Bailey, "An Evaluation of Image Based Steganography Methods," *International Journal of Digital Evidence*, vol. 2, no. 2, 2003.

[9] E. Cole, *Hiding in Plain Sight: Steganography and the Art of Covert Communication*, 1st ed.: John Wiley & Sons, 2003.

[10] Interagency Working Group (IWG) On Cyber Security and Information Assurance (CSIA). (2006) Federal Plan for Cyber Security and Information Assurance Research and Development. [Online]. URL: http://www.nitrd.gov/pubs/csia/csia_federal_plan.pdf (accessed May 18, 2011).

[11] D. Feng, W. Siu, and H. Zhang, *Multimedia Information Retrieval and Management*, 1st ed.: Springer, 2003.

[12] Y. Lim, C. Xu, and D. Feng, "Web based image authentication using invisible Fragile watermark," in *ACM International Conference Proceeding Series, Proceedings of the Pan-Sydney area workshop on Visual information processing*, vol. 11, 2001, pp. 31-34.

[13] C. Rey and J Dugelay, "A Survey of Watermarking Algorithms for Image Authentication," *EURASIP Journal on Applied Signal Processing*, vol. 6, pp. 613-621, 2002.

[14] C. Lu, *Multimedia Security: Steganography and Digital Watermarking Techniques for Protection of Intellectual Property*, 1st ed.: Idea Group Publishing, 2005.

[15] F. Peticolas, R. Anderson, and M. Kuhn, "Information Hiding - A Survey," *Proceedings of the IEEE, special issue on protection of multimedia content*, vol. 87, no. 7, pp. 1062 - 1078, July 1999.

[16] R. J. Anderson and F. Petitcolas, "On the limits of steganography," *IEEE Journal on*

Selected Areas in Communications, vol. 16, no. 4, pp. 474-481, 1998.

[17] Y. Shi and H. Sun, *Image and Video Compression for Multimedia Engineering*, 1st ed.: CRC Press, 2000.

[18] J. Mitchell, W. Pennebaker, C. Fogg, and D. LeGall, *MPEG Video Compression Standard*, 1st ed.: Kluwer Academic Publishers, 2002.

[19] A. Uhl and A. Pommer, *Image and Video Encryption*, 1st ed.: Springer, 2005.

[20] W. Stallings, *Cryptography and Network Security Principles and Practices*, 4th ed.: Prentice Hall, 2005.

[21] W. Mao, *Modern Cryptography: Theory and Practice*, 1st ed.: Prentice Hall, 2003.

[22] W. Zeng, H. Yu, and C. Lin, *Multimedia security technologies for digital rights management*, 1st ed.: Elsevier, 2006.

[23] M. Peinado, F. Petitcolas, and D. Kirovski, "Digital rights management for digital cinema," *Multimedia Systems*, vol. 9, no. 3, September 2003.

[24] Electronic Privacy Information Center. Cryptography Policy. [Online]. URL: http://www.epic.org/crypto/ (accessed May 18, 2011).

[25] World Wide Web Consortium. About W3C: Goals. [Online]. URL: http://www.w3.org/Consortium/mission (accessed March 24, 2011).

[26] Miniwatts Marketing Group. Internet Usage Statistics. [Online]. URL: http://www.internetworldstats.com/stats.htm (accessed May 9, 2011).

[27] T. O'Reilly. (2005) What is Web 2.0. [Online]. URL: http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html?page=1 (accessed May 9, 2011).

[28] T. O'Reilly. (2006) Web 2.0 Compact Definition: Trying Again. [Online]. URL: http://radar.oreilly.com/2006/12/web-20-compact-definition-tryi.html (accessed May 9, 2011).

[29] T. O'Reilly. (2006) Harnessing Collective Intelligence. [Online]. URL: http://radar.oreilly.com/2006/11/harnessing-collective-intellig.html (accessed May 9, 2011).

[30] Y. Peng and Q. Wu, "Secure Communication and Access Control for Web Services Container," in *Fifth International Conference on Grid and Cooperative Computing (GCC)*, 2006, pp. 412-415.

[31] S. Zarandioon and Y., Ganapathy, V. Danfeng, "OMOS: A Framework for Secure Communication in Mashup Applications," in *Computer Security Applications Conference*, 2008, pp. 355-364.

[32] M. Farley, "Web 2.0, wikis, and the IP Community," *Journal of Intellectual Property Law & Practice*, vol. 2, no. 4, 2007.

[33] R. Gil, R. Tous, and J. Delgado, "Managing intellectual property rights in the WWW: patterns and semantics," in *First International Conference on Automated Production of Cross Media Content for Multi-Channel Distribution*, 2005.

[34] J. Garofalakis, P. Kappos, S. Sirmakessis, and G. Tzimas, "Digital Data Processing For Intellectual Property Rights Preservation Over World Wide Web," in *13th International Conference on Digital Signal Processing Proceedings*, vol. 2, 1997, pp. 833-836.

[35] M. Backes and C. Cachin, "Public-key steganography with active attacks," in *2nd*

Theory of Cryptography Conference (TCC), vol. 3378 of Lecture Notes in Computer Science, 2005, pp. 210-226.

[36] A. Westfeld, "Steganalysis in the Presence of Weak Cryptography and Encoding," in *Digital Watermarking. 5th International Workshop (IWDW)*, vol. LNCS 4283, Jeju Island, Korea, 2006, pp. 19-34.

[37] S. Katzenbeisser and F. Petitcolas, *Information Hiding Techniques for Steganography and Digital Watermarking*, 1st ed.: Artech House, 2000.

[38] G. Kipper, *Investigator's Guide to Steganography*, 1st ed.: Auerbach Publications, 2004.

[39] R. J. Anderson, R. M. Needham, and A. Shamshir, "The Steganographic File System," in *Second International Workshop on Information Hiding*, vol. 1525 of Lecture Notes in Computer Science, Portland, 1998, pp. 73-82.

[40] J. Fridrich and M. Goljan, "Protection of Digital Images Using Self Embedding," in *Symposium on Content Security and Data Hiding in Digital Media*, New Jersey Institute of Technology, 1999.

[41] G. Friedman, "The Trustworthy Digital Camera: Restoring Credibility to the Photographic Image," *IEEE Transactions on Consumer Electronics*, vol. 39, no. 4, pp. 905-910, 1993.

[42] S. Meyer. (2008) Midnight Sun: Edward's Version of Twilight. [Online]. URL: http://www.stepheniemeyer.com/midnightsun.html (accessed June 18, 2011).

[43] G. Booch et al., *Object-Oriented Analysis and Design with Applications*, 3rd ed.: Addison-Wesley, 2007.

[44] S. Voloshynovskiy, F. Deguillaume, O. Koval, and T. Pun, "Information-theoretic Data-hiding: Recent Achievements and Open Problems," *International Journal of Image and Graphics*, vol. 5, no. 1, pp. 1-31, 2005.

[45] E. Lin and E. Delp, "A Review of Fragile Image Watermarks," in *Proceedings of the Multimedia and Security Workshop (ACM Multimedia '99)*, 1999, pp. 25-29.

[46] R. Gonzalez and R. Woods, *Digital Image Processing*, 2nd ed.: Prentice Hall, 2002.

[47] W. B. Pennebaker and J. L. Mitchell, *JPEG Still Image Data Compression Standard*, 1st ed.: Van Rostrand Reinhold, New York, 1993.

[48] E. Hamilton. (1992, September) JPEG File Interchange Format. [Online]. URL: http://www.w3.org/Graphics/JPEG/jfif3.pdf (accessed March 16, 2011).

[49] F. Hartung and M. Kutter, "Multimedia watermarking techniques," *Proceedings of the IEEE*, vol. 87, no. 7, pp. 1079-1107, July 1999.

[50] D. Upham. JSteg. [Online]. URL: http://zooid.org/~paul/crypto/jsteg/ (accessed March 18, 2011).

[51] P. Wayner, *Disappearing Cryptography*, 2nd ed.: Morgan Kaufmann Publishers, 2002.

[52] M. Wu, Z. Zhu, and S. Jin, "A New Steganalytic Algorithm for Detecting Jsteg," *Lecture Notes in Computer Science*, vol. 3619, pp. 1073-1082, 2005.

[53] N. Provos and P. Honeyman, "Detecting Steganographic Content on the Internet," in *Internet Society Network and Distributed System Security Symposium (ISOC NDSS)*, San Diego, California, 2002.

[54] J. Zhao and E. Koch, "Towards Robust and Hidden Image Copyright Labeling," in

IEEE Workshop on Nonlinear Signal and Image Processing, Neos Marmaras, Greece, 1995.

[55] J. Zhao and E. Koch, "Embedding Robust Labels into Images for Copyright Protection," in *International Congress on Intellectual Property Rights for Specialized Information, Knowledge and New Technologies*, Vienna, Austria, 1995.

[56] J. J. K. O'Ruanaidh, W. J. Dowling, and F. M. Boland, "Watermarking digital images for copyright protection," *Vision, Image and Signal Processing, IEEE Proceedings*, vol. 143, no. 4, pp. 250-256, 1996.

[57] I. J. Cox, J. Kilian, T. Leighton, and T. Shamoon, "Secure Spread Spectrum Watermarking for Multimedia," *IEEE Transactions on Image Processing*, vol. 6, no. 12, pp. 1673-1687, 1997.

[58] M. Wu and B. Liu, "Watermarking for image authentication," in *IEEE International Conference on Image Processing*, vol. 2, Chicago, Illinois, 1998, pp. 437-441.

[59] C.-Y. Lin and S.-F. Chang, "Semi-fragile watermarking for authenticating JPEG visual content," in *SPIE International Conference on Security and Watermarking of Multimedia Contents II*, vol. 3971, San Jose, California, USA, 2000.

[60] C.-Y. Lin and S.-F. Chang, "A Robust Image Authentication Method Distinguishing JPEG Compression from Malicious Manipulation," *Ieee Transactions On Circuits And Systems Of Video Technology*, vol. 11, no. 2, 2001.

[61] Q. Sun, S. Ye, C.-J. Lin, and S.-F. Chang, "A Crypto Signature Scheme for Image Authentication over Wireless Channel," *International Journal of Image and Graphics, special issue on Image Data Hiding*, vol. 5, no. 1, 2005.

[62] N. Provos, "Defending against statistical steganalysis," in *10th USENIX Security Symposium*, 2001.

[63] N. Provos. (2008, July) OutGuess - universal Steganography. [Online]. URL: http://www.outguess.org/ (accessed May 15, 2011).

[64] A. Westfeld, "F5—A Steganographic Algorithm," in *Proceedings of the 4th International Workshop on Information Hiding, Lecture Notes In Computer Science*, vol. 2137, 2001, pp. 289-302.

[65] R. Crandall. (1998) Some Notes on Steganography. Posted on Steganography. Posted on Steganography Mailing List. [Online]. URL: http://os.inf.tu-dresden.de/westfeld/crandall.pdf (accessed May 9, 2011).

[66] J. Fridrich, M. Goljan, and D. Hogea, "Attacking the OutGuess," in *Proceedings of the ACM Workshop on Multimedia and Security*, Juan-les-Pins, France, 2002.

[67] J. Fridrich, M. Goljan, and D. Hogea, "Steganalysis of JPEG Images: Breaking the F5 Algorithm," in *5th Information Hiding Workshop*, Noordwijkerhout, The Netherlands, 2002, pp. 310-323.

[68] P. Sallee, "Model-Based Steganography," *Lecture Notes in Computer Science (LNCS)*, vol. 2939, pp. 254-260, 2004.

[69] C.-C. Chang, T.-S. Chen, and L.-Z. Chung, "A steganographic method based upon JPEG and quantization table modification," *Information Sciences - Informatics and Computer Science*, vol. 141, no. 1-2, pp. 123-138, 2002.

[70] J. Fridrich, "Image Watermarking for Tamper Detection," in *IEEE International Conference on Image Processing (ICIP)*, Chicago, 1998.

[71] J. Fridrich, "Methods for Detecting Changes in Digital Images," in *Proceedings of The 6th IEEE International Workshop on Intelligent Signal Processing and Communication Systems (ISPACS)*, Melbourne, Australia, 1998, pp. 173-177.

[72] J. Fridrich and M. Goljan, "Images with Self-Correcting Capabilities," in *IEEE International Conference on Image Processing*, Kobe, Japan, 1999.

[73] J. Fridrich, M. Goljan, and R. Du, "Invertible Authentication Watermark for JPEG Images," in *International Symposium on Information Technology (ITCC)*, Las Vegas, Nevada, 2001, pp. 223-227.

[74] J. Fridrich, M. Goljan, and R. Du, "Lossless Data Embedding - New Paradigm in Digital Watermarking," in *Special Issue on Emerging Applications of Multimedia Data Hiding*, 2002, pp. 185-196.

[75] J. Friedrich, M. Goljan, Q. Chen, and V. Pathak, "Lossless Data Embedding with File Size Preservation," in *Proceedings EI SPIE*, San Jose, CA, 2004.

[76] E. W. Weisstein. Arnold's Cat Map. [Online]. URL: http://mathworld.wolfram.com/ArnoldsCatMap.html (accessed March 2, 2012).

[77] R.-M. Zhao, H. Lian, H.-W. Pang, and B.-N. Hu, "A Watermarking Algorithm by Modifying AC Coefficies in DCT Domain," in *International Symposium on Information Science and Engieering (ISISE)*, vol. 2, Shanghai, China, 2008, pp. 159-162.

[78] X.-P. Zhang, K. Li, and X. Wang, "A Novel Look-Up Table Design Method for Data Hiding With Reduced Distortion," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 6, pp. 769-776, 2008.

[79] M. Costa, "Writing on dirty paper," *IEEE Transactions on Information Theory*, vol. 29, no. 3, pp. 439 - 441 , 1983.

[80] B. Chen and G. Wornell, "Quantization Index Modulation: A Class of Provably Good Methods for Digital Watermarking and Information Embedding," *IEEE Transactions on Information Theory*, vol. 47, no. 4, pp. 1423-1443, 2001.

[81] Q. Li and I. J. Cox, "Using Perceptual Models to Improve Fidelity and Provide Resistance to Valumetric Scaling for Quantization Index Modulation Watermarking," *IEEE Transactions on Information Forensics and Security*, vol. 2, no. 2, 2007.

[82] A. B. Watson, "DCT quantization matrices optimized for individual images," in *Human Vision, Visual Processing, and Digital Display IV*, vol. SPIE-1913, 1993, pp. 202-216.

[83] X. Sun, J. Liu, J. Sun, N. Yang, and S. Wu, "An Improved Adaptive QIM Watermark Iterative Algorithm," in *Intelligent Information Hiding and Multimedia Signal Processing (IIHMSP)*, Harbin, China, 2008, pp. 748-751.

[84] H. Izadinia, F. Sadeghi, and M. Rahmati, "A New Steganographic Method Using Quantization Index Modulation," in *International Conference on Computer and Automation Engineering (ICCAE)*, 2009, pp. 181-185.

[85] Y.-H. Yu, C.-C. Chang, and Y.-C. Hub, "Hiding secret data in images via predictive coding," *Pattern Recognition*, vol. 38, no. 5, pp. 691-705, 2005.

[86] M. Raval, "A Secure Steganographic Technique for Blind Steganalysis Resistance," in *Seventh International Conference on Advances in Pattern Recognition, 2009. ICAPR '09*, 2009, pp. 25-28.

[87] Steganos GmbH. (2011, August) Steganos Privacy Suite: Overview. [Online]. URL:

http://www.steganos.com/us/products/data-security/privacy-suite/overview/ (accessed August 20, 2011).

[88] A. Latham. (1999, August) Steganography. [Online]. URL: http://linux01.gwdg.de/~alatham/stego.html (accessed August 20, 2011).

[89] NeoByte Solutions. Invisible Secrets 4. [Online]. URL: http://www.invisiblesecrets.com/ (accessed August 20, 2011).

[90] Digimarc Corporation. Digimarc. [Online]. URL: https://www.digimarc.com/ (accessed May 9, 2011).

[91] Digimarc Corporation. (2011, May) Digimarc Search Service. [Online]. URL: https://www.digimarc.com/solutions/enterprise_tracking.asp

[92] CSG Copyright Services GmbH & Co. KG. Photopatrol.eu. [Online]. URL: http://www.photopatrol.eu/ (accessed August 20, 2011).

[93] Fraunhofer-Institut SIT. (2011, August) Fraunhofer-Institut SIT. [Online]. URL: http://www.sit.fraunhofer.de/

[94] F. Krolupper. (2008) Advanced Photo Tools. [Online]. URL: http://www.adptools.com/ (accessed August 20, 2011).

[95] F. Krolupper. (2008) Image Spider. [Online]. URL: http://www.adptools.com/signmyimage/eng/spider.html (accessed August 20, 2011).

[96] Phibit Software. (2011, August) Icemark Overview. [Online]. URL: http://www.phibit.com/icemark/

[97] Alpha Tec Ltd. (2011, August) Eikonamark. [Online]. URL: http://www.alphatecltd.com/watermarking/eikonamark/eikonamark.html

[98] Alpha Tec Ltd. Alphacrawler. [Online]. URL: http://www.alphatecltd.com/watermarking/alphacrawler.html (accessed August 25, 2011).

[99] DataMark Technologies. (2009) DataMark Technologies. [Online]. URL: http://www.datamark.com.sg/ (accessed August 20, 2011).

[100] M. Weissfeld, *The Object-Oriented Thought Process*, 3rd ed.: Addison-Weslez, 2008.

[101] T. Strutz, *Bilddatenkompression: Grundlagen, Codierung, JPEG, MPEG, Wavelets*, 2nd ed.: Vieweg, 2002.

[102] H. Levkowitz, *Color Theory and Modeling for Computer Graphics, Visualization, and Multimedia Applications*, 1st ed.: Kluwer Academic Publishers, 1997.

[103] R. Bracewell, *The Fourier Transform and its Applications*, 3rd ed.: McGraw-Hill, 2000.

