УДК 004.932.72

СЖАТИЕ ПОЛУТОНОВЫХ ИЗОБРАЖЕНИЙ НА ОСНОВЕ АДАПТИВНОГО RLE-КОДИРОВАНИЯ С ПЕРЕМЕННОЙ РАЗРЯДНОСТЬЮ СЧЕТЧИКА СЕРИИ

А.Г. ЛОПАТО, В.Ю. ЦВЕТКОВ

Белорусский государственный университет информатики и радиоэлектроники, Реуспблика Беларусь

Поступила в редакцию 16 ноября 2018

Аннотация. Предложена модификация алгоритма кодирования длин серий для сжатия полутоновых изображений без потерь, отличающийся от базового алгоритма изменением разрядности счетчика серий в зависимости от вероятности длины серии.

Ключевые слова: сжатие изображений, кодирование длин серий.

Введение

В настоящее время для сжатия изображений без потерь широко используются алгоритмы JPEG [1] и JPEG 2000 [2]. Они основаны на энтропийном кодировании коэффициентов дискретно-косинусного и вейвлетного преобразований. Для сжатия изображений без потерь часто используются также универсальные алгоритмы архивации данных: RAR и входящие в состав архиватора ZIP алгоритмы Deflate и LZMA, кодирующие значения пикселей [3]. Данные алгоритмы позволяют сжимать изображения без потерь примерно в 2 раза, однако их использование требует значительных вычислительных ресурсов. В случае когда временные и вычислительные ресурсы ограничены, для сжатия изображений необходимо использовать более простые алгоритмы эффективного кодирования, например, алгоритм кодирования длин серий RLE (Run-Length Encoding) [4], основанный на учете повторов символов. В RLE последовательности одинаковых символов заменяются двумя символами – значением серии и счетчиком серии. Один из недостатков данного алгоритма заключается в отсутствии адаптации разрядности счетчика серии к вероятности длины серии, что приводит к сравнительно низким коэффициентам сжатия.

Целью работы является разработка модификации алгоритма кодирования длин серий, основанного на изменении разрядности счетчика серии в зависимости от вероятности длины серии.

Адаптивный алгоритм кодирования длин серий

Предлагается модификация алгоритма кодирования длин серий RLE для сжатия полутоновых изображений, основанная на изменении разрядности счетчика серии в зависимости от вероятности длины серии. Сущность алгоритма состоит в уменьшении длины счетчика серии на один разряд при многократном повторении серий, кодирование длин которых не задействует старший разряд счетчика серии, и в увеличении длины счетчика серии на один разряд при кодировании длины серии, для учета которой разрядности счетчика серии недостаточно.

При достижении определенной битовой плоскости кодирование длин серий перестает обеспечивать коэффициент сжатия превышающий единицу. Когда длина поля адаптивно достигает двух бит, алгоритм переходит на реализацию встраивания в код несжатой последовательности бит. При этом для каждых 30 бит определяется количество изменений символа, т. е. количество последовательностей. Тем самым выявляется необходимость возврата к кодированию длин последовательностей.

Экспериментально установлено, что прирост коэффициента сжатия в данном алгоритме достигается за счет введения раздельных переменных значений разрядности счетчика серии для единиц и нулей, поскольку старшие битовые плоскости являются гораздо более разреженными, чем младшие. Переход к встраиванию несжатой последовательности при этом осуществляется при достижении длин нулевых и единичных серий наименьшего значения.

Для повышения эффективности алгоритма сжатия исходного изображения целесообразно выполнить его обратимое преобразование, в результате которого уменьшаются значения пикселей.

Исследованы дифференциальное кодирование и вейвлет-преобразование (последнее обеспечило наибольший коэффициент сжатия). Преобразования приводят к появлению отрицательных значений (единиц в старших битовых плоскостях для отрицательных коэффициентов, в том числе и близких к нулю), что обуславливает увеличение количества последовательностей. Это негативно сказывается на коэффициенте сжатия. Эффективным решением данной проблемы является переход от дополнительного кода к структуре, хранящей знаковый бит в младшем разряде и модуль значения пикселя в старших разрядах.

