УДК 004.932

С.Г. Удовенко, А.А. Шамраев, Е.О. Шамраева, С.Д. Лукьяненко

Харьковский национальный университет радиоэлектроники

МОДИФИЦИРОВАННЫЙ МЕТОД ПРЕДИКТИВНОГО КОДИРОВАНИЯ ДЛЯ СЖАТИЯ ГРАФИЧЕСКОЙ ИНФОРМАЦИИ

Усовершенствован алгоритм сжатия изображений без потерь и реализован на языке C++ для сжатия BMP-файлов. Результаты тестирования показали преимущество модифицированного алгоритма над существующими аналогами по степени сжатия при сопоставимой скорости сжатия.

Ключевые слова: *сжатие изображений, bmp, предиктивный метод кодирования, межканальная декорреляция, корректировка знака ошибки.*

Введение

С прогрессом средств вычислительной техники и широким распространением мультимедиа контента всё большая часть информации в вычислительных системах представляется в виде цифровых изображений. Поэтому проблема улучшения алгоритмов сжатия изображений достаточно актуальна. Для уменьшения объема графических данных используют большое число алгоритмов сжатия, к которым предъявляется много жёстких требований, как по объёму сжатого файла, качеству восстановленного изображения, так и по ресурсоёмкости самого алгоритма сжатия. Целью настоящей работы является усовершенствование алгоритма сжатия цифровых изображений на основе метода предиктивного кодирования.

Постановка задачи. Требуется усовершенствовать алгоритм сжатия полноцветных (24 бит) изображений без потерь. Исходные изображения хранятся в формате Windows Bitmap (BMP). Алгоритм должен быть ориентирован на улучшение сжатия фотореалистичных изображений, т.е. изображений, полученных при помощи цифрового либо аналогового фотоаппарата, сканера и др. Алгоритм реализовать в виде консольного Windows-приложения, принимающего в качестве параметров имя исходного и конечного файлов, а также параметр, сообщающий программе о том, какой процесс необходимо выполнить над файлом (сжатие или распаковка). Средняя степень сжатия изображений и среднее время работы программы должны быть сопоставимы с существующими аналогами.

Изложение основного материала

Общая структура алгоритма

За основу схемы работы алгоритма в данной работе взята общая схема сжатия изображений без потерь, включающая ряд модификаций и дополнений (рис. 1).

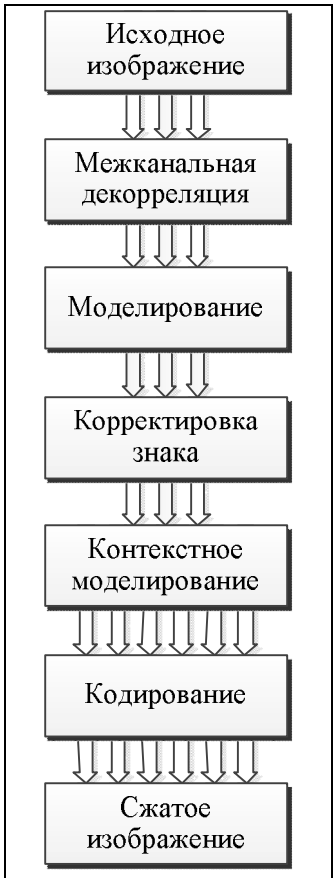

Рис. 1. Общая схема работы алгоритма

Поскольку алгоритм должен работать с полноцветными изображениями, состоящими из 3-х компонент (R, G и B), то необходимо рассматривать поток данных (пикселов) как три параллельных подпотока, для каждой из компонент соответственно. Этап предварительной обработки явно установлен в виде процесса межканальной декорреляции. В схему работы алгоритма также включен дополнительный этап, называемый корректировкой знака, о чём будет сказано ниже. Кроме того, на данной схеме отражена одна из особенностей использования контекстного моделирования, заключающаяся в определенного рода логическом разделении потоков данных, поступающих на вход кодера.

Выбор этапа предварительной обработки

Предварительная обработка изображения, как правило, способствует улучшению характеристик потоков сжимаемых данных, что в свою очередь увеличивает итоговую степень сжатия. В качестве такой обработки обычно выступает процесс межканальной декорреляции, т.е. уменьшения взаимной корреляции между значениями компонент пиксела. Этот процесс может быть реализован при помощи так называемого цветового преобразования, т.е. перевода изображения из одного цветового пространства в другое.

