The Results of Testing of Lossless Compression Algorithm Based on Cascade Fragmentation Method and Ordering Pixels Sequence

V. Smirnov [1], A. Korobeynikov [2]

[1]Dept. Software, Kalashnikov Izhevsk State Technical University, Izhevsk, Russia
E-mail: drolbs@gmail.com
[2] Dept. Software, Kalashnikov Izhevsk State Technical University, Izhevsk, Russia
E-mail: kav33@inbox.ru

Received: 25.11.2016

Abstract. The present paper describes testing the algorithm based on methods of cascade fragmentation and ordering pixels sequence. The test images were borrowed from the sample images database available on University of Southern California site. All images were divided into 4 groups. Each group featured images of certain class and size. Testing was done at the following algorithm parameters: with variable sizes of image fragments (5×5, 6×6, 16×16); with variable number of ordering methods (bypass methods) applied to fragment's pixels (16, 32, 64, 128, 256). Test results were represented as average, minimum, maximum and variance compression ratio. It was shown that average compression ratio can be increased by 4.7 % for some groups of images and compression algorithm parameters. Moreover, cascade fragmentation was found to have other advantages over "snake" image bypass.

Keywords: lossless image compression, compression algorithms testing, cascade fragmentation, ordering pixels sequence, optimal bypass, code book

INTRODUCTION

All of image compression algorithms can be classified into two groups: 1) the bypass of image pixels matrix and obtaining the ordered pixels sequence (array); 2) the transformation of image matrix; transformation results are bypassed subsequently.

The methods of image bypass [1–4] generally come down to simple bypass methods; in particular, "snake" bypass is applied to JPEG images. At this bypass, the value of each subsequent pixel is predicted from previous values. Regardless of complex predictors, a simple method of pixels matrix bypass is used. In most cases delta-coding (the first finite difference) is applied as predictor. For transformation purposes, Fourier discrete transform [5] as well as wavelet transform is commonly employed [6].

In the previous paper [7] the authors proposed cascade splitting of an image into multiple fragments of fixed size. Then, finding optimal bypass trajectory was considered which was thought to result in greater compression ratio compared to traditional "snake" bypass. A more detailed theoretical description of the tested algorithm can be found in papers [7–9].

ORDER AND RESULTS OF TESTING

The present research was aimed at developing a beta software version for image compression and comparing the results of compression with and without the use of cascade fragmentation and bypass optimization. The following parameters are assumed variable at testing: image fragment size; the number of ordering methods (bypasses) applied to fragment's pixels. Delta-coding is used as predictor. To compress the prediction error (modulo) Huffman's method is applied [1].

The developed method was tested on sample test images [10] taken from the site of *University of Southern California* on their *Signal and Image Processing Institute* page. The following images were tested: 2.1.01...2.1.12, 2.2.01...2.2.24, 4.1.01...4.1.08, 4.2.01...4.2.07.

In order to understand the efficiency of this or that compression mode, the tested images were divided into 4 groups based on the following criteria: size and class of an image. Thus, the first «2.1.*.*tiff*» and the second «2.2.*.*tiff*» groups represent aerial photographs of 512×512 and 1024×1024 pixels size respectively. The third «4.1.*.*tiff*» and the fourth «4.2.*.*tiff*» groups are photographs of 256×256 and 512×512 size respectively. Additionally, the general result for all images was provided. Each group was tested at the following bypass parameters: without cascade fragmentation and optimization ("snake" bypass) and with cascade fragmentation and bypass optimization [7]. The examples of free parameters of compression algorithm included fragment sizes at *nxn* splitting and the number of fragment bypasses *{k}*: 5×5 {824}, 6×6 {22144}, 16×16 {16}, 16×16 {32}, 16×16 {64}, 16×16 {128}, 16×16 {256}. The 16×16 fragment was limited reagrding the number of its bypasses with the view of saving on compression time and the expenses of coding the bypass number.

Test results are shown in tables 1…4 with average, minimum, maximum, and variance compression ratio represented by groups of images and compression types respectively. Similarly, the test results by compression time are given in tables 5…8.

