УДК 004.932+004.056.55+004.421.5

КОДИРОВАНИЕ ВИДЕО СО СКРЕМБЛИРОВАНИЕМ

Н.А. ЛАВРИНОВИЧ

Белорусский государственный университет информатики и радиоэлектроники
П. Бровки, 6, Минск, 220013, Беларусь

Поступила в редакцию 27 февраля 2013

Приведены результаты исследования алгоритмов кодирования видеоинформации семейства MPEG со встроенными механизмами защиты от несанкционированного доступа. Представлена разработка оригинального метода защиты видео путем скремблирования при кодировании исходного видеопотока по стандарту H.264\AVC (MPEG 4 Part 10) с использованием случайной числовой последовательности большого периода.

Ключевые слова: защита информации, H.264, скремблирование, MPEG.

Введение

В современно мире технологий, где важнейшими ресурсом и силой является информация, необходимо иметь средства для предотвращения несанкционированного доступа к данным. Видеоконференции, спутниковое и интернет телевидение – все эти приложения не могут быть успешны и прибыльны без применения защиты в том или ином виде. Все большую важность на территории стран СНГ приобретает задача соблюдения авторских прав, которая также требует инженерных решений по конфиденциальной передаче информации, для которых, в свою очередь, актуальна проблема минимизации объема передаваемых данных и требуемых для работы вычислительных ресурсов.

Исследовния, проведенные в рамках данной работы, посвящены вопросам защиты видеоинформации. Особое внимание уделено видеокодекам семейства MPEG, в частности H.264\AVC, как наиболее распространенным и востребованным.

Методы сокрытия видеоинформации

Видеоданные представляют собой это набор битов, которые определенным образом структурированы. Самый простой подход к защите видеоданных − это использование классического шифрования по схемам с открытым или закрытым ключом, например использование блочного алгоритма AES. Объем шифруемых видеоданных по сравнению с текстовыми и даже звуковыми данными значительно больше и требует больших вычислительных ресурсов для обработки. Это приводит к ограничению возможности использования классического шифрования в таких областях, как, например, интернет телевидение, поскольку пользователи таких сервисов должны иметь мощную систему, способную в реальном времени, без задержек расшифровывать, а затем декодировать и отображать полученное видео.

Поточные шифры могут быть очень быстрыми в работе. Шифрующая последовательность, или гамма, состоит из бит, выбираемых случайным образом, и побитно накладывается на открытый текст. Гамма имеет ту же длину, что и открытый текст, но может вырабатываться до процесса шифрования или расшифровки, которому остается лишь единственная легкая операция наложения [1].

Перечислим способы защиты данных для алгоритмов сжатия семейства MPEG.

Алгоритм, предложенный Тангом [2], так называемый алгоритм перестановки «зигзаг», заключается в считывании квантованных коэффициентов дискретного косинусного преобразования (ДКП) не способом «зигзаг» для последующего кодирования, как это определено в формате, а случайным образом. Алгоритм Танга неустойчив к криптоатакам с использованием как открытого текста, так и только шифротекста. При наличии только шифротекста вскрытие возможно, поскольку коэффициенты, как правило, сосредоточены в верхнем левом углу матрицы, и, зная это, можно найти их исходное расположение. Для усиления защиты, в дополнение к использованию считывания коэффициентов случайным образом, в способе предложены следующие преобразования (рис. 1). Сначала коэффициенты восьми матриц коэффициентов ДКП размером 8×8, находящиеся в верхнем левом углу (так называемые DC-коэффициенты с нулевой частотой), объединяют. Затем коэффициенты шифруют по алгоритму DES и вновь записывают в соответствующие матрицы. После этого каждый 8-битовый DC-коэффициент разбивают на 2 части − старшую и младшую. Старшая часть записывается как сам DC-коэффициент, а младшая часть записывается как коэффициент самой высокой частоты ДКП в матрице.