[104] D. Salomon, *Data Compression, The Complete Reference*, 3rd ed.: Springer, 2004.

[105] M. Nelson and J. Gailly, *The Data Compression Book*. New York: M&T Books, 1995.

[106] T. Moon, *Error Correction Coding. Mathematical Methods and Algorithms*, 1st ed.: John Wiley & Sons, 2005.

[107] J. H. van Lint, *Introduction to Coding Theory*, 3rd ed.: Springer, 1999.

[108] D. Knuth, *The Art of Computer Programming*, 3rd ed.: Addison-Wesley, 1998, vol. 2.

[109] G. Marsaglia, "Random number generators," *Journal of Modern Applied Statistical Methods*, vol. 2, no. 1, pp. 2-13, May 2003.

[110] S. Ilchev, "Accurate Data Embedding in JPEG Images for Image Authentication," *Comptes rendus de l'Acad´emie bulgare des Sciences*, vol. 66, no. 9, pp. 1247-1254, Sep. 2013.

[111] S. Ilchev and Z. Ilcheva, "Protection of Intellectual Property in Web Communities by Modular Digital Watermarking," in *IEEE Signature Conference on Computers, Software and Applications (COMPSAC 2011), 35th IEEE Annual Computer Software and Applications Conference Workshops*, Munich, Germany, 2011, pp. 374-379.

[112] S. Ilchev and Z. Ilcheva, "Modular Data Hiding Approach For Web Based Applications," in *Proceedings of the International Conference "Automatics and Informatics'10"*, Sofia, Bulgaria, 2010, pp. I253-I256, Best presented paper award.

[113] S. Ilchev and Z. Ilcheva, "Modular data hiding for digital image authentication," in *Proceedings of the IADIS European Conference on Data Mining*, Freiburg, Germany, 2010, pp. 122-127.

[114] S. Ilchev, "Modular Digital Watermarking Method for Image Tampering Detection," in *Proceedings of the International Conference "Automatics and Informatics'11"*, Sofia, Bulgaria, 2011, pp. B221-B224.

[115] S. Ilchev and Z. Ilcheva, "Modular Data Hiding as an Alternative of Classic Data Hiding for Web-based Applications," *Information Technologies and Control*, no. 1/2012, pp. 9-15, Jan. 2012.

[116] D. Chappel, *Understanding.NET*, 2nd ed.: Addison-Wesley, 2006.

[117] J. Duffy, *Professional.NET Framework 2.0*, 1st ed.: Wrox Press, 2006.

[118] L. Hoang and T. Thuan,.*NET Framework Essentials*, 3rd ed.: O'Reilly, 2003.

[119] Novell, Inc. The Mono project. [Online]. URL: http://www.mono-project.com (accessed May 9, 2011).

[120] J. Gailly and M. Adler. (2012) zlib Home Site. [Online]. URL: http://www.zlib.net (accessed Feb. 04, 2012).

[121] libjpeg. [Online]. URL: http://libjpeg.sourceforge.net/ (accessed January 8, 2012).

[122] Independent JPEG Group. [Online]. URL: http://www.ijg.org/ (accessed January 8, 2012).

[123] I. E. Richardson, *H.264 and MPEG-4 Video Compression. Video Coding for Next-generation Multimedia.*, 1st ed.: John Wiley & Sons, 2003.

[124] W3C. (2013, January) SOAP Specifications. [Online]. URL: http://www.w3.org/TR/soap/

[125] Network Workiing Group. (2013, Jan.) RFC2616 - Hypertext Transfer Protocol -- HTTP/1.1. [Online]. URL: http://tools.ietf.org/html/rfc2616

[126] Network Working Group. (2013, Jan.) RFC4648 - The Base16, Base32, and Base64 Data Encodings. [Online]. URL: http://tools.ietf.org/html/rfc4648

[127] K. R. Rao and P. C. Yip, *The Transform and Data Compression Handbook*, 1st ed.: CRC Press, 2001.

[128] S. Grgic, M. Mrak, and M. Grgic, "Comparison of JPEG Image Coders," in *Proceedings of the 3rd International Symposium on Video Processing and Multimedia Communications (VIPromCom-2001)*, Zadar, Croatia, 2001, pp. 79-85.

[129] N. Provos and P. Honeyman, "Hide and seek: an introduction to steganography," *IEEE*

Security & Privacy, vol. 1, no. 3, pp. 32-44, May - June 2003.

[130] C. Cachin, "An information theoretic model for steganography," in *Proc. Int.Workshop Information Hiding*, Portland, 1998, p. 306–318.

[131] J. Zoellner et al., "Modeling the security of steganographic systems," *Lecture Notes in Computer Science*, vol. LNCS 1525, Information Hiding,2nd International Workshop (D. Aucsmith, ed.), pp. 344-354, 1998.

[132] T. Mittelholzer, "An information-theoretic approach to steganography and watermarking," *Lecture Notes in Computer Science*, vol. LNCS 1768, Information Hiding, 3rd International Workshop, IH'99 (A. Pfitzmann, ed.), pp. 1-16, 1999.

[133] K. Sullivan, U. Madhow, S. Chandrasekaran, and B. Manjunath, "Steganalysis for Markov Cover Data With Applications to Images," *IEEE Transactions on Forensics and Security*, vol. 1, no. 2, pp. 275-287, June 2006.

[134] R. Blahut, *Principles and Practice of Information Theory*.: Addison-Wesley, 1987.

[135] T. Cover and J. Thomas, *Elements of Information Theory*.: Wiley, 1991.

[136] G. Cancelli, G. Doerr, I. Cox, and M. Barni, "Detection of ±1 LSB steganography based on the amplitude of histogram local extrema," in *15th IEEE International Conference on Image Processing (ICIP)*, San Diego, 2008, pp. 1288 - 1291.

[137] A. Westfeld and A. Pfitzmann, "Attacks on Steganographic Systems," in *Proc. 3rd Information Hiding Workshop*, Dresden, Germany, 1999, pp. 61 -75.

[138] J. Fridrich, M. Goljan, and R. Du, "Steganalysis of LSB Encoding in Color Images," in *Proc. IEEE International Conference Multimedia and Expo*, Piscataway, N.J., 2000, pp. CD-ROM.

[139] S. Geetha, S. Sindhu, R. Renganathan, P. Raman, and N. Kamraj, "StegoHunter: Steganalysis of LSB Embedded Images based on Stego-Sensitive Threshold Close Color Pair Signature," in *Sixth Indian Conference on Computer Vision, Graphics & Image Processing, 2008. ICVGIP '08*, 281-288, 2008.

[140] A. Ker, "Quantitative evaluation of pairs and RS steganalysis," in *Proceedings of SPIE: Security, Steganography, and Watermarking of Multimedia Contents VI*, vol. 5306, 2004, pp. 83 - 97.

[141] J. Fridrich, M. Goljan, and R. Du, "Reliable Detection of LSB Steganography in Color and Grayscale Images," in *MM&Sec '01 Proceedings of the 2001 workshop on Multimedia and security: new challenges* , New York, 2001, pp. 27 - 30.

[142] J. Fridrich, M. Goljan, and R. Du, "Distortion-free Data Embedding for Images," in *Proc. 4th Information Hiding Workshop*, Pittsburgh, Pennsylvania, 2001.

[143] J. Fridrich, M. Goljan, and R. Du, "Detecting LSB Steganography in Color and GrayScale Images," *IEEE Multimedia, DOI: 10.1109/93.959097* , vol. 8, no. 4, pp. 22 - 28, October - December 2001.

[144] S. Dumitrescu, X. Wu, Wang, and Z., "Detection of LSB steganography via sample pair analysis," *IEEE transactions on Signal Processing*, pp. 1995-2007, 2003.

[145] B. Roue, P. Bas, and J.-M. Chassery, "Improving LSB steganalysis using marginal and joint probabilistic distributions," in *ACM Multimedia Security Workshop*, Magdeburg, Germany, 2009, pp. 75-80.

[146] X. Ma and J. Lin, "Research on Efficiency Evaluation for Steganalysis," in *International Conference on Web Information Systems and Mining (WISM)*, 2009, pp.

543-547.

[147] T. Zhang and X. Ping, "A new approach to reliable detection of LSB steganography in natural images," *Elsevier Signal Processing*, vol. 83, no. 10, pp. 2085 - 2093, 2003.

[148] A. Hernandez-Chamorro, A. Espejel-Trujillo, J. Lopez-Hernandez, M. Nakano-Miyatake, and H. Perez-Meana, "A Methodology of Steganalysis for Images," in *International Conference on Electrical, Communications, and Computers (CONIELECOMP)*, Cholula, Puebla, Mexico, 2009, pp. 102 - 106.

[149] A. Chamorro and M. Miyatake, "A New Methodology of Image Steganalysis including for JPEG Steganography," in *Electronics, Robotics and Automotive Mechanics Conference (CERMA)*, 2010, pp. 434-438.

[150] J. Fridrich, M. Goljan, D. Hogea, and D. Soukal, "Quantitative Steganalysis: Estimating Secret Message Length," *ACM Multimedia Systems Journal. Special issue on Multimedia Security*, vol. 9, no. 3, pp. 288 - 302, 2003.

[151] J. Fridrich, "Feature-Based Steganalysis for JPEG Images and Its Implications for Future Design of Steganographic Schemes," *Lecture Notes in Computer Science*, vol. 3200/2005, pp. 67 - 81, 2005.

[152] Z. He, W. Ng, P. Chan, and D. Yeung, "Feature Selection For Blind Steganalysis Using Localized Generalization Error Model," in *International Conference on Machine Learning and Cybernetics (ICMLC)*, Qingdao, 2010, pp. 500 - 505.

[153] H. Cai and S. Agaia, "JPEG Steganalysis Using Color Correlation and Training on Clean Images Only," in *Proceedings of the Seventh International Conference on Machine Learning and Cybernetics*, Kunming, 2008, pp. 3710-3713.

[154] S. Hetzl. (2003, October) Steghide. [Online]. URL: http://steghide.sourceforge.net/ (accessed November 5, 2011).

[155] J. Huang and Y. Shi, "Adaptive image watermarking scheme based on visual masking," *Elect. Lett.*, vol. 34, no. 8, pp. 748-750, 1998.

[156] A. Piva, M. Barni, F. Bartolini, and V. Capellini, "DCT-based watermark recovering without restoring to the uncorrupted original image," in *Proc. of IEEE ICIP*, 1997, pp. 520-523.

[157] J. Harmsen and W. Pearlman, "Kernel Fisher Discriminant for Steganalysis of JPEG Hiding Methods," *Proceedings SPIE*, vol. 5306, pp. 13-22, 2004.

[158] B. Li, J. Huang, and Y. Shi, "Steganalysis of YASS," *IEEE Transactions on Information Forensics and Security*, vol. 4, no. 3, pp. 369-382, September 2009.

[159] Y. Shi, C. Chen, and W. Chen, "A Markov process based approach to effective attacking JPEG steganography," *Lecture Notes on Computer Science (LCNS)*, vol. 4437, pp. 249-264, 2007.

[160] C. Chen and Y. Shi, "JPEG image steganalysis utilizing both intrablock and interblock correlations," in *IEEE International Symposium on Circuits and Systems*, 2008, p. 3029–3032.

[161] Q. Liu, A. Sung, B. Ribeiro, and R. Ferriera, "Steganalysis of multiclass JPEG images based on expanded Markov features and polynomial fitting," in *Proc. 21st IJCNN*, 2008, pp. 3351-3356.

[162] Q. Liu, A. Sung, and M. Qiao, "Improved detection and evaluation for JPEG steganalysis," in *Proceedings of the 17th ACM international conference on Multimedia*

MM'09, Beijing, China, 2009, pp. 873-876.

[163] S. Cho, J. Wang, and Kuo C.-C., "BLOCK-BASED IMAGE STEGANALYSIS FOR A MULTI-CLASSIFIER," in *International Conference on Multimedia and Expo (ICME)*, 201ß, pp. 1457-1552.

[164] S. Cho, B.-H. Cha, J. Wang, and Kuo C.-C., "Block-Based Image Steganalysis: Algorithm and Performance Evaluation," in *Proc. IEEE Int'l Sym. Circuits and Systems*, Paris, 2010, p. 1679—1682.

[165] T. Holotyak, J. Fridrich, and S. Voloshynovskiy, "Blind statistical steganalysis of additive steganography using wavelet higher order," in *International Conference on Communications and Multimedia Security*, 2005, pp. 273-284.

[166] S. Lyu and H. Farid, "Steganalysis using higher-order image statistics," *IEEE Transactions on Information Forensics and Security*, vol. 1, no. 1, pp. 111-119, 2006.

[167] L. Zhang, "Wavelet Domain Steganalysis Based on Predictability Analysis and Magnitude Prediction," in *2nd International Congress on Image and Signal Processing, CISP*, 2009, pp. 1-4.

[168] N. Provos. (2008) Steganography Detection with Stegdetect. [Online]. URL: http://www.outguess.org/detection.php (accessed May 12, 2011).

[169] M. Spiegel, J. Schiller, and R. Srinivasan, *Probability and Statistics*.: McGraw-Hill, 2001.

[170] P. Hoel, *Introduction to Mathematical Statistics*, 3rd ed.: John Wiley & Sons, 1966.

[171] D. Montgomery and G. Runger, *Applied Statistics and Probability for Engineers*, 3rd ed.: John Wiley & Sons, 2003.

[172] State University of New York, Oswego. (2012, Feb.) The 68-95-99.7 Rule For Normal Distributions. [Online]. URL: http://www.oswego.edu/~srp/stats/6895997.htm (accessed Feb. 14, 2012).

[173] S. Ilchev and R. Andreev, "Steganalysis Evaluation of Modular Data Hiding Methods," in *Proceedings of the International Conference "Automatics and Informatics'12"*, Sofia, Bulgaria, 2012, pp. 290-293.

[174] E. Harold and W. Means, *XML in a Nutshell*, 3rd ed.: O'Reilly Media, 2004.

[175] Aaron Boodman. (2010) Greasespot. [Online]. URL: http://www.greasespot.net (accessed Nov. 02, 2010).

[176] K. Dsouza. (2008, August) Run Greasemonkey User Scripts in IE, Opera and Safari. [Online]. URL: http://techie-buzz.com/tips-and-tricks/greasemonkey-alternatives-for-ie-opera-and-safari.html (accessed August 26, 2011).

[177] C. Ullman and L. Dykes, *Beginning Ajax*, 1st ed.: Wrox, 2007.

[178] W3C. (2010) XMLHttpRequest / W3C Candidate Recommendation 3 August 2010. [Online]. URL: http://www.w3.org/TR/XMLHttpRequest/ (accessed November 02, 2011).

[179] Technical Standardization Committee on AV & IT Storage Systems and Equipment. (2002, April) JEITA CP-3451 - Exchangeable image file format for digital still cameras: Exif Version 2.2. [Online]. URL: http://www.kodak.com/global/plugins/acrobat/en/service/digCam/exifStandard2.pdf (accessed March 16, 2011).

[180] Creative Commons. (2002) Creative Commons. [Online]. URL: http://creativecommons.org/ (accessed Feb. 8, 2012).

Приложение 1

Табл. 1: Резултати от изследването на корелации в изображението - *Interblock1*

Изображение	Размер	Оригинал (о)	JPEG 70			JPEG 80			JPEG 90		
			μ	σ	(μ-о)/σ	μ	σ	(μ-о)/σ	μ	σ	(μ-о)/σ
airplane.png	512x512	0,41915	0,41688	0,00121	-1,87515	0,41780	0,00063	-2,15856	0,41899	0,00030	-0,54502
arctichare.png	594x400	0,43655	0,43303	0,00260	-1,35308	0,43495	0,00121	-1,32932	0,43642	0,00036	-0,36441
baboon.bmp	512x512	0,29395	0,29289	0,00071	-1,49451	0,29304	0,00036	-2,53690	0,29387	0,00020	-0,37202
baboon.png	512x512	0,29746	0,29695	0,00064	-0,79967	0,29706	0,00046	-0,86976	0,29744	0,00023	-0,08051
barbara.bmp	512x512	0,41395	0,41302	0,00067	-1,38277	0,41326	0,00034	-2,02672	0,41388	0,00015	-0,48984
barbara.gif	512x512	0,41373	0,41273	0,00098	-1,02536	0,41344	0,00036	-0,79737	0,41375	0,00022	0,11627
bates.bmp	487x727	0,39781	0,38680	0,00302	-3,64940	0,39186	0,00227	-2,62285	0,39751	0,00064	-0,45678
bird256gray.bmp	256x256	0,48091	0,45434	0,00499	-5,32218	0,46515	0,00527	-2,99165	0,47716	0,00209	-1,78884
boat.bmp	512x512	0,41899	0,41671	0,00102	-2,23516	0,41763	0,00063	-2,14633	0,41867	0,00036	-0,88505
boat256.jpg	256x256	0,43372	0,42919	0,00221	-2,05437	0,43095	0,00221	-1,25589	0,43340	0,00093	-0,34583
bridge256.bmp	256x256	0,44449	0,44395	0,00113	-0,47692	0,44508	0,00074	0,80053	0,44468	0,00029	0,64837
bridge256.jpg	256x256	0,44357	0,44328	0,00124	-0,23660	0,44341	0,00099	-0,16389	0,44364	0,00036	0,20099
Bw1.bmp	267x234	0,31597	0,34612	0,00885	3,40527	0,34605	0,00960	3,13393	0,34518	0,00811	3,59994
Bw3.bmp	296x295	0,30972	0,32437	0,00400	3,66737	0,32090	0,00465	2,40767	0,31767	0,00317	2,50882
bw4.bmp	470x221	0,42179	0,42729	0,00284	1,93561	0,42783	0,00345	1,75190	0,42849	0,00253	2,65101
Bw7.bmp	513x389	0,36944	0,37636	0,00232	2,97817	0,37648	0,00219	3,20875	0,37613	0,00190	3,50912
camera256.bmp	256x256	0,47097	0,46030	0,00386	-2,76757	0,46318	0,00346	-2,25453	0,47027	0,00175	-0,39963
cat.png	490x733	0,35929	0,35842	0,00097	-0,89786	0,35892	0,00068	-0,53295	0,35928	0,00011	-0,08414
DSC00001.JPG	768x1024	0,36608	0,36417	0,00033	-5,74258	0,36544	0,00022	-2,95986	0,36605	0,00012	-0,24754
DSC00007.JPG	1024x768	0,36301	0,36356	0,00075	0,73306	0,36391	0,00017	5,19158	0,36346	0,00014	3,34518
DSC00014.JPG	1024x768	0,51083	0,48815	0,00502	-4,51983	0,49785	0,00423	-3,06661	0,50897	0,00155	-1,19374
DSC00016.JPG	768x1024	0,47353	0,46204	0,00260	-4,41686	0,46753	0,00214	-2,79888	0,47255	0,00087	-1,13554
DSC00040.JPG	768x1024	0,38437	0,38295	0,00057	-2,50947	0,38402	0,00021	-1,69981	0,38447	0,00009	1,09107
DSC00047.JPG	1024x768	0,38876	0,37862	0,00352	-2,88241	0,38654	0,00104	-2,14192	0,38785	0,00056	-1,62698
DSC00080.JPG	1024x768	0,45952	0,45714	0,00099	-2,39738	0,45772	0,00048	-3,71884	0,45929	0,00019	-1,20798
DSC00852.JPG	1024x768	0,31659	0,31722	0,00090	0,68995	0,31621	0,00075	-0,51684	0,31687	0,00051	0,54306
DSC00861.JPG	1024x768	0,32803	0,32644	0,00068	-2,35059	0,32823	0,00034	0,56660	0,32836	0,00012	2,86238
DSC00901.JPG	768x1024	0,27888	0,27590	0,00088	-3,39424	0,27724	0,00046	-3,54615	0,27812	0,00049	-1,53405
DSC00911.JPG	768x1024	0,30954	0,30928	0,00034	-0,78419	0,30929	0,00015	-1,70894	0,30954	0,00008	0,00028
DSC01263.JPG	1024x768	0,46759	0,45439	0,00424	-3,11832	0,46034	0,00209	-3,47632	0,46744	0,00022	-0,72557
DSC01269.JPG	1024x768	0,52133	0,51913	0,00045	-4,86222	0,51998	0,00019	-7,20180	0,52114	0,00010	-1,93137
DSC01432.JPG	1024x768	0,31739	0,31680	0,00021	-2,77043	0,31724	0,00008	-1,85136	0,31746	0,00005	1,31590
DSC01443.JPG	1024x768	0,28438	0,28413	0,00018	-1,43152	0,28451	0,00017	0,71302	0,28454	0,00005	3,35403