Алгоритм имеет несколько параметров, изменение которых сказывается на коэффициенте сжатия:

– изначальные значения длин последовательностей (для единиц выбирается намного меньшее значение, чем для нулей);

– длина серии последовательностей, не использующих старший бит поля, после достижения которой уменьшается его размер;

– количество переходов к противоположному символу при встраивании несжатой последовательности за последние 30 бит.

Пошаговое описание алгоритма

Блок-схема предлагаемого алгоритма представлена на рис. 1. Для его выполнения необходимо предварительно выполнить двумерное вейвлетное преобразование, квантование с определенными коэффициентами (опционально) и переход от дополнительного кода к прямому для каждого пикселя.

Поскольку алгоритм работает с отдельными битами, необходимо реализовать развертку изображения. Для ускорения аппаратного выполнения алгоритма (распараллеливания, конвееризации) его необходимо разрабатывать таким образом, чтобы отдельные участки изображения кодировались независимо друг от друга. Эксперименты показали, что увеличение размеров блока приводит к увеличению коэффициента сжатия. Однако при аппаратной реализации это ведет также к увеличению необходимого размера буфера. Удовлетворительным по коэффициенту сжатия является размер блока 4096 пикселей.

Апробированы два варианта развертки – по блокам (64×64) и по вертикальным линиям (512×8, 1024×4, 2048×2, 768×5,33). При сравнении этих вариантов установлено, что коэффициент сжатия при линейной развертке лишь немного уступает блочной, однако проводить линейную развертку аппаратно гораздо менее ресурсозатратно, т. к. исключается необходимость дополнительной буферизации данных во внутренней памяти ПЛИС. Такой вариант является предпочтительным. Если размер изображения не кратен 4096 пикселям, то при линейной развертке можно добавить недостающее число нулевых пикселей.

Следующим этапом является определение номера старшей битовой плоскости, в которой есть единичные биты, и запись его в результирующую последовательность. С этой плоскости начинается кодирование. При аппаратной реализации это можно сделать при последовательном занесении каждого пикселя в буфер кодера.

Далее выполняется адаптивное кодирование длин серий. Первоначально проводится инициализация переменных значений алгоритма (длин полей нулей и единиц, флагов состояния кодера: флага включения несжатых бит в результирующую последовательность и флага первого бита в последовательности).