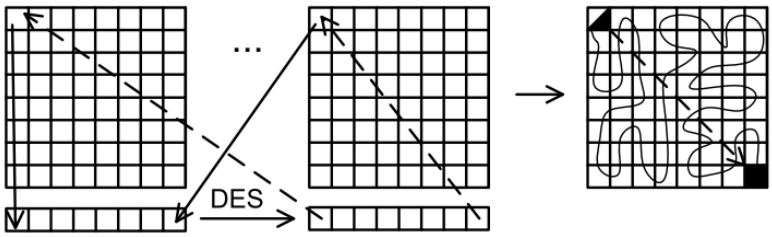

Рис. 1. Усовершенствованный алгоритм «зигзаг»

Ченгом и Ли был предложен метод выборочного шифрования, в котором шифруются только низкочастотные составляющие дискретного косинусного преобразования ДКП [3]. Способ, предложенный Ценгом и Лей [4], предусматривает в каждом сегменте кадра перестановку значений коэффициентов ДКП, занимающих одну и ту же позицию в матрице, с использованием некоторой таблицы правил перестановки. В дополнение к перестановкам коэффициентов внутри каждого сегмента в предложенном способе осуществляется изменение случайным образом знаков коэффициентов ДКП. Кроме перестановки коэффициентов, осуществляется перестановка векторов движения P-кадров, а также знаков этих векторов. Бхаргава и Ши в своей работе [5] предложили выборочно шифровать только биты, отвечающие за знак коэффициентов ДКП в *I*-кадрах, что понижает требования к вычислительным ресурсам, поскольку объем знаковых битов составляет лишь около 13 % от всего объема потока. В дальнейшем Бхаргава и Ши предложили помимо знаковых битов шифровать также и знаковые биты векторов движения в P- и B-кадрах. При этом в I-кадрах предложено шифровать только DC-коэффициенты для еще большего сокращения вычислений. Однако, такой алгоритм уязвим для атак по методу открытого сообщения. Алгоритм RVEA [5] предполагает шифрование знаковых бит всех коэффициентов ДКП I-кадров и векторов движения P- и B-кадров, а шифрование битов осуществляется с помощью стойких, с одной стороны, и относительно ресурсоемких, с другой стороны, алгоритмов DES или IDEA.

Скремблирование видеоинформации при кодировании в стандарте H.264

В разработанном способе защиты видеоинформации использование сложных алгоритмов шифрования не предусматривается. При трудоемком процессе сжатия видео, использование трудоемких алгоритмов шифрования не является быстрым и удобным решением, особенно при необходимости мгновенного просмотра видео авторизованным пользователем, владеющим секретным ключом, т. е. при необходимости сразу расшифровывать, декодировать и отображать видеоинформацию.

Стандарт H.264\AVC предусматривает после каждой итерации кодирование разностных коэффициентов их последующее дискретное косинусное преобразование и квантование. Проблема модификации частотного преобразования заключается в том, что часть

преобразования совмещена с процессом квантования в целях оптимизации производительности. Нарушать такой оптимизированный подход нецелесообразно, учитывая высокую вычислительную сложность самого алгоритма кодирования.

Предложено применить классический вариант скремблирования: побитное наложение шифрующей последовательности, гаммы, на результаты квантования. Для каждого макроблока кодируемого кадра используется свой отрезок шифрующей последовательности, что обеспечивает большую защищенность.

Пусть P − исходная последовательность квантованных коэффициентов ДКП. Определим преобразование T, имеющее точное обратное преобразование T^{-1}, по некоторому симметричному ключу k как: $E_i = T_{i,k}(P_i)$, где E_i – зашифрованная последовательность бит i-го макроблока P_i, T^k_i – шифрующее преобразование, зависящее от начального состояния (ключа) k и номера макроблока i.

Тогда полный процесс преобразования информации, включая функцию дешифрования D, можно представить следующим образом: $D_i(E_i) = T^{-1}_{i,k}(E_i) = T^{-1}_{i,k}(T_{i,k}(P_i)) = P_i$.

Скремблирование предполагает использование тривиальной функции преобразования. Такое преобразование можно определить как $T_{i,k}(P_i) = (g_0 \oplus p_0)\ldots(g_m \oplus p_m)$, где $g_0 \ldots g_m$, $p_0 \ldots p_m$, биты гаммы и элементов макроблока соответственно, а m-размер машинного слова, $\oplus$ – логическая операция побитового исключающего *или*. Конкретные значения битов гаммы зависят от номера итерации (макроблока) и начального состояния (симметричного ключа).

Скремблирование коэффициентов требует большого количества элементов скремблирующей последовательности, непосредственно участвующих в наложении. Это обеспечивает большую защищенность и усложняет криптоаналитику задачу определения первичных данных. Для обратной операции дешифрации последовательность должна быть воспроизводима на клиентской стороне, поэтому был выбран один из алгоритмов генерации псевдослучайных чисел (ГПСЧ).