Табл. 1: Резултати от изследването на корелации в изображението - *Interblock1*

Изображение	Размер	Оригинал (o)	JPEG 70			JPEG 80			JPEG 90		
			μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ
DSC01481.JPG	768x1024	0,33186	0,33277	0,00062	1,46551	0,33293	0,00021	5,00177	0,33336	0,00010	14,44454
DSC01534.JPG	1024x768	0,47094	0,47054	0,00025	-1,55627	0,47106	0,00018	0,72928	0,47113	0,00009	2,15284
DSC02037.JPG	1000x750	0,46666	0,46576	0,00025	-3,66404	0,46637	0,00022	-1,33196	0,46669	0,00007	0,49259
DSC02087.JPG	750x1000	0,46475	0,46393	0,00043	-1,88425	0,46494	0,00016	1,16472	0,46537	0,00012	4,99696
Flowers.bmp	1043x474	0,35336	0,35230	0,00037	-2,84430	0,35291	0,00022	-2,00621	0,35339	0,00007	0,39078
FRUIT.BMP	512x512	0,41941	0,41816	0,00114	-1,09837	0,41919	0,00061	-0,36359	0,41932	0,00021	-0,42198
Fruit.color.bmp	487x414	0,37801	0,37835	0,00087	0,39690	0,37852	0,00068	0,74968	0,37883	0,00020	4,03358
fruits.bmp	512x512	0,37766	0,37606	0,00158	-1,00722	0,37614	0,00050	-3,02789	0,37750	0,00015	-1,04442
geom2.bmp	256x256	0,33731	0,34181	0,00120	3,74423	0,34048	0,00078	4,06536	0,33850	0,00036	3,32995
girl.png	768x512	0,37899	0,37380	0,00146	-3,54422	0,37490	0,00170	-2,40315	0,37775	0,00038	-3,27073
goldhill.bmp	512x512	0,44844	0,44626	0,00150	-1,45537	0,44729	0,00053	-2,16582	0,44812	0,00039	-0,81556
goldhill256.jpg	256x256	0,46599	0,45896	0,00245	-2,87326	0,45936	0,00295	-2,24900	0,46425	0,00128	-1,36344
kai.bmp	256x256	0,38702	0,38648	0,00111	-0,48677	0,38603	0,00074	-1,34694	0,38649	0,00045	-1,18453
kaplan1.bmp	360x624	0,39327	0,38757	0,00255	-2,23223	0,39108	0,00196	-1,11932	0,39372	0,00090	0,50342
Lena256.bmp	256x256	0,41044	0,40793	0,00182	-1,37441	0,40915	0,00103	-1,25304	0,41035	0,00046	-0,18322
lena512.bmp	512x512	0,35740	0,35579	0,00072	-2,25838	0,35715	0,00077	-0,32344	0,35749	0,00024	0,36212
Leonardo.bmp	695x1040	0,33096	0,32974	0,00031	-3,87144	0,33042	0,00029	-1,83057	0,33100	0,00008	0,55249
Mouse.bmp	936x1035	0,37453	0,37680	0,00091	2,50540	0,37621	0,00064	2,63459	0,37480	0,00064	0,41759
MRI.bmp	256x256	0,46476	0,44636	0,00334	-5,51542	0,45077	0,00431	-3,24495	0,45701	0,00386	-2,01015
Niagara.bmp	399x450	0,39983	0,39513	0,00247	-1,89959	0,39846	0,00117	-1,16413	0,39890	0,00059	-1,56717
P1.bmp	345x406	0,40648	0,40956	0,00170	1,80989	0,40749	0,00192	0,52610	0,40806	0,00119	1,32390
P10.bmp	457x507	0,44350	0,42826	0,00389	-3,91655	0,43077	0,00180	-7,06795	0,43903	0,00136	-3,27983
P2.bmp	274x406	0,37435	0,37801	0,00073	5,02878	0,37672	0,00046	5,10910	0,37540	0,00019	5,48748
P3.bmp	406x433	0,40125	0,39763	0,00268	-1,35054	0,39981	0,00137	-1,05633	0,39952	0,00074	-2,35984
P4.bmp	406x433	0,35985	0,36093	0,00073	1,47184	0,36052	0,00038	1,74720	0,36062	0,00020	3,92343
P5.bmp	323x418	0,53921	0,52660	0,00457	-2,75687	0,51912	0,00698	-2,88008	0,53750	0,00186	-0,91947
P6.bmp	487x368	0,46263	0,46043	0,00120	-1,82705	0,46139	0,00100	-1,23866	0,46280	0,00060	0,28379
P7.bmp	442x554	0,51242	0,50940	0,00174	-1,73545	0,51154	0,00049	-1,78262	0,51242	0,00027	-0,00651
P8.bmp	267x347	0,33591	0,33464	0,00174	-0,73121	0,33416	0,00138	-1,26651	0,33515	0,00040	-1,91362
P9.bmp	454x323	0,37334	0,36877	0,00156	-2,92592	0,37149	0,00147	-1,25916	0,37304	0,00054	-0,55738
pentagon.tiff	1024x1024	0,30536	0,30372	0,00020	-8,04807	0,30478	0,00013	-4,53027	0,30534	0,00008	-0,21336
Peppers.bmp	512x512	0,41058	0,40841	0,00087	-2,50190	0,40968	0,00058	-1,56004	0,41049	0,00022	-0,38697
pepperscolour.png	512x512	0,40999	0,40884	0,00070	-1,63823	0,40989	0,00057	-0,17726	0,40992	0,00019	-0,33966
Picasso.bmp	1618x2064	0,36791	0,36539	0,00010	-26,00332	0,36722	0,00007	-9,68477	0,36789	0,00003	-0,44646
pool.bmp	510x383	0,42548	0,39589	0,00881	-3,35797	0,40656	0,01084	-1,74508	0,41702	0,00610	-1,38570
pool.png	510x383	0,44055	0,41337	0,00743	-3,65607	0,42324	0,00922	-1,87746	0,42854	0,00370	-3,24970
thoraxl.bmp	280x301	0,56581	0,55703	0,00349	-2,51525	0,56259	0,00194	-1,66116	0,56583	0,00065	0,02594
watch.bmp	1024x768	0,33717	0,33710	0,00076	-0,08845	0,33708	0,00024	-0,35472	0,33729	0,00011	1,16503
watch.png	1024x768	0,33691	0,33612	0,00064	-1,22351	0,33685	0,00058	-0,10246	0,33697	0,00013	0,44642

Табл. 1: Резултати от изследването на корелации в изображението - *Interblock1*

Изображение	Размер	Оригинал (o)	JPEG 70			JPEG 80			JPEG 90		
			μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ
water.png	1024x768	0,32531	0,32517	0,00023	-0,62284	0,32546	0,00026	0,58752	0,32561	0,00007	4,45647
zelda256.bmp	256x256	0,51691	0,51347	0,00159	-2,16494	0,51446	0,00138	-1,77319	0,51655	0,00048	-0,74485
zelda512.jpg	512x512	0,54007	0,53554	0,00244	-1,85789	0,53728	0,00220	-1,26793	0,53959	0,00055	-0,88358

Табл. 2: Резултати от изследването на корелации в изображението - *Interblock2*

Изображение	Размер	Оригинал (o)	JPEG 70			JPEG 80			JPEG 90		
			μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ
airplane.png	512x512	0,45344	0,44921	0,00072	-5,86946	0,45162	0,00070	-2,58467	0,45300	0,00011	-4,03056
arctichare.png	594x400	0,44027	0,43792	0,00162	-1,44760	0,43915	0,00103	-1,08740	0,44016	0,00034	-0,32132
baboon.bmp	512x512	0,32476	0,32308	0,00042	-4,00202	0,32375	0,00045	-2,22503	0,32467	0,00021	-0,42996
baboon.png	512x512	0,32680	0,32546	0,00044	-3,04373	0,32556	0,00041	-3,02055	0,32639	0,00015	-2,82655
barbara.bmp	512x512	0,43029	0,42930	0,00069	-1,43659	0,42960	0,00019	-3,56919	0,43009	0,00015	-1,34476
barbara.gif	512x512	0,43000	0,42899	0,00083	-1,21597	0,42959	0,00029	-1,38001	0,42999	0,00015	-0,00374
bates.bmp	487x727	0,49805	0,48187	0,00482	-3,35822	0,49053	0,00368	-2,04555	0,49764	0,00071	-0,58062
bird256gray.bmp	256x256	0,46481	0,43596	0,00626	-4,60757	0,44654	0,00493	-3,70136	0,46171	0,00171	-1,80965
boat.bmp	512x512	0,42900	0,42787	0,00046	-2,43248	0,42849	0,00020	-2,51663	0,42902	0,00008	0,21564
boat256.jpg	256x256	0,44462	0,44357	0,00128	-0,82165	0,44410	0,00088	-0,58717	0,44474	0,00047	0,26061
bridge256.bmp	256x256	0,47920	0,47810	0,00065	-1,70539	0,47851	0,00062	-1,11883	0,47930	0,00028	0,32300
bridge256.jpg	256x256	0,47966	0,47728	0,00067	-3,52072	0,47818	0,00052	-2,85100	0,47914	0,00037	-1,37616
Bw1.bmp	267x234	0,41226	0,43071	0,00270	6,84315	0,42639	0,00269	5,24575	0,42496	0,00289	4,38740
Bw3.bmp	296x295	0,42920	0,43184	0,00170	1,54854	0,43116	0,00122	1,61009	0,43114	0,00063	3,09925
bw4.bmp	470x221	0,53354	0,54388	0,00152	6,78991	0,54057	0,00144	4,89027	0,53899	0,00127	4,29146
Bw7.bmp	513x389	0,42476	0,42870	0,00157	2,51256	0,42927	0,00147	3,05917	0,42859	0,00151	2,53827
camera256.bmp	256x256	0,45942	0,45018	0,00233	-3,96594	0,45250	0,00192	-3,61215	0,45815	0,00069	-1,84981
cat.png	490x733	0,34827	0,34701	0,00059	-2,12542	0,34755	0,00024	-2,94591	0,34812	0,00011	-1,38199
DSC00001.JPG	768x1024	0,36631	0,36568	0,00029	-2,15177	0,36589	0,00024	-1,73436	0,36626	0,00009	-0,57818
DSC00007.JPG	1024x768	0,40186	0,39980	0,00054	-3,82218	0,40073	0,00015	-7,41677	0,40179	0,00024	-0,28045
DSC00014.JPG	1024x768	0,56552	0,54378	0,00683	-3,18242	0,55262	0,00584	-2,21063	0,56293	0,00199	-1,30097
DSC00016.JPG	768x1024	0,51554	0,50716	0,00276	-3,03923	0,51115	0,00160	-2,75346	0,51505	0,00052	-0,93787
DSC00040.JPG	768x1024	0,38997	0,38946	0,00029	-1,77048	0,38992	0,00022	-0,20390	0,39002	0,00007	0,71731
DSC00047.JPG	1024x768	0,34727	0,34321	0,00095	-4,28157	0,34519	0,00081	-2,57456	0,34629	0,00051	-1,92316
DSC00080.JPG	1024x768	0,50304	0,49999	0,00061	-4,97074	0,50136	0,00030	-5,63003	0,50270	0,00014	-2,41373
DSC00852.JPG	1024x768	0,32039	0,32171	0,00026	5,07395	0,32080	0,00020	2,09861	0,32085	0,00011	4,16554
DSC00861.JPG	1024x768	0,34755	0,34702	0,00067	-0,80221	0,34807	0,00018	2,81998	0,34788	0,00007	5,05115
DSC00901.JPG	768x1024	0,26542	0,26385	0,00047	-3,35153	0,26503	0,00020	-1,98498	0,26525	0,00012	-1,37408
DSC00911.JPG	768x1024	0,31181	0,31110	0,00032	-2,23117	0,31161	0,00008	-2,41086	0,31181	0,00008	0,04580
DSC01263.JPG	1024x768	0,52854	0,51725	0,00426	-2,64896	0,52263	0,00262	-2,25289	0,52850	0,00035	-0,11067
DSC01269.JPG	1024x768	0,54665	0,54454	0,00043	-4,89563	0,54523	0,00017	-8,12588	0,54651	0,00006	-2,05002