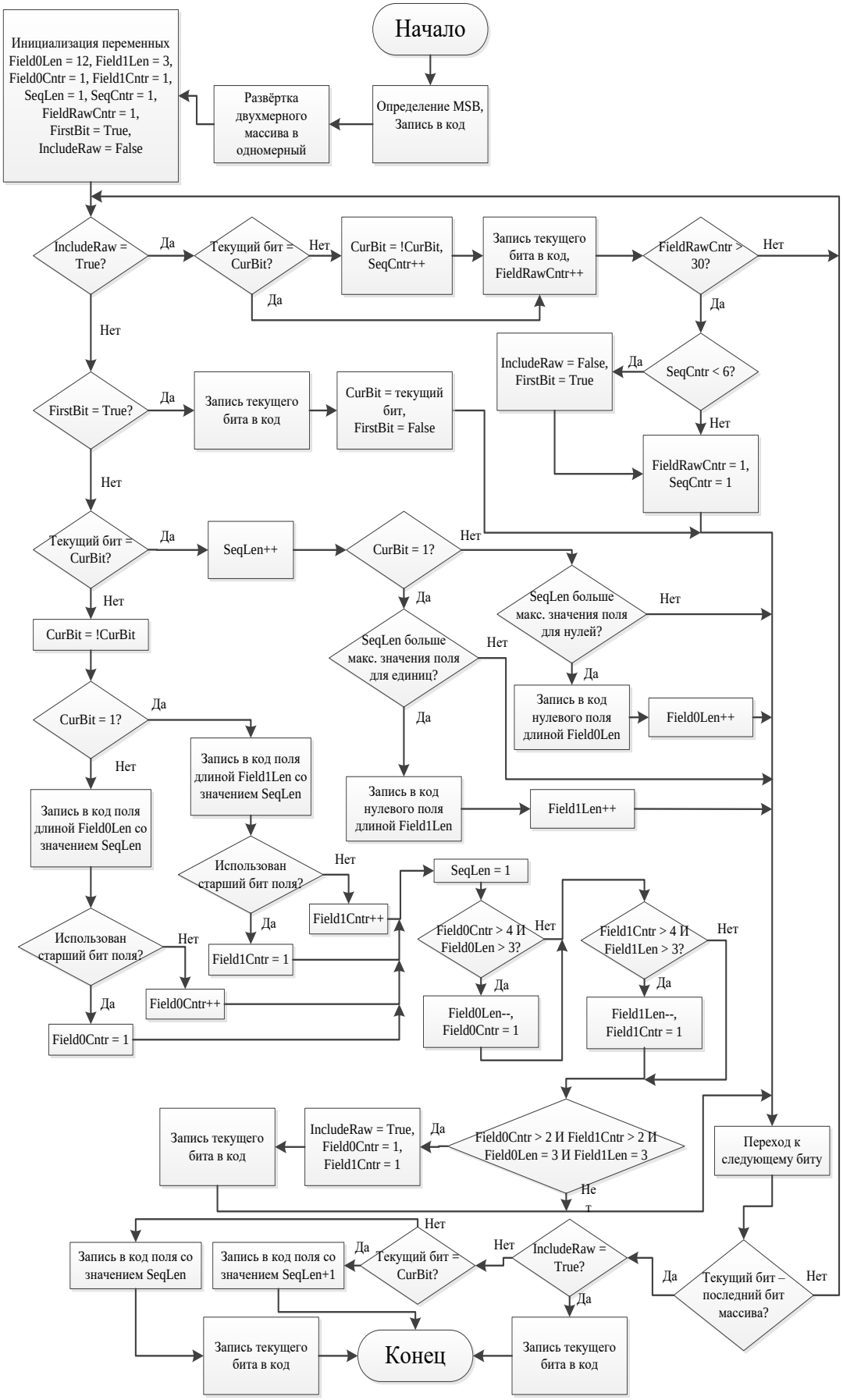

Рис. 1. Алгоритм адаптивного кодирования длин серий

В процессе выполнения алгоритма осуществляется проход по каждому писклею каждой битовой плоскости. При этом формируется сжатый поток данных, состоящий из блоков переменной длины, что при программной реализации проще выполнить в виде последовательного двоичного массива. При аппаратной реализации необходимо непосредственно на этапе кодирования формировать выходной поток с определенной длиной слова, равной разрядности шины данных (16 бит в данном случае). Для этих целей выделяются регистр, разрядность которого равна двукратной разрядности шины, и указатель на его биты. При кодировании очередного некратного блока данных он заносится в регистр по адресу указателя, а значение указателя увеличивается на размер этого блока. Если указатель переходит на старшую половину регистра, то его младшая половина выгружается на шину, старшая сдвигается в младшую половину и значение указателя уменьшается на значение разрядности шины.

При кодировании пикселя изначально проверяется флаг включения несжатых данных (IncludeRaw). Если он равен единице, бит заносится в результирующую последовательность. Так же при этом проводится оценка частоты смены бит. Если за 30 бит ноль сменяется единицей (и наоборот) менее шести раз, что говорит о достаточном для сжатия количестве последовательностей в коде, то флаг IncludeRaw сбрасывается в ноль и далее снова начинает выполняться кодирование длин серий. Если флаг IncludeRaw сброшен в ноль, то для текущего бита проверяется флаг первого бита в последовательности (FirstBit). Если он установлен в единицу, то бит заносится в последовательность, чтобы в последствии при декодировании его можно было восстановить. При нулевых значениях обоих флагов проверяется значение текущего бита. Если оно совпадает с предыдущим, то соответствующее значение длины последовательности увеличивается на единицу. При этом значение может превысить максимально возможное для текущей длины поля. Если это происходит, длина поля увеличивается на единицу, а для информирования об этом декодера в результирующую последовательность заносится нулевое поле текущей длины.