Из современных ГПСЧ широкое распространение получил так называемый «вихрь Мерсенна», предложенный в 1997 году Мацумото и Нисимурой [6]. Его достоинствами являются колоссальный период ($2^{19937}-1$), равномерное распределение в 623 измерениях (линейный конгруэнтный метод дает более или менее равномерное распределение максимум в 5 измерениях), быстрая генерация случайных чисел (в 2–3 раза быстрее, чем стандартные ГПСЧ, использующие линейный конгруэнтный метод).

Вихрь Мерсенна основывается на свойствах простых чисел Мерсенна и обеспечивает быструю генерацию высококачественных псевдослучайных чисел. Вихрь Мерсенна лишен многих недостатков, присущих другим ГПСЧ, таких как малый период, предсказуемость, легко выявляемая статистическая зависимость. Благодаря большому периоду алгоритма, «вихрь Мерсенна» был выбран для создания случайной последовательности, которая в дальнейшем применяется для кодирования видеоданных.

Приведем описание механизма генерации псевдослучайных чисел согласно выбранному алгоритму. Пусть x обозначает векторы-слова, которые представляют собой w-мерные векторы над полем $F_2 = \{0,1\}$, соответствующие машинному слову размера w. Вихрь Мерсенна генерирует последовательность вектор-слов, которые являются псевдослучайными целыми из диапазона от 0 до 2^{w-1}. Алгоритм основан на следующем рекуррентном выражении:

$x_{k+n} = x_{k+m} \oplus \left(x^u_k \oplus x^l_{k+1}\right)A,\ k = 0, 1, \ldots$, где n – целое, обозначающее степень рекуррентности; m – целое, такое что $1 \le m \le n$; A матрица размера $w \times w$, с элементами из F_2. В правой части x_k^u обозначает старшие $w-r$ бит x_k и x^l_{k+1} младшие r бит x_{k+1}. Вектор $\left(x^u_k \oplus x^l_{k+1}\right)$ является конкатенацией старших $w-r$ бит x_k и младших r бит x_{k+1}.

Возьмем $(x_0, x_1, \ldots, x_{n-1})$ в качестве начального заполнения. Тогда генератор вычислит x_n по рекуррентному выражению при $k = 0$. Полагая $k = 1, 2, \ldots$, генератор вычислит $x_{n+1}, x_{n+2}, \ldots$ Форма матрицы A выбрана из расчета скорости выполнения умножения на A:

$$A = \begin{pmatrix} 0 & 1 & & & \\ 0 & 0 & 1 & & \\ 0 & \cdots & & \ddots & \\ & & & & 1 \\ a_{w-1} & a_{w-2} & & & a_0 \end{pmatrix}.$$

Вычисление xA сводится к побитовым операциям (>> – сдвиг вправо):

$$xA = \begin{cases} x >> 1 & , \ x_0 = 0 , \\ (x >> 1) \oplus a & , \ x_0 = 1 , \end{cases}$$

где

$$x = \left(x^u{}_k \oplus x^l{}_{k+1} \right), k = 0,1,\ldots ,$$
$$a = (a_{w-1}, a_{w-1}, \ldots, a_0),$$
$$x = (x_{w-1}, x_{w-1}, \ldots, x_0).$$

Для улучшения качества генерируемых чисел в целях криптографии предложено подвергать получаемые результаты хешированию. В качестве быстрой и простой функции хеширования использована функция, предложенная Д. Кнутом в своей фундаментальная монографии [7]: $h_{i+1} = ((h_i << 5) \oplus (h_i >> 27) \oplus s_i$, где i – размер целого в байтах, s входное слово, h – выходное слово.

Получаемая псевдослучайная последовательность генерируется одновременно с процессом кодирования видео. Реализация алгоритма такова, что за один раунд генерации создается 623 значения, которые используются в дальнейшем преобразовании. По достижении конца сгенерированного отрезка процедура повторяется.

Шифрующая гамма состоит из целых чисел одной разрядности с машинным словом, при этом для наложения используется только один младший байт из числа, чтобы не достигнуть предела возможного значения ДКП коэффициента и не превысить максимально допустимое количество бит для макроблока. Например, в случае 32-х разрядной системы, одно сгенерированное число используется 4 раза.

В ходе разработки опробованы различные варианты скремблирования преобразованных и квантованных разностных коэффициентов: выборочное и полное скремблирование коэффициентов, изменение только DC коэффициентов закодированного макроблока (DC коэффициенты трансформированные преобразованием Адамара в режиме кодирования INTRA16x16), изменение только AC коэффициентов закодированного макроблока, различные комбинации таких режимов. Общая схема работы шифрования представлена на рис. 2.