Табл. 2: Резултати от изследването на корелации в изображението - *Interblock2*

Изображение	Размер	Оригинал (o)	JPEG 70			JPEG 80			JPEG 90		
			μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ
DSC01432.JPG	1024x768	0,31435	0,31385	0,00015	-3,29926	0,31427	0,00009	-0,82599	0,31444	0,00005	1,79788
DSC01443.JPG	1024x768	0,28242	0,28259	0,00011	1,51421	0,28259	0,00010	1,81772	0,28259	0,00003	5,44490
DSC01481.JPG	768x1024	0,29724	0,29787	0,00036	1,76114	0,29893	0,00026	6,50204	0,29948	0,00014	15,98804
DSC01534.JPG	1024x768	0,52801	0,52775	0,00018	-1,45057	0,52808	0,00008	0,88422	0,52819	0,00012	1,56502
DSC02037.JPG	1000x750	0,47651	0,47578	0,00025	-2,90927	0,47612	0,00014	-2,83528	0,47651	0,00004	-0,09551
DSC02087.JPG	750x1000	0,51688	0,51275	0,00081	-5,12133	0,51513	0,00050	-3,47237	0,51668	0,00017	-1,13326
Flowers.bmp	1043x474	0,38002	0,37928	0,00040	-1,85377	0,37989	0,00012	-1,08509	0,38003	0,00006	0,21083
FRUIT.BMP	512x512	0,44188	0,43977	0,00087	-2,42375	0,44140	0,00052	-0,90158	0,44194	0,00010	0,62267
Fruit.color.bmp	487x414	0,38492	0,38260	0,00051	-4,53388	0,38396	0,00062	-1,54481	0,38479	0,00017	-0,78739
fruits.bmp	512x512	0,35542	0,35413	0,00112	-1,14800	0,35498	0,00036	-1,22345	0,35544	0,00022	0,07742
geom2.bmp	256x256	0,51277	0,51714	0,00035	12,48789	0,51672	0,00034	11,64673	0,51398	0,00016	7,54107
girl.png	768x512	0,53907	0,53864	0,00100	-0,42804	0,53933	0,00075	0,34979	0,53936	0,00014	2,06303
goldhill.bmp	512x512	0,46621	0,45971	0,00144	-4,52086	0,46308	0,00096	-3,27061	0,46583	0,00035	-1,09246
goldhill256.jpg	256x256	0,46270	0,45809	0,00162	-2,84350	0,45985	0,00111	-2,55468	0,46246	0,00094	-0,25740
kai.bmp	256x256	0,38553	0,38569	0,00055	0,28474	0,38604	0,00049	1,03431	0,38558	0,00021	0,21614
kaplan1.bmp	360x624	0,57631	0,57184	0,00178	-2,51485	0,57412	0,00075	-2,90415	0,57552	0,00033	-2,40228
Lena256.bmp	256x256	0,41243	0,41141	0,00101	-1,00860	0,41158	0,00053	-1,58441	0,41231	0,00024	-0,50413
lena512.bmp	512x512	0,37013	0,36865	0,00109	-1,36142	0,36936	0,00046	-1,68064	0,37002	0,00017	-0,62677
Leonardo.bmp	695x1040	0,47556	0,47179	0,00038	-9,79430	0,47337	0,00026	-8,45055	0,47478	0,00004	-20,82413
Mouse.bmp	936x1035	0,32895	0,33232	0,00019	18,02842	0,33121	0,00025	9,19057	0,32989	0,00017	5,49751
MRI.bmp	256x256	0,45698	0,45465	0,00137	-1,69927	0,45704	0,00089	0,06979	0,45757	0,00096	0,61046
Niagara.bmp	399x450	0,58170	0,57809	0,00087	-4,13283	0,58007	0,00043	-3,80521	0,58142	0,00017	-1,60440
P1.bmp	345x406	0,36424	0,36766	0,00103	3,32433	0,36508	0,00122	0,68851	0,36635	0,00039	5,36186
P10.bmp	457x507	0,41459	0,40309	0,00424	-2,71648	0,40438	0,00191	-5,35615	0,41120	0,00087	-3,89547
P2.bmp	274x406	0,40154	0,40696	0,00048	11,41118	0,40489	0,00033	10,21237	0,40287	0,00019	7,14809
P3.bmp	406x433	0,40892	0,40915	0,00070	0,31989	0,40856	0,00063	-0,57467	0,40936	0,00026	1,65072
P4.bmp	406x433	0,37913	0,37755	0,00037	-4,24774	0,37841	0,00027	-2,65621	0,37891	0,00016	-1,38334
P5.bmp	323x418	0,57015	0,55617	0,00444	-3,14923	0,54975	0,00556	-3,66980	0,56471	0,00147	-3,70794
P6.bmp	487x368	0,52292	0,51692	0,00130	-4,62827	0,51933	0,00066	-5,44981	0,52151	0,00039	-3,57554
P7.bmp	442x554	0,49842	0,49512	0,00118	-2,78811	0,49772	0,00064	-1,08158	0,49836	0,00014	-0,41456
P8.bmp	267x347	0,34971	0,34651	0,00102	-3,14480	0,34796	0,00050	-3,48774	0,34912	0,00032	-1,89307
P9.bmp	454x323	0,43922	0,43289	0,00246	-2,57840	0,43693	0,00148	-1,55321	0,43860	0,00048	-1,29577
pentagon.tiff	1024x1024	0,29359	0,29241	0,00011	-11,11737	0,29328	0,00007	-4,72574	0,29360	0,00004	0,19425
Peppers.bmp	512x512	0,45529	0,45349	0,00060	-3,01903	0,45422	0,00031	-3,41335	0,45509	0,00018	-1,12295
pepperscolour.png	512x512	0,45472	0,45462	0,00063	-0,17194	0,45514	0,00051	0,81589	0,45505	0,00008	4,07951
Picasso.bmp	1618x2064	0,42825	0,42477	0,00007	-48,74118	0,42665	0,00006	-27,16992	0,42820	0,00003	-2,06231
pool.bmp	510x383	0,40040	0,35858	0,01974	-2,11855	0,36859	0,02023	-1,57160	0,38316	0,01242	-1,38797
pool.png	510x383	0,39492	0,36168	0,02095	-1,58656	0,36544	0,01763	-1,67239	0,37651	0,01037	-1,77627

Табл. 2: Резултати от изследването на корелации в изображението - *Interblock2*

Изображение	Размер	Оригинал (о)	JPEG 70			JPEG 80			JPEG 90		
			μ	σ	(μ-о)/σ	μ	σ	(μ-о)/σ	μ	σ	(μ-о)/σ
thoraxl.bmp	280x301	0,64271	0,63722	0,00142	-3,87984	0,64001	0,00098	-2,75607	0,64258	0,00031	-0,42423
watch.bmp	1024x768	0,33474	0,33306	0,00038	-4,38712	0,33437	0,00019	-1,97114	0,33469	0,00006	-0,75964
watch.png	1024x768	0,33461	0,33330	0,00066	-1,97564	0,33425	0,00031	-1,16398	0,33448	0,00008	-1,58726
water.png	1024x768	0,32285	0,32246	0,00021	-1,92788	0,32312	0,00009	3,10871	0,32332	0,00004	11,54906
zelda256.bmp	256x256	0,47518	0,47267	0,00107	-2,33963	0,47360	0,00080	-1,96847	0,47471	0,00026	-1,76330
zelda512.jpg	512x512	0,52546	0,52071	0,00201	-2,36583	0,52300	0,00208	-1,18309	0,52514	0,00035	-0,91367

Табл. 3: Резултати от изследването на корелации в изображението - *Intrablock*

Изображение	Размер	Оригинал (о)	JPEG 70			JPEG 80			JPEG 90		
			μ	σ	(μ-о)/σ	μ	σ	(μ-о)/σ	μ	σ	(μ-о)/σ
airplane.png	512x512	0,12331	0,35443	0,00467	49,45654	0,17550	0,01071	4,87329	0,12330	0,00434	-0,00384
arctichare.png	594x400	0,23883	0,42079	0,00473	38,46163	0,23780	0,01289	-0,07996	0,23253	0,03725	-0,16911
baboon.bmp	512x512	0,12475	0,18746	0,00246	25,45142	0,10007	0,00554	-4,45847	0,12375	0,00452	-0,22151
baboon.png	512x512	0,13524	0,16206	0,00258	10,38081	0,10406	0,00337	-9,24591	0,13023	0,00460	-1,09037
barbara.bmp	512x512	0,12975	0,26198	0,00376	35,16704	0,13384	0,00322	1,26972	0,12797	0,00367	-0,48703
barbara.gif	512x512	0,40456	0,28351	0,05194	-2,33054	0,26549	0,12074	-1,15180	0,54069	0,02987	4,55802
bates.bmp	487x727	0,11737	0,58875	0,00657	71,72446	0,26647	0,00503	29,64727	0,11662	0,00270	-0,27568
bird256gray.bmp	256x256	0,18252	0,56858	0,01624	23,77874	0,26593	0,01445	5,77162	0,18273	0,00759	0,02733
boat.bmp	512x512	0,12452	0,27424	0,00286	52,29661	0,12311	0,00528	-0,26581	0,12551	0,00403	0,24707
boat256.jpg	256x256	0,13381	0,32672	0,00965	19,99826	0,15450	0,01676	1,23430	0,19410	0,01317	4,57955
bridge256.bmp	256x256	0,16970	0,18734	0,00901	1,95757	0,12360	0,01220	-3,77878	0,16903	0,00954	-0,07081
bridge256.jpg	256x256	0,00000	0,18639	0,01468	12,69931	0,17658	0,08595	2,05431	0,55992	0,02641	21,20007
Bw1.bmp	267x234	0,45801	0,33945	0,01982	-5,98316	0,56633	0,09945	1,08916	0,41919	0,20514	-0,18923
Bw3.bmp	296x295	0,46760	0,50058	0,01472	2,24121	0,50921	0,09487	0,43862	0,42968	0,18477	-0,20518
bw4.bmp	470x221	0,38877	0,39229	0,01089	0,32279	0,46495	0,08045	0,94700	0,42601	0,16902	0,22035
Bw7.bmp	513x389	0,37005	0,37701	0,00909	0,76491	0,40533	0,11173	0,31575	0,50448	0,13200	1,01839
camera256.bmp	256x256	0,15667	0,42672	0,00952	28,36667	0,24086	0,01016	8,29088	0,17704	0,00869	2,34497
cat.png	490x733	0,11569	0,16477	0,00199	24,64100	0,10157	0,00308	-4,57690	0,11353	0,00239	-0,90308
DSC00001.JPG	768x1024	0,13465	0,33821	0,00459	44,31925	0,09825	0,00201	-18,13476	0,12084	0,00771	-1,79053
DSC00007.JPG	1024x768	0,16692	0,32109	0,00330	46,70533	0,13977	0,00504	-5,38380	0,18062	0,00711	1,92750
DSC00014.JPG	1024x768	0,09214	0,48129	0,00336	115,85232	0,24519	0,00083	185,03370	0,09637	0,00368	1,14608
DSC00016.JPG	768x1024	0,13915	0,47756	0,00218	155,30096	0,24576	0,00124	86,00626	0,13159	0,00440	-1,71955
DSC00040.JPG	768x1024	0,10490	0,36566	0,00200	130,61099	0,25178	0,00105	139,55337	0,10411	0,00204	-0,38552
DSC00047.JPG	1024x768	0,11180	0,32107	0,00916	22,84656	0,12923	0,00100	17,42789	0,10740	0,00174	-2,52205
DSC00080.JPG	1024x768	0,10112	0,35716	0,00249	102,83339	0,18257	0,00047	172,23489	0,10210	0,00168	0,58348
DSC00852.JPG	1024x768	0,17350	0,20208	0,00305	9,35855	0,11604	0,00352	-16,30590	0,14011	0,00530	-6,29541
DSC00861.JPG	1024x768	0,10286	0,25702	0,00302	51,01402	0,11747	0,00214	6,83637	0,10669	0,00495	0,77386
DSC00901.JPG	768x1024	0,10736	0,18926	0,00199	41,22631	0,10484	0,00182	-1,38422	0,11998	0,00458	2,75633

Табл. 3: Резултати от изследването на корелации в изображението - *Intrablock*

Изображение	Размер	Оригинал (o)	JPEG 70			JPEG 80			JPEG 90		
			μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ
DSC00911.JPG	768x1024	0,07704	0,14602	0,00319	21,59990	0,07045	0,00048	-13,62934	0,08855	0,00351	3,28361
DSC01263.JPG	1024x768	0,11108	0,48432	0,00405	92,07022	0,20236	0,00066	138,32143	0,10941	0,00166	-1,00790
DSC01269.JPG	1024x768	0,09491	0,23076	0,00360	37,78148	0,09712	0,00105	2,09684	0,09567	0,00236	0,32116
DSC01432.JPG	1024x768	0,07141	0,11298	0,00157	26,46347	0,06996	0,00062	-2,31575	0,08975	0,00250	7,34515
DSC01443.JPG	1024x768	0,00000	0,11319	0,00137	82,76805	0,06945	0,00054	129,00764	0,09109	0,00307	29,62948
DSC01481.JPG	768x1024	0,09715	0,33052	0,00190	122,81073	0,16664	0,00081	85,88538	0,09451	0,00162	-1,62839
DSC01534.JPG	1024x768	0,11173	0,20850	0,00308	31,41963	0,07367	0,00085	-44,95155	0,11091	0,00202	-0,40736
DSC02037.JPG	1000x750	0,10633	0,32616	0,00234	93,87179	0,16626	0,00180	33,21754	0,10418	0,00248	-0,86564
DSC02087.JPG	750x1000	0,09788	0,34895	0,00224	112,13681	0,18501	0,00206	42,34050	0,09648	0,00220	-0,63702
Flowers.bmp	1043x474	0,11494	0,15276	0,00195	19,38321	0,10002	0,00249	-5,98661	0,11441	0,00321	-0,16346
FRUIT.BMP	512x512	0,12545	0,28731	0,00299	54,05470	0,11907	0,00908	-0,70283	0,12780	0,00478	0,49047
Fruit.color.bmp	487x414	0,12338	0,23697	0,00520	21,82534	0,12308	0,00387	-0,07705	0,13190	0,00574	1,48435
fruits.bmp	512x512	0,12711	0,36942	0,00332	73,00355	0,15625	0,00766	3,80202	0,12596	0,00292	-0,39452
geom2.bmp	256x256	0,19227	0,23259	0,00932	4,32671	0,14002	0,00733	-7,13070	0,18848	0,00963	-0,39437
girl.png	768x512	0,15162	0,41974	0,00663	40,44297	0,19816	0,00356	13,07609	0,13610	0,00616	-2,52069
goldhill.bmp	512x512	0,13377	0,26646	0,00611	21,71016	0,12160	0,00359	-3,38813	0,13018	0,00460	-0,77919
goldhill256.jpg	256x256	0,14493	0,27291	0,01261	10,15155	0,14316	0,01111	-0,15929	0,18497	0,00707	5,66644
kai.bmp	256x256	0,16943	0,26550	0,00770	12,47512	0,14206	0,01104	-2,47939	0,17965	0,00984	1,03852
kaplan1.bmp	360x624	0,13933	0,49223	0,00871	40,51981	0,25354	0,00400	28,57251	0,13332	0,00486	-1,23780
Lena256.bmp	256x256	0,19602	0,30546	0,00958	11,42976	0,14507	0,00964	-5,28598	0,18995	0,00846	-0,71752
lena512.bmp	512x512	0,12004	0,31057	0,00322	59,09384	0,15994	0,00574	6,95175	0,12352	0,00398	0,87322
Leonardo.bmp	695x1040	0,10592	0,17441	0,00177	38,68964	0,07816	0,00099	-28,06706	0,10363	0,00219	-1,04205
Mouse.bmp	936x1035	0,35410	0,42259	0,00162	42,28784	0,31389	0,01524	-2,63901	0,46941	0,02952	3,90639
MRI.bmp	256x256	0,26437	0,29765	0,01874	1,77604	0,29505	0,02655	1,15526	0,22822	0,05826	-0,62060
Niagara.bmp	399x450	0,15356	0,29899	0,01268	11,46480	0,19320	0,00333	11,92048	0,15312	0,00578	-0,07735
P1.bmp	345x406	0,40357	0,42716	0,00687	3,43298	0,39562	0,06602	-0,12039	0,37451	0,13974	-0,20791
P10.bmp	457x507	0,19644	0,45248	0,00531	48,26218	0,24629	0,01005	4,96196	0,18683	0,02401	-0,40029
P2.bmp	274x406	0,15776	0,16933	0,00612	1,89125	0,11208	0,00677	-6,75015	0,15733	0,01056	-0,04060
P3.bmp	406x433	0,19512	0,28516	0,00797	11,29525	0,15122	0,01636	-2,68313	0,17689	0,02331	-0,78234
P4.bmp	406x433	0,21605	0,28845	0,00432	16,74277	0,18914	0,00634	-4,24228	0,19556	0,03644	-0,56230
P5.bmp	323x418	0,39548	0,43512	0,00824	4,81053	0,40734	0,07416	0,15996	0,38415	0,14867	-0,07621
P6.bmp	487x368	0,13295	0,26953	0,00754	18,10423	0,13787	0,00597	0,82477	0,13969	0,00626	1,07704
P7.bmp	442x554	0,12674	0,31534	0,00550	34,27919	0,16852	0,00595	7,01780	0,13303	0,00409	1,53721
P8.bmp	267x347	0,15347	0,18530	0,00513	6,20472	0,12342	0,01072	-2,80401	0,15910	0,00783	0,71980
P9.bmp	454x323	0,14554	0,21828	0,00422	17,22003	0,11971	0,00783	-3,30051	0,14965	0,00464	0,88637
pentagon.tiff	1024x1024	0,08506	0,20367	0,00107	110,57072	0,08655	0,00000	N/A	0,08679	0,00131	1,31395
Peppers.bmp	512x512	0,11791	0,30207	0,00444	41,51430	0,14529	0,00992	2,76034	0,12229	0,00562	0,77745
pepperscolour.png	512x512	0,12164	0,27376	0,00402	37,87960	0,13637	0,00606	2,42873	0,12241	0,00446	0,17113

Табл. 3: Резултати от изследването на корелации в изображението - *Intrablock*

Изображение	Размер	Оригинал (o)	JPEG 70			JPEG 80			JPEG 90		
			μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ
Picasso.bmp	1618x2064	0,06783	0,20213	0,00028	487,99815	0,05225	0,00001	-1057,66020	0,06800	0,00027	0,62108
pool.bmp	510x383	0,42776	0,38072	0,00889	-5,28876	0,21340	0,02757	-7,77469	0,35292	0,03972	-1,88441
pool.png	510x383	0,47296	0,38914	0,00607	-13,81041	0,24571	0,05856	-3,88095	0,38015	0,06815	-1,36201
thoraxl.bmp	280x301	0,15779	0,26944	0,01059	10,53960	0,16608	0,01157	0,71705	0,15989	0,00769	0,27366
watch.bmp	1024x768	0,21246	0,36998	0,00224	70,38047	0,20644	0,00184	-3,26901	0,19356	0,01081	-1,74746
watch.png	1024x768	0,21269	0,36690	0,00183	84,26398	0,20915	0,00236	-1,49963	0,19564	0,01139	-1,49763
water.png	1024x768	0,09574	0,14975	0,00123	43,85674	0,08160	0,00030	-47,91129	0,09597	0,00140	0,16467
zelda256.bmp	256x256	0,18507	0,22489	0,00783	5,08277	0,14893	0,00961	-3,76174	0,17918	0,00790	-0,74654
zelda512.jpg	512x512	0,14376	0,24317	0,00412	24,12058	0,14919	0,00876	0,62032	0,15930	0,01490	1,04316