В процессе межкомпонентной декорреляции в модифицированном алгоритме выбрано цветовое пространство YCbCr. Данный выбор был сделан, во-первых, на основании того, что преобразование между пространствами RGB и YCbCr является обратимым, и во-вторых, по результатам многочисленных тестов, результаты которых могут быть приведены отдельно, которые говорят о статистически большей эффективности данного преобразования в сравнении с пространствами RGB, YFbFr и других, при достижении большей степени сжатия.

Выбор модели

За основу модифицированной предиктивной модели был взят существующий предиктор GAP, предложенный в алгоритме CALIC:

$$d_h = |W - WW| + |N - NW| + |N - NE| ;$$

//величина горизонтального градиента

$$d_v = |W - NW| + |N - NN| + |NE - NNE| ;$$

//величина вертикального градиента

IF (d_v - d_h >80) //четкое горизонтальное ребро

$p(i) = W$;

ELSE IF (d_v - d_h <-80) //четкое вертикальное ребро

$p(i) = N$;

ELSE

{

$p(i) = (N + W)/2 + (NE - NW)/4$;

IF (d_v - d_h >32) //горизонтальное ребро

$p(i) = (p(i) + W)/2$;

ELSE IF (d_v - d_h >8) //слабовыраженное горизонтальное ребро

$p(i) = (3p(i) + W)/4$;

ELSE IF (d_v - d_h <-32) //вертикальное ребро

$p(i) = (p(i) + N)/2$;

ELSE IF (d_v - d_h <-8) //слабовыраженное вертикальное ребро

$p(i) = (3p(i) + N)/4$;

}

Данный алгоритм работает по следующему принципу. Если величина вертикального градиента больше горизонтального на некоторое пороговое значение, в данном случае 80, то мы считаем, что в рассматриваемом участке изображения находится четко выраженное горизонтальное ребро, и поэтому предсказываем значение текущего пикселя равным значению левого пикселя. Аналогичным образом, если величина горизонтального градиента больше вертикального на 80, то мы предсказываем значение текущего пикселя равным значению верхнего пикселя.

В противном случае мы предсказываем значение текущего пикселя по формуле (обычный линейный предиктор), а затем, если разница градиентов достигла какого-либо другого порогового значения (32 или 8 в данном случае), то за предсказанное значение мы принимаем среднее взвешенное уже найденного значения и соответствующего пикселя с разными весами.

Данный предиктор был модифицирован для достижения большей степени сжатия двумя способами:

Экспериментально были определены оптимальные значения граничных параметров, которые в оригинальном GAP установлены как 80/32/8. Найденные оптимальные значения равны соответственно 65/15/4.

Введён этап корректировки знака получаемой ошибки, в зависимости от знака текущего градиента. Если градиент отрицателен, то знак выходной ошибки инвертируется. Таким способом частично выравниваются спады на гистограмме распределения ошибок, в результате чего несколько повышается общая степень сжатия.

Выбор кодера. Контекстное моделирование

Основная задача кодера состоит в максимальном использовании тех преимуществ в изменении характеристик потоков сжимаемых данных, которые предоставляет кодеру модель и предобработка. В результате работы кодера должен быть сгенерирован целостный битовый поток, представляющий максимально возможное компактное представление входного потока. Поскольку на этом этапе теоретически минимальный объём данных определяется энтропией входного потока кодера, то целью кодера также является достижение средней длины генерируемых им выходных кодов к этому значению. Наиболее близок по средней длине кодов к энтропии арифметический кодер. Стоит отметить, что в данной работе используется модификация арифметического кодера для целочисленной арифметики, называемая range-кодером.

В данной работе применяется особый способ контекстного моделирования, который может быть описан следующим образом. Для текущего значения ошибки на выходе модели анализируются значения

её трёх соседей: левого, верхнего и левого-верхнего (по диагонали). Для анализа знак ошибки отбрасывается, и выбирается максимальное значение из трёх.