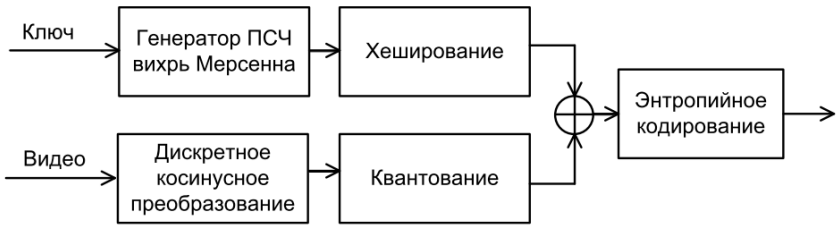

Рис. 2. Общая схема шифрования

Результаты экспериментов

На рис. 3 представлены исходные изображения и наиболее удачные результаты кодирования со скремблированием в различных режимах, в зависимости от того, кодировались

ли только AC или только DC коэффициенты, нулевые либо нет и комбинации таких вариантов. Было проведено 16 экспериментов с различными вариантами скремблирования, только 6 из них дали приемлемые по субъективному наблюдению результаты, из этих 6 только 3 комбинации дали хороший результат искажения для обоих типов изображения.

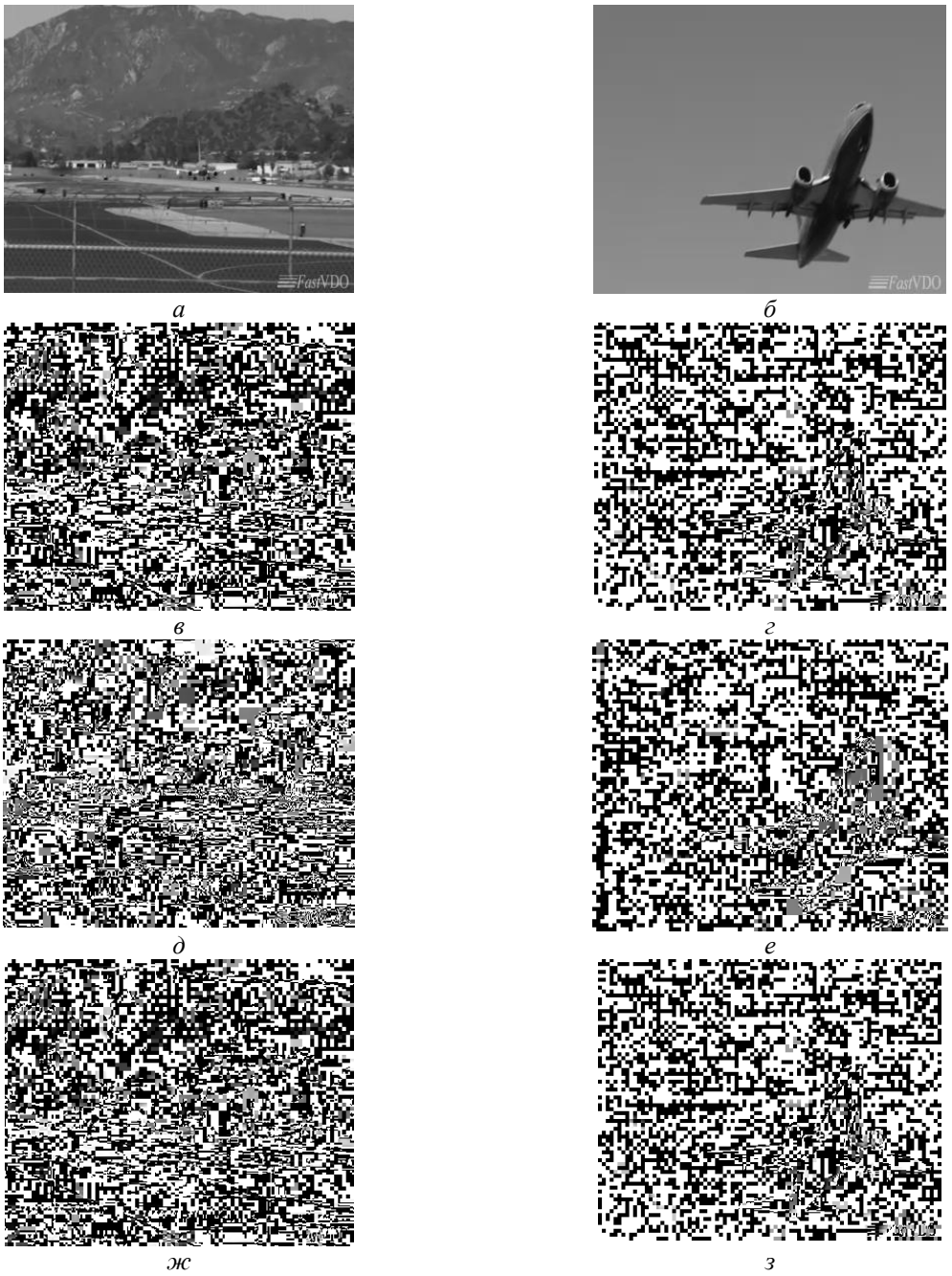

Рис. 3. Результаты кодирования видео со скремблированием в зависимости от режима:
а – исходное изображение (тип 1); *б* – исходное изображение (тип 2);
в – только все DC коэфф. (тип 1); *г* – только все DC коэфф. (тип 2);
д – нулевые DC и ненулевые AC коэфф. (тип 1); е – нулевые DC и ненулевые AC коэфф. (тип 2);
ж – все DC и ненулевые AC коэфф. (тип 1); з –все DC и ненулевые AC коэфф. (тип 2).

В качестве исходной последовательности для проведения экспериментов с различными вариантами шифрования взят видеоролик, имеющий два различных с точки зрения восприятия отрезка. В первой части видео присутствует большое количество деталей, как следствие большое число ненулевых коэффициентов преобразования (тип 1). Во второй части видео присутствует практически неизменный фон с движением контрастного объекта на этом фоне, что влечет за собой появление большого числа нулевых разностных блоков и, как следствие,

большой последовательности повторяющихся данных (тип 2). Такие варианты кадров позволяют определить, насколько хорошо способен алгоритм исказить исходную последовательность изображений и дают возможность сделать вывод о типе видеоинформации, для которой алгоритм подходит наилучшим образом и применим ли он в принципе.

Внедрение скремблирования влияет на качество сжатия информации в худшую сторону. В таблице приведены временные затраты и объем итогового зашифрованного видеофайла в зависимости от режима шифрования.

Временные затраты и объем зашифрованного файла

YUV 4:2:0 352x288 298 кадров. Кодер H.264. (Intel Core i7-2600 @ 3.40 GHz)	Характеристики шифрования		