Табл. 4: Резултати от изследването на корелации в изображението - *0-Analysis*

Изображение	Размер	Оригинал (o)	JPEG 70			JPEG 80			JPEG 90		
			μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ
airplane.png	512x512	0,06995	0,20925	0,01591	8,75404	0,14852	0,00805	9,75739	0,07244	0,00203	1,22651
arctichare.png	594x400	0,25649	0,30839	0,04498	1,15408	0,25897	0,04319	0,05742	0,20065	0,03773	-1,48004
baboon.bmp	512x512	0,02124	0,07813	0,00172	33,11676	0,05129	0,00136	22,04197	0,02168	0,00109	0,40416
baboon.png	512x512	0,01868	0,06392	0,00143	31,68100	0,04245	0,00107	22,28262	0,01957	0,00090	0,99331
barbara.bmp	512x512	0,03979	0,12836	0,00579	15,29267	0,08611	0,00230	20,11795	0,04034	0,00079	0,69149
barbara.gif	512x512	0,15308	0,11700	0,02687	-1,34230	0,11477	0,02796	-1,36992	0,11435	0,02779	-1,39375
bates.bmp	487x727	0,09648	0,31379	0,01665	13,04928	0,21654	0,00348	34,48821	0,10037	0,00277	1,40251
bird256gray.bmp	256x256	0,07031	0,24802	0,01012	17,55230	0,17717	0,00597	17,90922	0,07981	0,00433	2,19180
boat.bmp	512x512	0,04224	0,13973	0,00613	15,90656	0,09463	0,00124	42,17584	0,04330	0,00128	0,83585
boat256.jpg	256x256	0,11914	0,13943	0,00912	2,22397	0,10212	0,00565	-3,00915	0,06619	0,01173	-4,51504
bridge256.bmp	256x256	0,02393	0,05515	0,00377	8,28722	0,03574	0,00294	4,02440	0,01567	0,00283	-2,91163
bridge256.jpg	256x256	0,06348	0,03574	0,00646	-4,29049	0,03489	0,00640	-4,46508	0,03489	0,00640	-4,46508
Bw1.bmp	267x234	0,86050	0,48250	0,08825	-4,28335	0,47732	0,09125	-4,19935	0,46541	0,09883	-3,99773
Bw3.bmp	296x295	0,78529	0,46586	0,10370	-3,08030	0,46104	0,10701	-3,02998	0,45606	0,11095	-2,96738
bw4.bmp	470x221	0,70817	0,43238	0,10001	-2,75773	0,42858	0,10301	-2,71429	0,42259	0,10803	-2,64355
Bw7.bmp	513x389	0,69792	0,48818	0,12211	-1,71763	0,48415	0,12650	-1,68985	0,48418	0,12627	-1,69266
camera256.bmp	256x256	0,16162	0,21096	0,01042	4,73601	0,15623	0,00656	-0,82191	0,09377	0,02174	-3,12153
cat.png	490x733	0,02090	0,06405	0,00115	37,48216	0,04842	0,00229	12,04195	0,02096	0,00074	0,08513
DSC00001.JPG	768x1024	0,16187	0,16835	0,00490	1,32542	0,14041	0,01806	-1,18805	0,13995	0,01802	-1,21587
DSC00007.JPG	1024x768	0,16951	0,24737	0,01648	4,72426	0,16364	0,00601	-0,97759	0,14943	0,01571	-1,27826
DSC00014.JPG	1024x768	0,09920	0,28618	0,00522	35,81079	0,20731	0,01362	7,94000	0,09928	0,00098	0,07651
DSC00016.JPG	768x1024	0,12976	0,32533	0,01133	17,26289	0,23405	0,01357	7,68602	0,12638	0,00269	-1,25384
DSC00040.JPG	768x1024	0,11776	0,30933	0,01615	11,85991	0,23797	0,01016	11,83692	0,11561	0,00143	-1,50589
DSC00047.JPG	1024x768	0,12793	0,22294	0,02268	4,18970	0,12501	0,00540	-0,54072	0,10709	0,01638	-1,27275
DSC00080.JPG	1024x768	0,07756	0,21026	0,00226	58,70423	0,15813	0,00810	9,94460	0,07703	0,00077	-0,68448

Табл. 4: Резултати от изследването на корелации в изображението - *0-Analysis*

Изображение	Размер	Оригинал (o)	JPEG 70			JPEG 80			JPEG 90		
			μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ
DSC00852.JPG	1024x768	0,11995	0,12538	0,00597	0,90945	0,10679	0,00970	-1,35727	0,09608	0,00962	-2,48208
DSC00861.JPG	1024x768	0,07589	0,19114	0,01282	8,99290	0,08315	0,00480	1,51428	0,06583	0,00771	-1,30500
DSC00901.JPG	768x1024	0,06620	0,11953	0,00701	7,60249	0,06850	0,00127	1,80714	0,05756	0,00606	-1,42610
DSC00911.JPG	768x1024	0,02999	0,05799	0,00217	12,92337	0,03428	0,00310	1,38301	0,02701	0,00247	-1,20766
DSC01263.JPG	1024x768	0,06954	0,39520	0,03015	10,80190	0,26670	0,00131	149,99441	0,08785	0,01523	1,20223
DSC01269.JPG	1024x768	0,03076	0,10275	0,00284	25,36254	0,07351	0,00547	7,81363	0,03092	0,00037	0,44472
DSC01432.JPG	1024x768	0,02458	0,04099	0,00105	15,56744	0,02689	0,00177	1,30680	0,02253	0,00152	-1,35113
DSC01443.JPG	1024x768	0,02779	0,03196	0,00227	1,83091	0,02808	0,00038	0,74543	0,02393	0,00251	-1,53811
DSC01481.JPG	768x1024	0,05461	0,17913	0,00140	89,00348	0,12441	0,00171	40,75470	0,05427	0,00060	-0,55658
DSC01534.JPG	1024x768	0,04708	0,13354	0,00545	15,86701	0,05234	0,00400	1,31669	0,04156	0,00425	-1,29921
DSC02037.JPG	1000x750	0,02632	0,19131	0,00527	31,32196	0,11975	0,00363	25,75472	0,03367	0,00621	1,18309
DSC02087.JPG	750x1000	0,08856	0,24794	0,00711	22,42933	0,18400	0,01008	9,46635	0,08783	0,00082	-0,89291
Flowers.bmp	1043x474	0,03070	0,07934	0,00261	18,61479	0,05463	0,00171	13,98565	0,02924	0,00149	-0,98147
FRUIT.BMP	512x512	0,04016	0,13932	0,00242	40,89144	0,09428	0,00131	41,21245	0,04265	0,00209	1,19150
Fruit.color.bmp	487x414	0,05850	0,08382	0,00513	4,93875	0,06571	0,00409	1,76244	0,04056	0,00957	-1,87324
fruits.bmp	512x512	0,04614	0,16749	0,00465	26,07047	0,10883	0,00132	47,40736	0,04795	0,00129	1,39561
geom2.bmp	256x256	0,02051	0,05657	0,00355	10,15102	0,04023	0,00292	6,76725	0,01958	0,00243	-0,38238
girl.png	768x512	0,09888	0,28371	0,02372	7,79200	0,20532	0,01152	9,23993	0,10025	0,00248	0,55355
goldhill.bmp	512x512	0,03369	0,11650	0,00352	23,53839	0,07704	0,00208	20,86334	0,03539	0,00158	1,07404
goldhill256.jpg	256x256	0,06543	0,08308	0,00516	3,42155	0,06006	0,00394	-1,36229	0,03669	0,00750	-3,83221
kai.bmp	256x256	0,03174	0,10007	0,00532	12,83742	0,06948	0,00386	9,78481	0,03059	0,00278	-0,41268
kaplan1.bmp	360x624	0,11595	0,31787	0,02159	9,35413	0,23279	0,00808	14,46545	0,11784	0,00169	1,11669
Lena256.bmp	256x256	0,02979	0,10310	0,00416	17,63088	0,06787	0,00275	13,83187	0,02976	0,00165	-0,01478
lena512.bmp	512x512	0,03845	0,14276	0,00368	28,33802	0,09531	0,00166	34,19759	0,04034	0,00215	0,87602
Leonardo.bmp	695x1040	0,02804	0,09684	0,00063	109,10192	0,06078	0,00047	70,24189	0,02824	0,00037	0,54757
Mouse.bmp	936x1035	0,59113	0,59171	0,04401	0,01317	0,56860	0,04229	-0,53282	0,54283	0,04170	-1,15840
MRI.bmp	256x256	0,25879	0,20034	0,02805	-2,08399	0,18286	0,02791	-2,72040	0,15786	0,02618	-3,85477
Niagara.bmp	399x450	0,21793	0,18321	0,04321	-0,80356	0,17652	0,04087	-1,01329	0,15909	0,03282	-1,79292
P1.bmp	345x406	0,60326	0,41524	0,09583	-1,96188	0,40581	0,09790	-2,01668	0,39474	0,09784	-2,13106
P10.bmp	457x507	0,18672	0,27405	0,02921	2,98970	0,21712	0,02394	1,27010	0,15019	0,02324	-1,57127
P2.bmp	274x406	0,00971	0,04224	0,00225	14,44302	0,02678	0,00178	9,57609	0,01290	0,00134	2,38895
P3.bmp	406x433	0,09407	0,14126	0,01324	3,56492	0,10832	0,01131	1,25947	0,07319	0,01229	-1,69939
P4.bmp	406x433	0,17222	0,17283	0,02752	0,02220	0,15101	0,02810	-0,75495	0,12474	0,02598	-1,82770
P5.bmp	323x418	0,59856	0,42094	0,09282	-1,91362	0,40659	0,09452	-2,03109	0,39029	0,09580	-2,17400
P6.bmp	487x368	0,04764	0,13358	0,00667	12,87955	0,09341	0,00330	13,86657	0,04648	0,00185	-0,63082
P7.bmp	442x554	0,06996	0,18703	0,01328	8,81641	0,13674	0,01011	6,60808	0,07040	0,00210	0,21023
P8.bmp	267x347	0,01656	0,06838	0,00395	13,10819	0,04269	0,00274	9,54670	0,01649	0,00141	-0,05006
P9.bmp	454x323	0,02433	0,09605	0,00326	21,99396	0,06066	0,00251	14,44813	0,02530	0,00172	0,56476

Табл. 4: Резултати от изследването на корелации в изображението - *0-Analysis*

Изображение	Размер	Оригинал (o)	JPEG 70			JPEG 80			JPEG 90		
			μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ
pentagon.tiff	1024x1024	0,02954	0,10615	0,00085	90,39714	0,06821	0,00033	115,58517	0,02976	0,00035	0,62297
Peppers.bmp	512x512	0,04504	0,16255	0,00603	19,47341	0,11028	0,00185	35,34194	0,04826	0,00224	1,43362
pepperscolour.png	512x512	0,03650	0,13620	0,00245	40,77702	0,09120	0,00181	30,25366	0,03873	0,00210	1,06060
Picasso.bmp	1618x2064	0,09515	0,09757	0,00180	1,34641	0,09297	0,00193	-1,13260	0,09244	0,00189	-1,43492
pool.bmp	510x383	0,43533	0,45834	0,08523	0,27004	0,41960	0,07331	-0,21455	0,33301	0,05398	-1,89556
pool.png	510x383	0,45221	0,46876	0,08817	0,18769	0,43046	0,07589	-0,28658	0,35117	0,05805	-1,74073
thoraxl.bmp	280x301	0,05367	0,13840	0,00433	19,57756	0,09488	0,00485	8,50368	0,04927	0,00296	-1,48491
watch.bmp	1024x768	0,17643	0,36489	0,03318	5,67947	0,28995	0,01779	6,38150	0,17272	0,00734	-0,50611
watch.png	1024x768	0,17688	0,35881	0,03297	5,51766	0,28364	0,01763	6,05580	0,16944	0,00728	-1,02121
water.png	1024x768	0,02458	0,08455	0,00066	91,24170	0,05331	0,00031	93,06671	0,02465	0,00026	0,26743
zelda256.bmp	256x256	0,05469	0,07554	0,00489	4,26340	0,05854	0,00332	1,16114	0,03489	0,00724	-2,73607
zelda512.jpg	512x512	0,07629	0,09411	0,01283	1,38905	0,07634	0,00154	0,03169	0,05837	0,01386	-1,29328

Приложение 2

Табл. 1: Резултати от прилагането на филтри върху изображението - усредняващ филтър

Изображение	Размер	Оригинал (o)	JPEG 70			JPEG 80			JPEG 90		
			μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ
airplane.png	512x512	39,4161	39,5257	0,0185	5,9254	39,4859	0,0129	5,3967	39,4026	0,0056	-2,3975
arctichare.png	594x400	29,8526	29,3067	0,0195	-28,0064	29,3696	0,0179	-26,9201	29,4749	0,0091	-41,7188
baboon.bmp	512x512	30,9148	31,1005	0,0138	13,4891	31,0177	0,0121	8,5006	30,9059	0,0061	-1,4573
baboon.png	512x512	31,3599	31,5306	0,0224	7,6285	31,4492	0,0094	9,5028	31,3478	0,0096	-1,2619
barbara.bmp	512x512	32,2899	32,4166	0,0163	7,7760	32,3604	0,0126	5,5843	32,2880	0,0038	-0,4932
barbara.gif	512x512	32,2931	32,4559	0,0198	8,2184	32,3928	0,0244	4,0860	32,3069	0,0096	1,4357
bates.bmp	487x727	35,4375	35,5278	0,0089	10,0974	35,4991	0,0075	8,2556	35,4344	0,0057	-0,5375
bird256gray.bmp	256x256	30,0729	30,2133	0,0487	2,8839	30,1546	0,0345	2,3660	30,0571	0,0147	-1,0704
boat.bmp	512x512	34,0426	34,1869	0,0155	9,3396	34,1390	0,0105	9,1769	34,0336	0,0057	-1,5711
boat256.jpg	256x256	38,1977	38,2615	0,0375	1,7015	38,2329	0,0391	0,9004	38,1433	0,0258	-2,1063
bridge256.bmp	256x256	42,4747	42,4258	0,0399	-1,2243	42,5078	0,0310	1,0650	42,4195	0,0191	-2,8853
bridge256.jpg	256x256	42,4844	42,5022	0,0526	0,3377	42,5402	0,0325	1,7167	42,4425	0,0095	-4,4017
Bw1.bmp	267x234	57,0639	49,5723	0,2664	-28,1182	52,0778	0,1432	-34,8160	54,8830	0,1168	-18,6799
Bw3.bmp	296x295	88,7220	74,7439	0,1583	-88,2917	79,2751	0,1214	-77,8397	84,8292	0,1070	-36,3742
bw4.bmp	470x221	119,2579	101,0429	0,2480	-73,4543	107,6790	0,1094	-105,8372	114,4706	0,1909	-25,0777
Bw7.bmp	513x389	117,0746	99,1526	0,1146	-156,4044	105,8792	0,1101	-101,6459	112,3445	0,1781	-26,5538
camera256.bmp	256x256	45,7088	45,2036	0,0371	-13,6255	45,6801	0,0253	-1,1297	45,6598	0,0157	-3,1177
cat.png	490x733	49,8553	49,9326	0,0125	6,2129	49,9106	0,0132	4,1963	49,8279	0,0047	-5,8791
DSC00001.JPG	768x1024	29,3007	29,3011	0,0085	0,0394	29,3750	0,0124	5,9766	29,2726	0,0019	-14,4459
DSC00007.JPG	1024x768	27,0700	26,1262	0,0086	-109,6068	26,1732	0,0024	-374,4712	26,6509	0,0143	-29,3124
DSC00014.JPG	1024x768	27,1186	27,2475	0,0071	18,2492	27,2077	0,0141	6,3071	27,1041	0,0021	-6,7717
DSC00016.JPG	768x1024	11,9791	12,1047	0,0135	9,3124	12,0424	0,0061	10,3911	11,9672	0,0045	-2,6203
DSC00040.JPG	768x1024	34,1707	33,9789	0,0097	-19,8614	34,0964	0,0115	-6,4452	34,0753	0,0016	-58,4369
DSC00047.JPG	1024x768	31,8305	31,5381	0,0112	-26,1190	31,4881	0,0066	-51,7398	31,6987	0,0190	-6,9358
DSC00080.JPG	1024x768	20,5744	20,6603	0,0142	6,0546	20,6507	0,0241	3,1629	20,5419	0,0033	-9,8171
DSC00852.JPG	1024x768	45,5736	42,9328	0,0082	-322,3192	44,4349	0,0055	-205,5635	44,7457	0,0108	-76,7079
DSC00861.JPG	1024x768	24,2533	24,2489	0,0147	-0,3016	24,1235	0,0046	-28,3138	24,1952	0,0236	-2,4599
DSC00901.JPG	768x1024	25,4107	25,3262	0,0077	-11,0096	25,1143	0,0106	-27,9822	25,2988	0,0250	-4,4832
DSC00911.JPG	768x1024	28,6703	28,7825	0,0093	12,0880	28,6951	0,0123	2,0147	28,6255	0,0234	-1,9146
DSC01263.JPG	1024x768	12,1979	12,2477	0,0254	1,9649	12,2122	0,0062	2,3361	12,1732	0,0103	-2,3916
DSC01269.JPG	1024x768	30,7443	30,8889	0,0049	29,2680	30,8542	0,0330	3,3269	30,7089	0,0032	-11,0219
DSC01432.JPG	1024x768	35,8475	35,9244	0,0117	6,5814	35,8514	0,0153	0,2575	35,7918	0,0280	-1,9893
DSC01443.JPG	1024x768	45,7541	45,6373	0,0041	-28,5976	45,6156	0,0127	-10,9063	45,6095	0,0018	-80,6885
DSC01481.JPG	768x1024	23,5474	23,4074	0,0129	-10,8753	23,4397	0,0102	-10,5652	23,4676	0,0026	-31,2313

Табл. 1: Резултати от прилагането на филтри върху изображението - усредняващ филтър