Table 1. Average compression ratio of images

Groups of images	Average compression ratio							
	"Snake" bypass	5×5 {824}	6×6 {22144}	16×16 {16}	16×16 {32}	16×16 {64}	16×16 {128}	16×16 {256}
2.1.*.*tiff* 512×512	1.424	1.414	1.424	1.426	1.426	1.426	1.426	1.426
2.2.*.*tiff* 1024×1024	1.547	1.536	1.546	1.548	1.548	1.548	1.548	1.547
4.1.*.*tiff* 256×256	1.618	1.627	1.644	1.622	1.622	1.622	1.622	1.622
4.2.*.*tiff* 512×512	1.549	1.556	1.569	1.553	1.553	1.553	1.552	1.552
All images	1.539	1.531	1.542	1.540	1.540	1.540	1.540	1.540

Table 2. Minimum compression ratio

Groups of images	Minimum compression ratio							
	"Snake" bypass	5×5 {824}	6×6 {22144}	16×16 {16}	16×16 {32}	16×16 {64}	16×16 {128}	16×16 {256}
2.1.*.*tiff* 512×512	1.219	1.212	1.220	1.219	1.219	1.219	1.219	1.219
2.2.*.*tiff* 1024×1024	1.284	1.271	1.275	1.283	1.283	1.283	1.283	1.283
4.1.*.*tiff* 256×256	1.222	1.197	1.203	1.218	1.218	1.218	1.218	1.217
4.2.*. *tiff* 512×512	1.222	1.197	1.203	1.218	1.218	1.218	1.218	1.217
All images	1.219	1.197	1.203	1.218	1.218	1.218	1.218	1.217

Table 3. Maximum compression ratio

Groups of images	**Maximum compression ratio**							
	“Snake” bypass	**5×5 {824}**	**6×6 {22144}**	**16×16 {16}**	**16×16 {32}**	**16×16 {64}**	**16×16 {128}**	**16×16 {256}**
2.1.*.*tiff* 512×512	1.618	1.613	1.624	1.621	1.621	1.621	1.621	1.621
2.2.*.*tiff* 1024×1024	2.054	1.980	2.017	2.042	2.042	2.042	2.043	2.042
4.1.*.*tiff* 256×256	2.108	2.120	2.167	2.107	2.107	2.107	2.106	2.107
4.2.*.*tiff* 512×512	1.883	1.915	1.938	1.887	1.887	1.887	1.887	1.886
All images	2.409	2.463	2.510	2.423	2.422	2.422	2.422	2.421

Table 4. Variance image compression ratio

Groups of images	**Variance compression ratio**							
	“Snake” bypass	**5×5 {824}**	**6×6 {22144}**	**16×16 {16}**	**16×16 {32}**	**16×16 {64}**	**16×16 {128}**	**16×16 {256}**
2.1.*.*tiff* 512×512	0.016	0.016	0.017	0.017	0.017	0.017	0.017	0.017
2.2.*.*tiff* 1024×1024	0.027	0.026	0.029	0.026	0.026	0.026	0.026	0.026
4.1.*.*tiff* 256×256	0.081	0.089	0.097	0.081	0.081	0.081	0.081	0.081
4.2.*.*tiff* 512×512	0.049	0.058	0.061	0.050	0.050	0.050	0.050	0.050
All images	0.049	0.053	0.057	0.050	0.050	0.050	0.050	0.050

Table 5. Average image compression time

Groups of images	**Average compression time**							
	“Snake” bypass	**5×5 {824}**	**6×6 {22144}**	**16×16 {16}**	**16×16 {32}**	**16×16 {64}**	**16×16 {128}**	**16×16 {256}**
2.1.*.*tiff* 512×512	0:01:36	0:02:30	0:18:41	0:01:30	0:01:30	0:01:31	0:01:36	0:01:41
2.2.*.*tiff* 1024×1024	0:05:56	0:09:31	1:20:36	0:05:59	0:05:55	0:06:04	0:06:15	0:06:44
4.1.*.*tiff* 256×256	0:00:23	0:00:35	0:04:33	0:00:23	0:00:22	0:00:23	0:00:23	0:00:24
4.2.*.*tiff* 512×512	0:02:14	0:02:51	0:19:55	0:02:07	0:02:06	0:02:10	0:02:07	0:02:21