Затем модуль ошибки квантуется до восьми уровней для компоненты Y и до девяти для компонент U и V следующим образом:

$$QV = \log_2(C), \quad (1)$$

где QV – квантованное значение ошибки;

C – исходное значение ошибки.

Полученное квантованное значение представляет собой контекст данной ошибки. Этот контекст в дальнейшем используется для выбора одной из восьми (девяти) статистических моделей для кодирования соответствующего значения ошибки.

Реализация метода

Практическая реализация приложения, осуществляющего требуемые алгоритмом действия и преобразования, была выполнена в среде разработки Microsoft Visual C++. Созданное в процессе работы решение (solution) составлено из двух проектов (рис. 2).

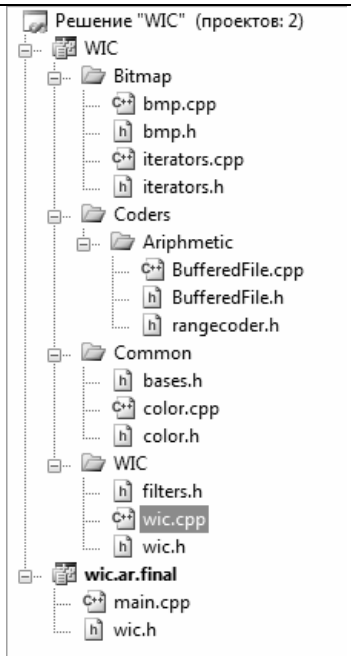

Рис. 2. Структура решения в VS 2008

Проект «WIC» представляет собой статическую библиотеку, реализующую две основные функции для работы (заголовочный файл wic.h):

```
// convert Windows BMP file to WIC file
    void compress( const char _infile[],
const char _outfile[] );

// convert WIC file to Windows BMP file
    void decompress( const char _infile[],
const char _outfile[]);
```

Остальные файлы проекта «WIC» реализуют эти функции. Результатом компиляции данного проекта является файл статической библиотеки WIC.lib.

Проект «wic.ar.final» представляет собой консольное приложение, использующее библиотеку WIC.lib.

Оба проекта написаны на языке C++ с использованием принципов ООП, и логически представляют собой иерархию классов.

Тестирование приложения

Приложение было протестировано на нескольких различных ПК под управлением ОС Windows XP и Windows 7 x86. Тестирование в 64-битном окружении не проводилось, нормальная работа приложения под такими ОС не гарантируется.

Тестирование степени сжатия производилось на наборе полноцветных фотореалистичных тестовых bitmap-изображений и нескольких дополнительных bitmap-изображений, найденных при помощи поисковой системы Google. Усовершенствованный алгоритм сравнивался с существующими популярными на данный момент форматами изображений, такими как JPEG 2000 и JPEG-LS, PNG, а также со сжатием исходных изображений при помощи универсального архиватора WinRAR. Численные результаты тестов приведены в табл. 1 и на рис. 3.

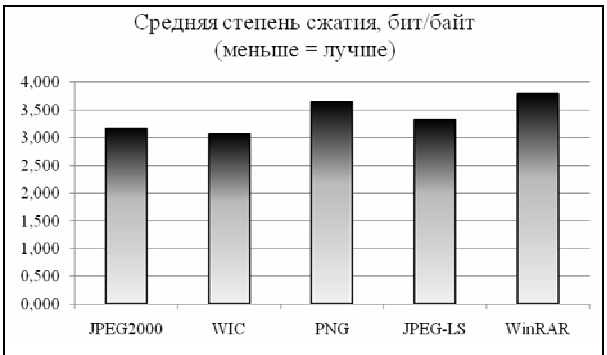

Рис. 3. Результаты тестирования степени сжатия

Как видно из результатов тестирования, модифицированный в ходе данной работы алгоритм и его реализация обеспечивают лучшую степень сжатия среди всех тестируемых продуктов.

Выводы

В рамках данного исследования был усовершенствован алгоритм сжатия изображений без потерь, который был программно реализован на языке C++. Программа имеет простой консольный интерфейс и позволяет сжимать bitmap-файлы. Тестирование приложения показало его преимущество над существующими алгоритмами сжатия изображений без потерь по степени сжатия.