Изображение	Размер	Оригинал (о)	JPEG 70			JPEG 80			JPEG 90		
			μ	σ	(μ-о)/σ	μ	σ	(μ-о)/σ	μ	σ	(μ-о)/σ
DSC01534.JPG	1024x768	42,6422	42,6133	0,0110	-2,6376	42,5293	0,0066	-17,2349	42,5905	0,0183	-2,8301
DSC02037.JPG	1000x750	36,8914	36,9198	0,0047	6,0603	36,8794	0,0116	-1,0393	36,8663	0,0213	-1,1809
DSC02087.JPG	750x1000	32,6342	31,9645	0,0085	-78,5435	32,0979	0,0171	-31,3291	32,1812	0,0024	-192,4993
Flowers.bmp	1043x474	34,8679	35,0061	0,0092	14,9472	34,9395	0,0068	10,5949	34,8553	0,0049	-2,5665
FRUIT.BMP	512x512	27,9624	28,1303	0,0226	7,4393	28,0475	0,0140	6,0761	27,9576	0,0088	-0,5487
Fruit.color.bmp	487x414	39,5007	38,7726	0,0294	-24,7954	39,0911	0,0090	-45,5916	39,2857	0,0085	-25,3411
fruits.bmp	512x512	28,5650	28,2851	0,0249	-11,2636	28,3457	0,0123	-17,7861	28,4025	0,0069	-23,5775
geom2.bmp	256x256	143,4119	133,3806	0,3265	-30,7205	136,7280	0,1417	-47,1667	140,8480	0,1493	-17,1702
girl.png	768x512	28,9092	28,4433	0,0136	-34,2248	28,5462	0,0150	-24,2522	28,6701	0,0041	-58,8852
goldhill.bmp	512x512	28,6807	28,8377	0,0199	7,9055	28,7629	0,0251	3,2806	28,6718	0,0051	-1,7535
goldhill256.jpg	256x256	34,8929	35,0092	0,0528	2,2036	34,9300	0,0321	1,1560	34,8428	0,0250	-2,0060
kai.bmp	256x256	55,9742	54,7150	0,0472	-26,6797	55,0783	0,0411	-21,7943	55,4561	0,0127	-40,6999
kaplan1.bmp	360x624	51,8189	51,2649	0,0248	-22,2982	51,3890	0,0134	-32,1266	51,5288	0,0075	-38,8868
Lena256.bmp	256x256	44,1892	44,3424	0,0416	3,6795	44,2608	0,0287	2,4962	44,1779	0,0171	-0,6574
lena512.bmp	512x512	34,0936	34,1894	0,0212	4,5161	34,1513	0,0082	7,0506	34,0922	0,0048	-0,2871
Leonardo.bmp	695x1040	26,1430	26,2267	0,0094	8,8631	26,1741	0,0083	3,7686	26,1069	0,0025	-14,7117
Mouse.bmp	936x1035	51,8767	49,8645	0,0061	-329,0015	50,3007	0,0037	-428,0664	50,9807	0,0020	-446,6784
MRI.bmp	256x256	52,6478	52,6564	0,0326	0,2645	52,6949	0,0214	2,2022	52,6372	0,0156	-0,6820
Niagara.bmp	399x450	29,5731	29,4699	0,0231	-4,4613	29,5041	0,0165	-4,1946	29,5183	0,0072	-7,6609
P1.bmp	345x406	40,8976	37,5807	0,0375	-88,4319	38,3814	0,0325	-77,4056	39,7755	0,0288	-39,0042
P10.bmp	457x507	18,3516	17,9738	0,0286	-13,2068	18,0292	0,0238	-13,5546	18,1685	0,0071	-25,8019
P2.bmp	274x406	76,1665	72,9085	0,0592	-54,9928	74,1447	0,0337	-59,9252	75,5373	0,0263	-23,8902
P3.bmp	406x433	40,1305	38,0210	0,0246	-85,7480	38,5638	0,0229	-68,3232	39,4292	0,0130	-54,0422
P4.bmp	406x433	42,5629	41,9098	0,0263	-24,8068	42,0631	0,0345	-14,4722	42,2578	0,0127	-23,9456
P5.bmp	323x418	22,4527	21,5903	0,0327	-26,3521	21,8651	0,0541	-10,8581	22,0226	0,0170	-25,2469
P6.bmp	487x368	47,8881	46,8377	0,0391	-26,8980	47,0903	0,0122	-65,3750	47,4799	0,0126	-32,5071
P7.bmp	442x554	20,9760	21,1020	0,0360	3,5027	21,0394	0,0157	4,0359	20,9629	0,0093	-1,4097
P8.bmp	267x347	34,4279	33,9835	0,0592	-7,5035	34,2142	0,0236	-9,0681	34,3854	0,0077	-5,5163
P9.bmp	454x323	18,8157	18,8909	0,0367	2,0459	18,8711	0,0228	2,4279	18,8014	0,0105	-1,3711
pentagon.tiff	1024x1024	26,1937	26,3593	0,0048	34,3167	26,2809	0,0075	11,5922	26,1940	0,0016	0,1641
Peppers.bmp	512x512	27,1836	27,3053	0,0228	5,3305	27,2535	0,0150	4,6718	27,1809	0,0069	-0,3960
pepperscolour.png	512x512	33,0533	32,5143	0,0133	-40,3937	32,5521	0,0159	-31,5085	32,6316	0,0097	-43,5236
Picasso.bmp	1618x2064	17,6448	17,8552	0,0020	106,5462	17,7382	0,0017	55,5169	17,6439	0,0011	-0,8961
pool.bmp	510x383	18,8055	18,8607	0,0355	1,5570	18,8355	0,0327	0,9173	18,7876	0,0059	-3,0528
pool.png	510x383	20,0139	19,8824	0,0272	-4,8296	19,9594	0,0180	-3,0284	19,9853	0,0086	-3,3493
thoraxl.bmp	280x301	19,5583	19,6047	0,0224	2,0729	19,5818	0,0386	0,6088	19,5270	0,0166	-1,8894
watch.bmp	1024x768	27,1712	27,1654	0,0103	-0,5625	27,1564	0,0057	-2,5745	27,1387	0,0022	-15,0673
watch.png	1024x768	28,2406	28,1390	0,0059	-17,2154	28,1593	0,0070	-11,5816	28,1663	0,0020	-37,7816
water.png	1024x768	33,8141	33,3270	0,0089	-54,5386	33,4174	0,0067	-59,0868	33,5207	0,0033	-88,6297

Табл. 1: Резултати от прилагането на филтри върху изображението - усредняващ филтър

Изображение	Размер	Оригинал (o)	JPEG 70			JPEG 80			JPEG 90		
			μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ
zelda256.bmp	256x256	29,7181	29,9093	0,0364	5,2495	29,8288	0,0293	3,7739	29,7061	0,0094	-1,2769
zelda512.jpg	512x512	18,8829	19,0161	0,0398	3,3440	18,9413	0,0355	1,6484	18,8775	0,0149	-0,3569

Табл. 2: Резултати от прилагането на филтри върху изображението - Гаусов филтър

Изображение	Размер	Оригинал (o)	JPEG 70			JPEG 80			JPEG 90		
			μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ
airplane.png	512x512	31,8677	31,9620	0,0168	5,5982	31,9326	0,0122	5,3377	31,8602	0,0051	-1,4811
arctichare.png	594x400	24,2526	23,7919	0,0149	-30,9576	23,8435	0,0145	-28,1668	23,9323	0,0069	-46,2980
baboon.bmp	512x512	25,5997	25,7564	0,0118	13,2416	25,6854	0,0100	8,5665	25,5883	0,0064	-1,7769
baboon.png	512x512	25,9405	26,0901	0,0175	8,5247	26,0175	0,0086	8,9408	25,9315	0,0072	-1,2390
barbara.bmp	512x512	26,0567	26,1571	0,0152	6,6148	26,1131	0,0098	5,7373	26,0544	0,0036	-0,6418
barbara.gif	512x512	26,0496	26,1938	0,0172	8,3845	26,1335	0,0195	4,3024	26,0645	0,0110	1,3628
bates.bmp	487x727	28,0979	28,1667	0,0098	7,0141	28,1464	0,0065	7,4644	28,0945	0,0045	-0,7577
bird256gray.bmp	256x256	24,3926	24,5158	0,0394	3,1252	24,4692	0,0322	2,3747	24,3890	0,0131	-0,2753
boat.bmp	512x512	27,3736	27,4888	0,0112	10,2720	27,4452	0,0105	6,8224	27,3630	0,0078	-1,3668
boat256.jpg	256x256	30,3550	30,4241	0,0320	2,1598	30,3950	0,0307	1,2998	30,3216	0,0183	-1,8276
bridge256.bmp	256x256	33,6406	33,6174	0,0323	-0,7193	33,6792	0,0272	1,4180	33,6050	0,0130	-2,7342
bridge256.jpg	256x256	33,6492	33,6885	0,0405	0,9706	33,7062	0,0255	2,2359	33,6248	0,0094	-2,6058
Bw1.bmp	267x234	46,9507	40,5275	0,2132	-30,1206	42,7161	0,1142	-37,0850	45,0646	0,0888	-21,2449
Bw3.bmp	296x295	73,4413	61,7761	0,1287	-90,6291	65,5384	0,0940	-84,0344	70,1921	0,0859	-37,8139
bw4.bmp	470x221	94,0691	79,5392	0,1959	-74,1724	84,8377	0,0799	-115,4864	90,2378	0,1475	-25,9799
Bw7.bmp	513x389	95,8114	80,9925	0,0940	-157,7181	86,5414	0,0898	-103,2624	91,8833	0,1465	-26,8038
camera256.bmp	256x256	37,2472	36,8142	0,0317	-13,6584	37,2123	0,0211	-1,6557	37,2008	0,0101	-4,6026
cat.png	490x733	41,8153	41,8765	0,0103	5,9698	41,8566	0,0096	4,3164	41,7890	0,0048	-5,4809
DSC00001.JPG	768x1024	24,0382	24,0420	0,0075	0,5064	24,0992	0,0096	6,3430	24,0156	0,0015	-15,2415
DSC00007.JPG	1024x768	22,0813	21,2904	0,0080	-98,8816	21,3344	0,0023	-321,7536	21,7344	0,0122	-28,5149
DSC00014.JPG	1024x768	21,0475	21,1518	0,0067	15,6419	21,1250	0,0122	6,3430	21,0349	0,0028	-4,4301
DSC00016.JPG	768x1024	9,5628	9,6718	0,0135	8,0658	9,6157	0,0052	10,2173	9,5580	0,0050	-0,9571
DSC00040.JPG	768x1024	27,7687	27,6073	0,0065	-24,8501	27,7100	0,0084	-6,9802	27,6933	0,0017	-43,7506
DSC00047.JPG	1024x768	26,0017	25,7636	0,0094	-25,3827	25,7171	0,0052	-55,2090	25,8939	0,0166	-6,4887
DSC00080.JPG	1024x768	16,2332	16,3050	0,0127	5,6384	16,2950	0,0186	3,3160	16,2080	0,0023	-10,7534
DSC00852.JPG	1024x768	37,6459	35,4524	0,0079	-276,0178	36,6967	0,0053	-179,3004	36,9533	0,0079	-87,7629
DSC00861.JPG	1024x768	19,9952	19,9932	0,0107	-0,1825	19,8822	0,0048	-23,5911	19,9450	0,0198	-2,5267
DSC00901.JPG	768x1024	21,2251	21,1540	0,0059	-11,9555	20,9772	0,0086	-28,9550	21,1318	0,0211	-4,4179
DSC00911.JPG	768x1024	23,7060	23,7998	0,0098	9,5500	23,7291	0,0100	2,3024	23,6690	0,0205	-1,8090
DSC01263.JPG	1024x768	9,6627	9,7016	0,0228	1,7086	9,6689	0,0062	1,0080	9,6427	0,0088	-2,2829
DSC01269.JPG	1024x768	24,0897	24,2079	0,0037	32,3574	24,1806	0,0270	3,3673	24,0630	0,0019	-13,8485
DSC01432.JPG	1024x768	29,5972	29,6632	0,0100	6,5796	29,6008	0,0123	0,2847	29,5538	0,0226	-1,9214

Табл. 2: Резултати от прилагането на филтри върху изображението - Гаусов филтър

Изображение	Размер	Оригинал (o)	JPEG 70			JPEG 80			JPEG 90		
			μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ
DSC01443.JPG	1024x768	38,1210	38,0177	0,0029	-36,0901	38,0011	0,0108	-11,1221	38,0013	0,0011	-112,6530
DSC01481.JPG	768x1024	19,7328	19,6063	0,0119	-10,5940	19,6357	0,0085	-11,4636	19,6594	0,0017	-42,2296
DSC01534.JPG	1024x768	33,1596	33,1300	0,0098	-3,0024	33,0684	0,0047	-19,3102	33,1147	0,0160	-2,8031
DSC02037.JPG	1000x750	29,2270	29,2515	0,0038	6,5030	29,2171	0,0086	-1,1600	29,2081	0,0173	-1,0952
DSC02087.JPG	750x1000	25,7855	25,2529	0,0084	-63,0275	25,3565	0,0128	-33,6193	25,4230	0,0025	-144,2579
Flowers.bmp	1043x474	28,5557	28,6671	0,0074	15,1464	28,6085	0,0051	10,3814	28,5415	0,0056	-2,5251
FRUIT.BMP	512x512	22,5625	22,7071	0,0198	7,3099	22,6339	0,0131	5,4542	22,5615	0,0063	-0,1542
Fruit.color.bmp	487x414	32,6579	32,0558	0,0235	-25,6671	32,3140	0,0105	-32,7830	32,4754	0,0068	-26,7698
fruits.bmp	512x512	23,7866	23,5893	0,0195	-10,1097	23,6240	0,0103	-15,7636	23,6625	0,0055	-22,6116
geom2.bmp	256x256	119,3849	110,8293	0,2527	-33,8569	113,6910	0,1115	-51,0599	117,2232	0,1264	-17,0997
girl.png	768x512	22,7182	22,3121	0,0124	-32,8606	22,3951	0,0126	-25,5633	22,5076	0,0039	-54,1063
goldhill.bmp	512x512	22,8059	22,9315	0,0165	7,6062	22,8710	0,0200	3,2525	22,7998	0,0042	-1,4556
goldhill256.jpg	256x256	27,8825	27,9894	0,0438	2,4424	27,9189	0,0261	1,3974	27,8438	0,0204	-1,8951
kai.bmp	256x256	45,1142	44,0515	0,0368	-28,8835	44,3436	0,0365	-21,1096	44,6754	0,0110	-40,0021
kaplan1.bmp	360x624	40,9848	40,5383	0,0212	-21,0909	40,6398	0,0107	-32,1344	40,7579	0,0077	-29,5088
Lena256.bmp	256x256	35,6812	35,8025	0,0374	3,2387	35,7393	0,0234	2,4848	35,6746	0,0128	-0,5168
lena512.bmp	512x512	28,2035	28,2886	0,0173	4,9137	28,2549	0,0076	6,8043	28,2023	0,0047	-0,2609
Leonardo.bmp	695x1040	20,7264	20,8227	0,0085	11,3181	20,7658	0,0074	5,3226	20,7023	0,0018	-13,5773
Mouse.bmp	936x1035	42,6596	40,9339	0,0045	-387,5629	41,3138	0,0033	-413,0469	41,8989	0,0014	-538,1792
MRI.bmp	256x256	42,6143	42,6195	0,0305	0,1708	42,6609	0,0173	2,6923	42,6102	0,0120	-0,3366
Niagara.bmp	399x450	22,5474	22,4914	0,0172	-3,2517	22,5085	0,0142	-2,7497	22,5108	0,0081	-4,5028
P1.bmp	345x406	34,3115	31,4940	0,0311	-90,7172	32,1813	0,0298	-71,4922	33,3574	0,0229	-41,6018
P10.bmp	457x507	15,9596	15,6079	0,0237	-14,8709	15,6670	0,0155	-18,8365	15,8013	0,0056	-28,0329
P2.bmp	274x406	60,4997	57,7754	0,0462	-58,9776	58,8148	0,0257	-65,5059	59,9650	0,0190	-28,2036
P3.bmp	406x433	33,2140	31,3968	0,0173	-104,9829	31,8592	0,0150	-90,2139	32,6156	0,0099	-60,6134
P4.bmp	406x433	34,7846	34,2602	0,0245	-21,4095	34,3809	0,0279	-14,4686	34,5384	0,0097	-25,4375
P5.bmp	323x418	17,7957	17,1026	0,0253	-27,3963	17,3246	0,0457	-10,3049	17,4390	0,0141	-25,3777
P6.bmp	487x368	37,5230	36,7268	0,0272	-29,2263	36,9177	0,0097	-62,1010	37,2135	0,0108	-28,5444
P7.bmp	442x554	16,9106	17,0215	0,0305	3,6387	16,9616	0,0131	3,9063	16,9013	0,0048	-1,9503
P8.bmp	267x347	28,3486	27,9845	0,0497	-7,3306	28,1745	0,0225	-7,7454	28,3158	0,0080	-4,1184
P9.bmp	454x323	15,2130	15,2954	0,0291	2,8369	15,2641	0,0196	2,6119	15,2007	0,0069	-1,7689
pentagon.tiff	1024x1024	21,6226	21,7590	0,0045	30,1742	21,6949	0,0053	13,6663	21,6227	0,0012	0,1162
Peppers.bmp	512x512	22,0443	22,1539	0,0240	4,5553	22,1135	0,0112	6,1756	22,0499	0,0056	1,0022
pepperscolour.png	512x512	26,8129	26,3777	0,0102	-42,5181	26,4039	0,0144	-28,4522	26,4704	0,0082	-41,6733
Picasso.bmp	1618x2064	14,1742	14,3485	0,0016	110,3772	14,2496	0,0017	43,8488	14,1742	0,0010	-0,0009
pool.bmp	510x383	16,2053	16,2388	0,0318	1,0520	16,2217	0,0286	0,5716	16,1894	0,0063	-2,5372
pool.png	510x383	17,2711	17,1543	0,0223	-5,2407	17,2186	0,0137	-3,8422	17,2413	0,0070	-4,2294
thoraxl.bmp	280x301	14,7582	14,8467	0,0167	5,3060	14,8003	0,0301	1,3965	14,7526	0,0079	-0,7011
watch.bmp	1024x768	22,7992	22,7901	0,0082	-1,1211	22,7850	0,0057	-2,4866	22,7701	0,0025	-11,5885

Табл. 2: Резултати от прилагането на филтри върху изображението - Гаусов филтър

Изображение	Размер	Оригинал (o)	JPEG 70			JPEG 80			JPEG 90		
			μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ
watch.png	1024x768	23,6785	23,5913	0,0048	-18,1785	23,6074	0,0066	-10,7254	23,6168	0,0014	-45,3241
water.png	1024x768	27,8942	27,4707	0,0087	-48,8021	27,5486	0,0060	-58,0336	27,6444	0,0029	-85,2566
zelda256.bmp	256x256	23,6743	23,8231	0,0276	5,3962	23,7597	0,0213	4,0056	23,6629	0,0080	-1,4295
zelda512.jpg	512x512	15,3385	15,4438	0,0362	2,9048	15,3822	0,0302	1,4499	15,3343	0,0110	-0,3824