При несовпадении текущего бита с предыдущим в результирующую последовательность вносится поле со значением длины последовательности бит. Если при этом не задействуется старший бит поля, то инкрементируется значение соответствующего счетчика полей избыточной длины (Field0Cntr или Field1Cntr). Если значение этого счетчика превышает четыре, то размер поля уменьшается на единицу. Поскольку данный процесс происходит и в кодере, и в декодере, то нет необходимости включать в код дополнительную информацию об этом.

Если значения обоих переменных Field0Cntr и Field1Cntr достигли наименьшего значения, то путем установки флага IncludeRaw выполняется переход к вставке в код несжатых данных.

Оценка эффективности использования алгоритмов кодирования длин серий для сжатия полутоновых изображений

Для тестовых изображений, представленных на рис. 2, в таблице приведены коэффициенты сжатия, полученные для предложенного адаптивного алгоритма RLE, а также алгоритмов SPECK и JPEG2000.

Из таблицы следует, что разработанный алгоритм находится на одном уровне по коэффициенту сжатия с алгоритмом SPECK, но уступает алгоритму JPEG2000. Однако кодирование изображений с помощью данного алгоритма в среде MATLAB выполняется в 2–2,5 раза быстрее, чем с помощью JPEG2000.

В случае аппаратной реализации адаптивного кодирования длин серий на обработку одного бита изображения затрачивается приблизительно один такт работы ПЛИС, что многократно превосходит JPEG2000, в котором необходимо выполнить несколько проходов по одним и тем же битам. При этом адаптивный кодер RLE уступает кодеру SPECK, реализация которого позволяет за один такт обрабатывать четыре и более бит. Однако отдельное ядро сжатия RLE в кристалле ПЛИС фирмы Xilinx серии Kintex-7 занимает всего порядка 600 таблиц истинности, что за счет распараллеливания позволяет увеличить эффективность использования ресурсов и подобрать необходимую пропускную способность для конкретной задачи путем вариации числа ядер.

а *б* *в*

Рис. 2. Тестовые изображения: *а* – М1 (128×128 пикселей); *б* – М2 (256×256 пикселей); *в* – М3 (512×12 пикселей)

Размеры кода при сжатии тестовых изображений без потерь

Алгоритм	Коэффициент сжатия		
	Airplane_1024	Airport_1024	Island_1024
RLE	1,79	1,54	1,73
SPECK	1,77	1,57	1,80
JPEG2000	1,91	1,71	1,97

Заключение

Предложена модификация алгоритма кодирования длин серий для сжатия без потерь полутоновых изображений, отличающиеся от базового алгоритма RLE изменением разрядности счетчика серии в зависимости от вероятности длины серии. Установлено, что предложенный алгоритм обеспечивает коэффициент сжатия изображений без потерь в области дискретного вейвлет-преобразования, близкий к коэффициенту сжатия алгоритма SPECK (1,5–1,7 раза), и скорость кодирования, в 2 раза превышающую скорость кодирования алгоритма JPEG2000.

COMPRESSION OF HALF-TONE IMAGES BASED ON ADAPTIVE RLE-CODING WITH VARIABLE DISTRIBUTION OF A SERIES COUNTER

A.G. LOPATO, V.Yu. TSVIATKOU

Abstract. A modified run-length coding algorithm for lossless compressing grayscale images is proposed. It differs from the baseline algorithm by adaptively changing run-lengths.

Keywords: images compression, run-length encoding.

Список литературы

1. Pennebaker W.B., Mitchell J.L. JPEG Still Image Compression Standard. New York, 1993.
2. Ebrahimi T. // Proc. of the SPIE. San Diego, July–August 2000. Vol. 4115. P. 446–454.
3. Ватолин Д. Методы сжатия данных. Устройство архиваторов, сжатие изображений и видео. М., 2003.
4. Golomb S.W. // IEEE Transactions on Information Theory. 1966. July. P. 399–401.