Таблица 1

Результаты тестирования степени сжатия

Имя файла	JPEG2000, bpB	WIC, bpB	PNG, bpB	JPEG-LS, bpB	WinRAR, bpB
artificial.bmp	0,977	0,766	0,736	0,774	0,752
bridge.bmp	4,533	4,474	4,746	4,528	5,279
building.bmp	3,558	3,438	4,262	3,812	4,418
cathedral.bmp	3,746	3,572	4,163	3,774	3,933
church.bmp	3,125	3,145	3,967	3,969	3,904
deer.bmp	5,317	5,368	5,307	5,222	6,074
eiffel_tower.bmp	3,427	3,355	4,185	4,154	4,185
fireworks.bmp	1,779	1,721	2,126	1,520	2,179
flower.bmp	2,136	2,018	2,541	2,031	2,454
fraumunster.bmp	2,987	2,845	3,832	3,889	3,664
hdr.bmp	2,531	2,436	3,016	2,458	3,090
leaves.bmp	4,492	4,305	5,044	4,635	5,173
munich.bmp	3,171	3,124	3,868	3,274	4,333
nightshot.bmp	2,431	2,273	2,856	2,195	2,842
spider_web.bmp	1,901	1,742	2,377	1,814	2,459
tree.bmp	4,387	4,235	4,648	4,310	5,105
windmill.bmp	3,308	3,350	4,254	4,179	4,559
Суммарное	3,386	**3,296**	3,947	3,680	4,114
Среднее	3,165	**3,069**	3,643	3,326	3,788

Дальнейшее развитие проекта предполагает усовершенствование разработанного приложения в плане введения дополнительных функциональных возможностей, а также улучшения алгоритма как по степени сжатия, так и по скорости работы.

Список литературы

1. Ватолин Д. Методы сжатия данных / Д. Ватолин, А. Ратушняк, М. Смирнов, В. Юкин. – М.: ДИАЛОГ-МИФИ, 2003. – 381 с.

2. Bell T. Modeling For Text Compression / T. Bell, I. Witten, J. Clearly // ACM Computing Surveys. – Dec. 1989. – Vol.21, No.4. – P. 557-591.

3. Memon N. Recent Developments in Context-Based Predictive Techniques for Lossless Image Compression /

4. N. Memon, X. Wu // The Computer Journal. – 1997. – Vol. 40, No. 2/3. – P. 127-136.

5. Solomon D. Data Compression / D. Solomon. – NY.: Springer-Verlag New York, Inc., 2004. – 898 p.

6. Сэломон Д. Сжатие данных, изображений и звука / Д. Сэломон. – М.: Техносфера, 2004. – 368 с.

7. Тестовые изображения. [Электронный ресурс]. – Режим доступа к ресурсу: http://www.image compression.info/test_images/

Поступила в редколлегию 25.07.2011

Рецензент: д-р техн. наук О.Г. Руденко, Харьковский национальный университет радиоэлектроники, Харьков.

МОДИФІКОВАНИЙ МЕТОД ПРЕДИКАТИВНОГО КОДУВАННЯ ДЛЯ СТИСНЕННЯ ГРАФІЧНОЇ ІНФОРМАЦІЇ

С.Г. Удовенко, А.А. Шамраєв, О.О. Шамраєва, С.Д. Лук'яненко

Удосконалено алгоритм стиснення зображень без втрат та реалізовано на мові С++ для стиснення bmp-файлів. Результати тестування показали перевагу модифікованого алгоритму над існуючими аналогами за ступенем стиснення при порівнянній швидкості стиснення.

Ключові слова: *стиснення зображень, bmp, предиктивний метод кодування, міжканальна декорреляція, коректування знаку помилки.*

A MODIFIED PREDICTIVE CODING METHOD FOR COMPRESSION OF GRAPHIC INFORMATION

S.G. Udovenko, A.A. Shamraev, E.O. Shamraeva, S.D. Lukyanenko

Lossless image compression algorithm for compress bmp-files was improved and implemented in C ++. Test results showed the superiority of modified algorithm over the existing analogs by compression performance with comparable compression rate.

Keywords: *compression of images, bmp, предиктивный method of encoding, interchannel decorrelation, adjustment of sign of error.*