Табл. 3: Резултати от прилагането на филтри върху изображението - медианен филтър

Изображение	Размер	Оригинал (o)	JPEG 70			JPEG 80			JPEG 90		
			μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ
airplane.png	512x512	42,4041	42,5229	0,0256	4,6492	42,4725	0,0164	4,1787	42,3944	0,0100	-0,9689
arctichare.png	594x400	31,2006	30,6516	0,0291	-18,8823	30,6998	0,0175	-28,6015	30,8257	0,0068	-54,9316
baboon.bmp	512x512	32,7272	32,9270	0,0178	11,2193	32,8381	0,0171	6,4830	32,7187	0,0062	-1,3744
baboon.png	512x512	33,2096	33,3983	0,0319	5,9177	33,3039	0,0147	6,3999	33,1917	0,0093	-1,9318
barbara.bmp	512x512	34,3375	34,4274	0,0242	3,7086	34,4048	0,0155	4,3417	34,3356	0,0048	-0,4075
barbara.gif	512x512	34,3745	34,5112	0,0250	5,4702	34,4354	0,0214	2,8488	34,3721	0,0067	-0,3543
bates.bmp	487x727	39,4955	39,5717	0,0225	3,3888	39,5575	0,0127	4,8854	39,4921	0,0065	-0,5259
bird256gray.bmp	256x256	32,3278	32,4417	0,0639	1,7828	32,4166	0,0417	2,1329	32,3152	0,0169	-0,7446
boat.bmp	512x512	35,9588	36,1107	0,0201	7,5573	36,0563	0,0110	8,8520	35,9547	0,0072	-0,5725
boat256.jpg	256x256	40,7397	40,8196	0,0561	1,4236	40,7751	0,0361	0,9785	40,6945	0,0282	-1,6042
bridge256.bmp	256x256	46,5133	46,4359	0,0540	-1,4311	46,4952	0,0418	-0,4327	46,4470	0,0158	-4,1878
bridge256.jpg	256x256	46,5193	46,4739	0,0645	-0,7033	46,5445	0,0597	0,4221	46,4466	0,0182	-3,9940
Bw1.bmp	267x234	72,1943	62,0958	0,3758	-26,8689	65,4559	0,1615	-41,7203	69,2063	0,1354	-22,0686
Bw3.bmp	296x295	101,8905	85,5466	0,2278	-71,7487	90,7910	0,1340	-82,8630	97,2775	0,1329	-34,6991
bw4.bmp	470x221	137,2435	116,2980	0,3640	-57,5390	123,8668	0,1475	-90,7061	131,7391	0,2465	-22,3325
Bw7.bmp	513x389	125,5305	106,2741	0,1950	-98,7589	113,4677	0,1617	-74,5935	120,4361	0,2245	-22,6934
camera256.bmp	256x256	49,1575	48,6042	0,0608	-9,1036	49,1264	0,0348	-0,8924	49,1068	0,0152	-3,3410
cat.png	490x733	51,6599	51,7553	0,0190	5,0286	51,7214	0,0224	2,7497	51,6305	0,0056	-5,2082
DSC00001.JPG	768x1024	31,0824	31,0709	0,0132	-0,8725	31,1532	0,0142	4,9931	31,0511	0,0037	-8,3999
DSC00007.JPG	1024x768	28,3437	27,3468	0,0084	-118,9832	27,3832	0,0036	-268,3847	27,8890	0,0128	-35,6215
DSC00014.JPG	1024x768	29,8789	30,0245	0,0085	17,0644	29,9738	0,0136	6,9796	29,8626	0,0028	-5,9304
DSC00016.JPG	768x1024	12,9949	13,1492	0,0101	15,2434	13,0665	0,0069	10,3848	12,9829	0,0068	-1,7586
DSC00040.JPG	768x1024	36,3081	36,1032	0,0121	-16,9264	36,2209	0,0094	-9,3187	36,2085	0,0032	-30,6889
DSC00047.JPG	1024x768	33,1865	32,9195	0,0127	-21,0184	32,8517	0,0079	-42,4666	33,0520	0,0190	-7,0778
DSC00080.JPG	1024x768	22,5273	22,5975	0,0115	6,1177	22,5894	0,0273	2,2763	22,4841	0,0029	-15,1461
DSC00852.JPG	1024x768	47,8120	45,0546	0,0113	-244,5466	46,6063	0,0060	-201,0679	46,9344	0,0096	-90,9915
DSC00861.JPG	1024x768	25,2722	25,2750	0,0111	0,2487	25,1439	0,0053	-24,1770	25,2119	0,0258	-2,3407
DSC00901.JPG	768x1024	26,5532	26,4784	0,0078	-9,6052	26,2520	0,0114	-26,3353	26,4322	0,0271	-4,4596
DSC00911.JPG	768x1024	29,9510	30,0804	0,0109	11,9117	29,9969	0,0158	2,9007	29,9011	0,0258	-1,9353
DSC01263.JPG	1024x768	13,2266	13,3032	0,0243	3,1517	13,2571	0,0078	3,8946	13,2044	0,0110	-2,0148

Табл. 3: Резултати от прилагането на филтри върху изображението - медианен филтър

Изображение	Размер	Оригинал (o)	JPEG 70			JPEG 80			JPEG 90		
			μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ
DSC01269.JPG	1024x768	30,6721	30,8324	0,0084	19,0161	30,7940	0,0334	3,6480	30,6411	0,0042	-7,3497
DSC01432.JPG	1024x768	37,4014	37,5043	0,0109	9,4782	37,4109	0,0173	0,5512	37,3432	0,0298	-1,9570
DSC01443.JPG	1024x768	47,5705	47,4587	0,0081	-13,7444	47,4318	0,0124	-11,2054	47,4201	0,0022	-67,7998
DSC01481.JPG	768x1024	24,2518	24,1378	0,0147	-7,7461	24,1611	0,0095	-9,5232	24,1814	0,0027	-25,6689
DSC01534.JPG	1024x768	43,7391	43,7082	0,0160	-1,9284	43,6161	0,0086	-14,3409	43,6808	0,0183	-3,1824
DSC02037.JPG	1000x750	38,9009	38,9366	0,0093	3,8242	38,8970	0,0153	-0,2532	38,8777	0,0200	-1,1629
DSC02087.JPG	750x1000	35,1265	34,4195	0,0067	-105,3632	34,5507	0,0196	-29,3146	34,6465	0,0034	-140,6439
Flowers.bmp	1043x474	36,9337	37,0670	0,0125	10,6844	36,9995	0,0089	7,3595	36,9155	0,0077	-2,3669
FRUIT.BMP	512x512	30,1818	30,3822	0,0224	8,9399	30,2722	0,0241	3,7497	30,1742	0,0114	-0,6726
Fruit.color.bmp	487x414	41,3179	40,5622	0,0310	-24,3978	40,8845	0,0156	-27,8207	41,0830	0,0132	-17,7432
fruits.bmp	512x512	29,9510	29,6547	0,0270	-10,9692	29,7198	0,0155	-14,9231	29,7863	0,0074	-22,3648
geom2.bmp	256x256	156,6279	145,4847	0,3765	-29,5969	149,2858	0,1775	-41,3553	153,7849	0,1690	-16,8233
girl.png	768x512	32,3492	31,7104	0,0216	-29,6349	31,8365	0,0204	-25,1299	32,0688	0,0043	-65,0183
goldhill.bmp	512x512	30,9478	31,1252	0,0256	6,9273	31,0269	0,0274	2,8873	30,9353	0,0090	-1,3878
goldhill256.jpg	256x256	36,2445	36,4064	0,0649	2,4957	36,2702	0,0447	0,5727	36,1908	0,0275	-1,9527
kai.bmp	256x256	59,5912	58,1990	0,0626	-22,2260	58,5828	0,0518	-19,4576	59,0338	0,0166	-33,6586
kaplan1.bmp	360x624	59,4356	58,7491	0,0551	-12,4684	58,8954	0,0244	-22,1154	59,0907	0,0064	-53,9910
Lena256.bmp	256x256	47,8318	47,9836	0,0533	2,8464	47,8772	0,0404	1,1240	47,7945	0,0191	-1,9578
lena512.bmp	512x512	36,6176	36,6921	0,0222	3,3529	36,6717	0,0142	3,8099	36,6111	0,0058	-1,1206
Leonardo.bmp	695x1040	31,2886	31,2784	0,0116	-0,8786	31,2410	0,0081	-5,8481	31,2054	0,0029	-28,8253
Mouse.bmp	936x1035	55,0130	52,7937	0,0142	-156,1581	53,2860	0,0046	-378,2618	54,0433	0,0054	-179,1384
MRI.bmp	256x256	53,4129	53,4054	0,0400	-0,1869	53,4459	0,0330	1,0001	53,3846	0,0167	-1,6950
Niagara.bmp	399x450	33,9151	33,7605	0,0413	-3,7433	33,7810	0,0243	-5,5089	33,8308	0,0192	-4,3899
P1.bmp	345x406	43,5336	39,8541	0,0592	-62,1867	40,7037	0,0521	-54,3080	42,2945	0,0275	-45,0023
P10.bmp	457x507	19,5227	19,1601	0,0238	-15,2500	19,1995	0,0258	-12,5082	19,3374	0,0082	-22,7392
P2.bmp	274x406	81,6255	78,0241	0,0731	-49,2960	79,3434	0,0349	-65,3486	80,9214	0,0294	-23,9158
P3.bmp	406x433	41,5788	39,3923	0,0343	-63,7709	39,9643	0,0353	-45,7632	40,8687	0,0212	-33,4223
P4.bmp	406x433	45,6577	44,8792	0,0568	-13,6957	45,0343	0,0328	-18,9955	45,3045	0,0159	-22,1850
P5.bmp	323x418	23,8471	22,9897	0,0501	-17,1136	23,2826	0,0656	-8,6032	23,4237	0,0240	-17,6282
P6.bmp	487x368	52,5246	51,3682	0,0547	-21,1244	51,6905	0,0162	-51,5439	52,0788	0,0158	-28,2237
P7.bmp	442x554	21,6333	21,7559	0,0374	3,2755	21,7008	0,0151	4,4742	21,6274	0,0088	-0,6704
P8.bmp	267x347	35,8475	35,4007	0,0665	-6,7187	35,6427	0,0312	-6,5678	35,8085	0,0080	-4,8901
P9.bmp	454x323	20,1289	20,2291	0,0437	2,2927	20,2035	0,0225	3,3114	20,1184	0,0126	-0,8340
pentagon.tiff	1024x1024	27,2319	27,4027	0,0075	22,6292	27,3238	0,0072	12,8033	27,2321	0,0020	0,0525
Peppers.bmp	512x512	28,6080	28,7010	0,0283	3,2848	28,6653	0,0180	3,1820	28,6051	0,0097	-0,2966
pepperscolour.png	512x512	34,7573	34,1685	0,0226	-26,0863	34,2347	0,0173	-30,1672	34,3380	0,0083	-50,7282
Picasso.bmp	1618x2064	17,6836	17,9662	0,0047	60,0372	17,7801	0,0014	66,8374	17,6821	0,0007	-2,0485
pool.bmp	510x383	19,9751	20,0064	0,0253	1,2397	19,9880	0,0300	0,4305	19,9466	0,0100	-2,8442
pool.png	510x383	21,3692	21,1984	0,0283	-6,0334	21,2957	0,0237	-3,1033	21,3268	0,0088	-4,8380

Табл. 3: Резултати от прилагането на филтри върху изображението - медианен филтър

Изображение	Размер	Оригинал (o)	JPEG 70			JPEG 80			JPEG 90		
			μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ
thoraxl.bmp	280x301	15,2145	15,3923	0,0402	4,4223	15,3492	0,0371	3,6331	15,2205	0,0106	0,5677
watch.bmp	1024x768	28,6851	28,6691	0,0082	-1,9483	28,6528	0,0027	-11,8992	28,6461	0,0039	-10,1038
watch.png	1024x768	29,8387	29,7261	0,0057	-19,6019	29,7382	0,0052	-19,2706	29,7584	0,0061	-13,0761
water.png	1024x768	35,2525	34,7687	0,0127	-38,2200	34,8509	0,0095	-42,3983	34,9520	0,0029	-102,7949
zelda256.bmp	256x256	31,1788	31,3347	0,0586	2,6615	31,2807	0,0452	2,2566	31,1524	0,0142	-1,8536
zelda512.jpg	512x512	19,4150	19,5703	0,0312	4,9710	19,4669	0,0343	1,5145	19,4066	0,0163	-0,5120

Табл. 4: Резултати от прилагането на филтри върху изображението - Винер-Колмогоров филтър

Изображение	Размер	Оригинал (o)	JPEG 70			JPEG 80			JPEG 90		
			μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ
airplane.png	512x512	517,483	307,545	14,192	-14,793	381,113	25,224	-5,406	488,453	16,597	-1,749
arctichare.png	594x400	345,632	192,973	32,256	-4,733	264,779	35,274	-2,292	406,975	121,925	0,503
baboon.bmp	512x512	73,692	70,701	0,773	-3,868	71,906	0,695	-2,567	73,924	0,580	0,401
baboon.png	512x512	62,712	61,709	0,696	-1,441	61,638	0,331	-3,241	62,626	0,157	-0,546
barbara.bmp	512x512	153,653	128,459	4,064	-6,200	140,049	3,025	-4,497	153,350	2,134	-0,142
barbara.gif	512x512	133,714	123,763	6,758	-1,472	129,953	3,499	-1,075	152,552	9,691	1,944
bates.bmp	487x727	500,692	383,162	19,457	-6,040	450,482	18,624	-2,696	500,568	2,599	-0,048
bird256gray.bmp	256x256	506,786	290,313	10,265	-21,088	387,470	18,446	-6,468	494,907	14,817	-0,802
boat.bmp	512x512	218,078	159,982	5,353	-10,852	177,507	6,017	-6,743	212,879	4,707	-1,104
boat256.jpg	256x256	221,040	178,599	4,410	-9,624	198,631	6,384	-3,510	229,937	13,710	0,649
bridge256.bmp	256x256	87,545	86,321	0,625	-1,957	87,081	0,564	-0,823	87,414	0,148	-0,883
bridge256.jpg	256x256	86,688	85,259	0,601	-2,378	86,036	0,415	-1,568	86,519	0,129	-1,314
Bw1.bmp	267x234	992,256	1290,793	73,029	4,088	2347,570	109,886	12,334	4029,438	1046,912	2,901
Bw3.bmp	296x295	84,498	2132,645	105,310	19,449	3743,804	342,118	10,696	6243,265	2000,751	3,078
bw4.bmp	470x221	440,831	2703,926	242,415	9,336	5000,046	340,841	13,376	7667,157	2513,779	2,875
Bw7.bmp	513x389	200,932	2659,155	343,995	7,146	5398,921	373,078	13,933	6423,663	2477,459	2,512
camera256.bmp	256x256	875,180	512,665	21,250	-17,060	669,211	31,356	-6,569	828,027	20,512	-2,299
cat.png	490x733	261,112	192,727	4,905	-13,943	205,392	7,605	-7,326	248,723	11,357	-1,091
DSC00001.JPG	768x1024	187,552	154,901	1,568	-20,827	180,329	1,492	-4,840	195,525	6,375	1,251
DSC00007.JPG	1024x768	93,232	113,405	10,666	1,891	172,967	19,047	4,186	153,564	46,808	1,289
DSC00014.JPG	1024x768	357,332	224,424	2,448	-54,294	272,090	4,243	-20,089	353,470	2,691	-1,435
DSC00016.JPG	768x1024	111,279	49,564	1,231	-50,133	65,008	3,258	-14,200	103,344	4,872	-1,629
DSC00040.JPG	768x1024	767,264	392,309	20,314	-18,458	518,431	12,925	-19,252	729,524	22,161	-1,703
DSC00047.JPG	1024x768	343,933	219,420	6,316	-19,714	296,497	12,957	-3,661	379,317	11,923	2,968
DSC00080.JPG	1024x768	143,712	81,403	0,997	-62,521	94,620	1,555	-31,563	141,510	1,605	-1,372
DSC00852.JPG	1024x768	360,182	221,800	21,235	-6,517	422,757	19,776	3,164	734,834	29,029	12,906
DSC00861.JPG	1024x768	191,995	120,849	3,161	-22,509	160,264	2,374	-13,363	192,122	2,794	0,045
DSC00901.JPG	768x1024	77,454	66,490	5,353	-2,048	99,032	8,250	2,616	99,257	12,646	1,724

Табл. 4: Резултати от прилагането на филтри върху изображението - Винер-Колмогоров филтър