	Время, с	Объем файла, Мб	Степень сжатия
1. Кодирование без скремблирования коэффициентов	46,375	1,6	27
2. Скремблирование всех DC коэффициентов	46,965	3,8	11
3. Скремблирование нулевых DC коэффициентов и ненулевых AC коэффициентов	47,121	5,5	8
4. Скремблирование всех DC коэффициентов и ненулевых AC коэффициентов	47,426	6,1	7

Заключение

В результате проведения исследований в области защиты видеоинформации, разработан простой и быстродействующий метод скремблирования видеоданных. Механизм, встроенный в кодер распространенного стандарта сжатия видео H.264\AVC (MPEG 4 Part 10), использует шифрующую числовую последовательность, полученную генератором на основе так называемого «вихря Мерсенна». Не внося существенного вклада в производительность видеокодера, не нарушая структуру видеопотока, что важно для систем цифрового вещания и распространения видео по запросу, предложенная методика позволяет достаточно эффективно исказить исходное изображение, сделав его восприятие недоступным, что позволяет решить задачу предотвращения несанкционированного доступа.

VIDEO SCRAMBLING

N.A. LAURYNOVICH

Abstract

Video encryption algorithms of MPEG family with embedded unauthorized access preventing support and proposes an original video data protection technique based on scrambling using long period random sequence when source video sequence is being encoded according to the H.264\AVC (MPEG 4 Part 10) standard is proposed.

Список литературы

1. Поточные шифры. Результаты зарубежной открытой криптологии: сб. науч. трудов. Москва, 1997.
2. *Tang L.* // In Proceedings of the ACM Multimedia. Boston, USA, 1996, P. 219–229.
3. *Yongcheng L., Zhigang C., See-Mong T. et. al.* // IEEE First International Workshop on Multimedia Software Development. Berlin, Germany, 1996. P. 169–175.
4. *Wenjun Z., Shawmin L.* // IEEE Transactions on Multimedia. 2003. P. 118–129.
5. *Bhargava B., Changgui S., Sheng-Yih W.* // Multimedia Tools and Applications. 2004. Vol. 24. P. 57–79.
6. *Matsumoto M., Nishimura T.* // ACM Transactions on Modeling and Computer Simulations. 1998. Vol. 8. P. 3–30.
7. *Дональд Кнут*. Искусство программирования. Москва, 2007.