Изображение	Размер	Оригинал (o)	JPEG 70			JPEG 80			JPEG 90		
			μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ
DSC00911.JPG	768x1024	45,803	45,478	0,092	-3,517	45,650	0,051	-2,960	45,787	0,014	-1,160
DSC01263.JPG	1024x768	92,332	51,718	1,375	-29,547	66,769	2,321	-11,012	92,583	0,732	0,343
DSC01269.JPG	1024x768	80,648	77,782	0,160	-17,929	78,546	0,282	-7,451	80,430	0,086	-2,534
DSC01432.JPG	1024x768	52,380	52,234	0,152	-0,958	52,295	0,073	-1,154	52,422	0,062	0,687
DSC01443.JPG	1024x768	92,319	87,414	1,906	-2,574	88,677	1,190	-3,061	96,542	3,307	1,277
DSC01481.JPG	768x1024	102,946	86,282	1,118	-14,899	92,929	0,337	-29,757	102,197	0,333	-2,250
DSC01534.JPG	1024x768	184,142	179,385	2,811	-1,692	174,719	1,024	-9,203	184,985	1,734	0,486
DSC02037.JPG	1000x750	404,644	238,892	6,819	-24,309	314,987	14,792	-6,061	400,177	5,170	-0,864
DSC02087.JPG	750x1000	446,823	270,894	11,056	-15,913	331,892	4,423	-25,982	422,880	10,014	-2,391
Flowers.bmp	1043x474	132,529	75,496	2,171	-26,265	79,901	4,819	-10,922	125,664	15,817	-0,434
FRUIT.BMP	512x512	143,236	113,601	4,339	-6,829	128,988	5,543	-2,571	141,311	1,893	-1,017
Fruit.color.bmp	487x414	185,039	152,022	1,647	-20,046	168,896	4,438	-3,637	183,323	3,346	-0,513
fruits.bmp	512x512	173,176	138,074	5,915	-5,935	154,229	5,454	-3,474	169,845	2,278	-1,462
geom2.bmp	256x256	467,948	427,689	17,158	-2,346	442,210	10,296	-2,500	457,268	6,082	-1,756
girl.png	768x512	670,580	274,077	10,715	-37,003	382,192	31,267	-9,223	557,614	32,261	-3,502
goldhill.bmp	512x512	159,385	106,734	5,726	-9,196	127,510	5,106	-6,243	147,734	3,230	-3,607
goldhill256.jpg	256x256	206,243	144,426	8,871	-6,968	179,216	10,464	-2,583	242,538	28,858	1,258
kai.bmp	256x256	391,580	306,091	18,968	-4,507	329,350	13,751	-4,525	377,009	24,900	-0,585
kaplan1.bmp	360x624	986,188	683,227	45,537	-6,653	834,321	38,060	-3,990	988,164	8,930	0,221
Lena256.bmp	256x256	240,248	178,062	11,725	-5,304	190,936	7,986	-6,175	213,892	10,613	-2,483
lena512.bmp	512x512	223,207	194,336	6,109	-4,726	211,431	4,634	-2,541	224,603	2,926	0,477
Leonardo.bmp	695x1040	71,450	69,430	0,415	-4,868	70,132	0,361	-3,650	71,224	0,129	-1,753
Mouse.bmp	936x1035	358,529	509,166	74,047	2,034	959,274	143,158	4,196	777,590	204,786	2,046
MRI.bmp	256x256	291,419	598,888	25,856	11,892	1025,528	57,199	12,834	1620,026	561,821	2,365
Niagara.bmp	399x450	323,810	169,769	5,481	-28,104	189,030	7,233	-18,635	276,970	19,447	-2,409
P1.bmp	345x406	437,220	568,666	20,598	6,382	553,518	279,459	0,416	1299,694	464,391	1,857
P10.bmp	457x507	203,150	94,501	2,830	-38,395	103,141	9,735	-10,273	172,057	16,302	-1,907
P2.bmp	274x406	140,959	147,293	0,897	7,060	146,477	1,345	4,103	141,424	0,522	0,893
P3.bmp	406x433	236,400	235,096	16,737	-0,078	250,481	54,268	0,259	420,153	142,166	1,293
P4.bmp	406x433	729,250	273,543	11,877	-38,369	290,712	73,724	-5,948	560,053	102,669	-1,648
P5.bmp	323x418	198,214	178,466	7,978	-2,475	177,712	83,124	-0,247	385,467	139,568	1,342
P6.bmp	487x368	565,294	370,743	23,496	-8,280	475,387	23,254	-3,866	556,003	14,073	-0,660
P7.bmp	442x554	149,917	102,252	4,362	-10,927	123,335	6,527	-4,073	150,775	3,808	0,225
P8.bmp	267x347	96,206	92,331	2,992	-1,296	94,131	2,423	-0,857	96,617	0,647	0,635
P9.bmp	454x323	39,793	37,162	0,892	-2,949	37,760	0,549	-3,706	39,381	0,563	-0,730
pentagon.tiff	1024x1024	58,645	56,450	0,359	-6,123	58,162	0,357	-1,353	58,767	0,113	1,083
Peppers.bmp	512x512	149,795	128,113	4,325	-5,013	143,162	8,020	-0,827	148,011	1,893	-0,942
pepperscolour.png	512x512	185,072	158,811	3,950	-6,648	172,375	4,018	-3,160	180,346	1,289	-3,666
Picasso.bmp	1618x2064	40,026	38,578	0,013	-107,978	39,888	0,010	-13,688	40,028	0,006	0,341

Табл. 4: Резултати от прилагането на филтри върху изображението - Винер-Колмогоров филтър

Изображение	Размер	Оригинал (o)	JPEG 70			JPEG 80			JPEG 90		
			μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ
pool.bmp	510x383	534,439	142,704	5,392	-72,651	222,050	14,968	-20,870	457,691	40,504	-1,895
pool.png	510x383	577,103	163,033	8,374	-49,448	246,061	11,119	-29,772	486,561	26,574	-3,407
thoraxl.bmp	280x301	105,174	78,431	2,574	-10,390	88,674	2,339	-7,055	99,600	3,657	-1,524
watch.bmp	1024x768	692,622	169,368	14,651	-35,715	287,599	14,726	-27,504	673,278	12,642	-1,530
watch.png	1024x768	639,270	175,841	17,299	-26,790	295,432	19,085	-18,016	661,456	12,111	1,832
water.png	1024x768	84,124	78,341	0,377	-15,357	80,358	0,241	-15,647	82,848	0,119	-10,733
zelda256.bmp	256x256	110,280	77,854	3,799	-8,534	96,233	11,118	-1,263	106,644	6,795	-0,535
zelda512.jpg	512x512	66,025	58,807	1,191	-6,059	62,602	1,393	-2,458	65,628	1,040	-0,381

Табл. 5: Резултати от прилагането на филтри върху изображението - Лапласиан на Гаусиана

Изображение	Размер	Оригинал (o)	JPEG 70			JPEG 80			JPEG 90		
			μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ
airplane.png	512x512	352,1359	353,1536	0,1654	6,1518	352,7955	0,1123	5,8747	352,0437	0,0638	-1,4442
arctichare.png	594x400	268,2487	263,3544	0,1773	-27,6125	263,9053	0,1533	-28,3359	264,8590	0,0736	-46,0353
baboon.bmp	512x512	276,1198	277,8343	0,1225	14,0004	277,0693	0,0930	10,2097	276,0557	0,0590	-1,0849
baboon.png	512x512	280,0724	281,6824	0,1968	8,1819	280,9020	0,0835	9,9328	279,9819	0,0801	-1,1304
barbara.bmp	512x512	289,3156	290,4446	0,1381	8,1745	289,9802	0,1073	6,1966	289,3139	0,0264	-0,0638
barbara.gif	512x512	289,3902	290,8183	0,1746	8,1798	290,2457	0,2068	4,1375	289,4943	0,0823	1,2640
bates.bmp	487x727	318,7891	319,5694	0,0852	9,1632	319,3071	0,0692	7,4849	318,7478	0,0510	-0,8079
bird256gray.bmp	256x256	269,1086	270,4255	0,4308	3,0569	269,9107	0,3025	2,6513	269,0351	0,1033	-0,7119
boat.bmp	512x512	305,5775	306,8884	0,1446	9,0674	306,4558	0,0977	8,9872	305,5127	0,0525	-1,2349
boat256.jpg	256x256	342,7114	343,3315	0,3422	1,8121	343,0724	0,3318	1,0879	342,2831	0,2253	-1,9010
bridge256.bmp	256x256	378,1024	377,8573	0,3624	-0,6763	378,5817	0,2776	1,7265	377,6940	0,1126	-3,6279
bridge256.jpg	256x256	378,1822	378,5943	0,4518	0,9122	378,8310	0,2997	2,1647	377,9078	0,0792	-3,4663
Bw1.bmp	267x234	513,6057	446,1356	2,3953	-28,1681	468,7074	1,3196	-34,0230	493,9513	1,0409	-18,8824
Bw3.bmp	296x295	789,6620	665,1972	1,4192	-87,7031	705,5547	1,0762	-78,1540	755,0728	0,9460	-36,5633
bw4.bmp	470x221	1072,4301	908,5997	2,2361	-73,2661	968,3126	0,9846	-105,7419	1029,3872	1,7343	-24,8191
Bw7.bmp	513x389	1053,6943	892,3682	1,0357	-155,7623	952,9215	1,0084	-99,9316	1011,1041	1,6145	-26,3797
camera256.bmp	256x256	411,0285	406,4157	0,3436	-13,4235	410,6962	0,2208	-1,5049	410,5547	0,1135	-4,1732
cat.png	490x733	447,4663	448,1981	0,1085	6,7427	447,9565	0,1225	4,0033	447,2222	0,0396	-6,1649
DSC00001.JPG	768x1024	262,7087	262,7485	0,0747	0,5326	263,3842	0,1136	5,9482	262,4714	0,0170	-13,9674
DSC00007.JPG	1024x768	243,3120	234,8167	0,0831	-102,2329	235,2733	0,0234	-344,0183	239,5555	0,1206	-31,1485
DSC00014.JPG	1024x768	243,3386	244,4910	0,0672	17,1377	244,1499	0,1270	6,3871	243,2093	0,0206	-6,2832
DSC00016.JPG	768x1024	107,6825	108,8229	0,1240	9,1968	108,2659	0,0536	10,8924	107,5826	0,0433	-2,3063
DSC00040.JPG	768x1024	307,1682	305,4496	0,0810	-21,2208	306,5338	0,0978	-6,4850	306,3274	0,0162	-51,7486
DSC00047.JPG	1024x768	286,0032	283,3963	0,0987	-26,4208	282,9238	0,0559	-55,0472	284,8261	0,1719	-6,8488
DSC00080.JPG	1024x768	184,7352	185,4545	0,1229	5,8540	185,3653	0,2181	2,8888	184,4241	0,0260	-11,9832
DSC00852.JPG	1024x768	409,8753	386,1290	0,0762	-311,5717	399,6332	0,0518	-197,8847	402,4090	0,0926	-80,6023

Табл. 5: Резултати от прилагането на филтри върху изображението - Лапласиан на Гаусиана

Изображение	Размер	Оригинал (o)	JPEG 70			JPEG 80			JPEG 90		
			μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ
DSC00861.JPG	1024x768	217,9294	217,8784	0,1269	-0,4018	216,7628	0,0453	-25,7400	217,4033	0,2133	-2,4663
DSC00901.JPG	768x1024	228,1275	227,3925	0,0687	-10,7060	225,4771	0,0871	-30,4136	227,1277	0,2256	-4,4329
DSC00911.JPG	768x1024	257,5541	258,5851	0,0875	11,7825	257,7992	0,1123	2,1821	257,1665	0,2100	-1,8454
DSC01263.JPG	1024x768	109,6692	110,1158	0,2316	1,9282	109,7862	0,0601	1,9464	109,4545	0,0971	-2,2118
DSC01269.JPG	1024x768	276,4153	277,7207	0,0415	31,4228	277,4089	0,2963	3,3538	276,1052	0,0190	-16,3583
DSC01432.JPG	1024x768	322,1480	322,8551	0,1057	6,6867	322,2072	0,1302	0,4553	321,6636	0,2460	-1,9689
DSC01443.JPG	1024x768	411,0775	410,0246	0,0337	-31,2179	409,8335	0,1142	-10,8892	409,7881	0,0124	-103,7052
DSC01481.JPG	768x1024	211,6443	210,3998	0,1213	-10,2601	210,7017	0,0903	-10,4351	210,9420	0,0181	-38,8354
DSC01534.JPG	1024x768	382,7303	382,4355	0,0964	-3,0590	381,6996	0,0558	-18,4635	382,2552	0,1675	-2,8366
DSC02037.JPG	1000x750	331,5724	331,8185	0,0386	6,3845	331,4728	0,1104	-0,9023	331,3497	0,1933	-1,1525
DSC02087.JPG	750x1000	293,3061	287,2591	0,0826	-73,1645	288,4545	0,1508	-32,1663	289,2120	0,0211	-193,9503
Flowers.bmp	1043x474	313,0850	314,3170	0,0821	15,0076	313,7033	0,0599	10,3194	312,9464	0,0519	-2,6741
FRUIT.BMP	512x512	251,1464	252,6366	0,1990	7,4868	251,9068	0,1308	5,8117	251,0977	0,0764	-0,6372
Fruit.color.bmp	487x414	353,8840	347,3963	0,2391	-27,1333	350,2125	0,0982	-37,3841	351,9411	0,0758	-25,6453
fruits.bmp	512x512	255,1961	252,7173	0,2153	-11,5125	253,2328	0,1179	-16,6536	253,7435	0,0528	-27,4938
geom2.bmp	256x256	1288,6402	1198,4416	2,9296	-30,7886	1228,5431	1,2749	-47,1379	1265,5805	1,3634	-16,9137
girl.png	768x512	242,4545	238,2789	0,1428	-29,2347	239,1143	0,1212	-27,5679	240,2723	0,0316	-69,0737
goldhill.bmp	512x512	257,2247	258,6477	0,1700	8,3717	257,9783	0,2135	3,5297	257,1620	0,0391	-1,6045
goldhill256.jpg	256x256	311,5352	312,6309	0,4684	2,3391	311,9421	0,2749	1,4800	311,1298	0,2152	-1,8842
kai.bmp	256x256	501,6639	490,1747	0,4240	-27,0940	493,4506	0,3669	-22,3872	496,9030	0,0962	-49,4779
kaplan1.bmp	360x624	460,0751	455,2209	0,2175	-22,3216	456,3303	0,1110	-33,7465	457,5702	0,0633	-39,5870
Lena256.bmp	256x256	391,9212	393,2655	0,3739	3,5949	392,5287	0,2442	2,4874	391,7881	0,1234	-1,0785
lena512.bmp	512x512	306,3328	307,1746	0,1800	4,6760	306,8305	0,0768	6,4781	306,2967	0,0373	-0,9689
Leonardo.bmp	695x1040	224,4394	225,4696	0,1000	10,2968	224,8821	0,0800	5,5343	224,2159	0,0180	-12,4105
Mouse.bmp	936x1035	466,8205	448,6930	0,0554	-327,1545	452,6344	0,0292	-485,4623	458,7722	0,0145	-556,6590
MRI.bmp	256x256	467,1289	467,1439	0,2916	0,0517	467,4810	0,1829	1,9252	466,9860	0,1155	-1,2377
Niagara.bmp	399x450	242,1419	241,5842	0,1931	-2,8878	241,7380	0,1435	-2,8151	241,7287	0,0648	-6,3805
P1.bmp	345x406	366,4568	336,7610	0,3483	-85,2591	343,9218	0,2934	-76,8046	356,3899	0,2475	-40,6763
P10.bmp	457x507	165,0369	161,6227	0,2540	-13,4393	162,1635	0,1888	-15,2231	163,4044	0,0521	-31,3299
P2.bmp	274x406	683,2887	654,1294	0,5170	-56,4024	665,2261	0,3141	-57,5112	677,6746	0,2110	-26,6074
P3.bmp	406x433	360,4507	341,4530	0,2118	-89,6930	346,3242	0,1903	-74,2518	354,1643	0,1163	-54,0394
P4.bmp	406x433	381,9751	376,1763	0,2626	-22,0834	377,5364	0,3040	-14,5994	379,2621	0,1089	-24,9207
P5.bmp	323x418	201,8389	194,0942	0,2972	-26,0580	196,5550	0,4868	-10,8547	197,9772	0,1532	-25,2109
P6.bmp	487x368	428,6078	419,1877	0,3361	-28,0279	421,4396	0,1097	-65,3692	424,9327	0,1158	-31,7256
P7.bmp	442x554	188,3097	189,4350	0,3279	3,4318	188,8705	0,1296	4,3267	188,2013	0,0773	-1,4034
P8.bmp	267x347	309,3532	305,3096	0,5318	-7,6042	307,4104	0,1934	-10,0437	308,9388	0,0699	-5,9268
P9.bmp	454x323	168,7182	169,4323	0,3260	2,1908	169,2640	0,1864	2,9287	168,6314	0,0750	-1,1578
pentagon.tiff	1024x1024	235,5981	237,0492	0,0457	31,7599	236,3824	0,0645	12,1598	235,5993	0,0122	0,0990
Peppers.bmp	512x512	242,1184	243,2273	0,2193	5,0562	242,7931	0,1273	5,3011	242,1326	0,0476	0,2986

Табл. 5: Резултати от прилагането на филтри върху изображението - Лапласиан на Гаусиана

Изображение	Размер	Оригинал (o)	JPEG 70			JPEG 80			JPEG 90		
			μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ	μ	σ	(μ-o)/σ
pepperscolour.png	512x512	294,5461	289,6772	0,1227	-39,6833	290,0333	0,1348	-33,4724	290,7293	0,0786	-48,5472
Picasso.bmp	1618x2064	155,7675	157,6974	0,0179	107,8147	156,6140	0,0154	54,7985	155,7599	0,0089	-0,8413
pool.bmp	510x383	169,3028	169,7447	0,2992	1,4773	169,5111	0,2913	0,7151	169,1116	0,0547	-3,4982
pool.png	510x383	180,1682	178,9328	0,2434	-5,0749	179,6388	0,1488	-3,5585	179,8623	0,0782	-3,9125
thoraxl.bmp	280x301	159,1120	159,9997	0,1882	4,7158	159,5704	0,2873	1,5955	158,9922	0,1132	-1,0581
watch.bmp	1024x768	244,1580	244,1150	0,0926	-0,4646	244,0271	0,0526	-2,4889	243,8687	0,0195	-14,8315
watch.png	1024x768	253,7475	252,8731	0,0492	-17,7682	253,0329	0,0622	-11,4953	253,0987	0,0168	-38,6145
water.png	1024x768	303,3588	298,9587	0,0825	-53,3470	299,7649	0,0640	-56,1417	300,7241	0,0236	-111,4236
zelda256.bmp	256x256	266,5668	268,2250	0,3158	5,2507	267,4839	0,2345	3,9101	266,4352	0,0733	-1,7964
zelda512.jpg	512x512	169,5751	170,7682	0,3631	3,2861	170,0716	0,3272	1,5171	169,5156	0,1347	-0,4417