Table 6. Minimum compression time

Groups of images	**Minimum compression time**							
	“Snake” bypass	**5×5 {824}**	**6×6 {22144}**	**16×16 {16}**	**16×16 {32}**	**16×16 {64}**	**16×16 {128}**	**16×16 {256}**
2.1.*.*tiff* 512×512	0:00:56	0:01:56	0:18:11	0:00:52	0:00:52	0:00:53	0:00:59	0:01:02
2.2.*.*tiff* 1024×1024	0:03:46	0:07:52	1:17:03	0:04:13	0:04:10	0:04:15	0:04:26	0:04:42
4.1.*.*tiff* 256×256	0:00:18	0:00:31	0:04:29	0:00:20	0:00:19	0:00:20	0:00:20	0:00:22
4.2.*. *tiff* 512×512	0:01:36	0:02:35	0:19:32	0:01:47	0:01:44	0:01:47	0:01:46	0:01:58

Table 7. Maximum compression time

Groups of images	**Maximum compression time**							
	“Snake” bypass	**5×5 {824}**	**6×6 {22144}**	**16×16 {16}**	**16×16 {32}**	**16×16 {64}**	**16×16 {128}**	**16×16 {256}**
2.1.*.*tiff* 512×512	0:02:33	0:03:04	0:19:07	0:02:19	0:02:17	0:02:17	0:02:26	0:02:28
2.2.*.*tiff* 1024×1024	0:08:19	0:12:33	1:23:32	0:08:21	0:08:14	0:08:26	0:08:48	0:09:07
4.1.*.*tiff* 256×256	0:00:30	0:00:42	0:04:43	0:00:29	0:00:29	0:00:30	0:00:29	0:00:31
4.2.*.*tiff* 512×512	0:02:49	0:03:02	0:20:14	0:02:25	0:02:24	0:02:26	0:02:24	0:02:37

Table 8. Variance compression time

Groups of images	Variance compression time							
	“Snake” bypass	5×5 {824}	6×6 {22144}	16×16 {16}	16×16 {32}	16×16 {64}	16×16 {128}	16×16 {256}
2.1.*.*tiff* 512×512	1.025E-07	6.9E-08	4E-08	8.57E-08	8.34E-08	8.39E-08	8.87E-08	8.89E-08
2.2.*.*tiff* 1024×1024	7.286E-07	7E-07	2E-06	5.46E-07	5.21E-07	5.36E-07	6.01E-07	5.93E-07
4.1.*.*tiff* 256×256	2.134E-09	1.5E-09	3E-09	1.01E-09	1.22E-09	1.24E-09	1.06E-09	1.26E-09
4.2.*.*tiff* 512×512	1.009E-07	1.3E-08	4E-08	2.45E-08	2.53E-08	2.51E-08	2.29E-08	2.55E-08

Acknowledgment

The work was performed as part of the grant GZ/TVG-14(01.10)

Conclusions

1. Let us consider test results by groups of images. According to average compression ratio the fragmentation and optimization method proves more efficient compared to the “snake” method: 1) in group «2.1.*.*tiff* 512x512» by 0.2%; 2) in group «2.2.*.*tiff* 1024x1024» the results do not differ considerably; 3) in group «4.1.*.*tiff* 256x256» by 4.7%; 4) in group «4.2.*.*tiff* 512x512» by 2%; 5) generally, throughout all groups by 0.3%.
2. If the difference in compression ratio at variable fragment sizes is examined, the larger the fragment, the higher the compression ratio. However, compression time along with expenses of storing the bypass number in the code book increase parallel to fragment size.
3. Provided the number of chosen bypasses is reduced and the size of fragments is big enough, it is possible to obtain higher compression ratio at compression time matching that of “snake” bypass. Hence, the proposed algorithm enables adjusting compression efficiency.
4. Besides higher compression ratio the cascade fragmentation has the following advantages: it enables displaying/downloading posterized images; displaying/downloading particular fragments with improved quality as required; editing pixels of particular fragments without re-coding the whole image.

References

1. Vatolin, D., Ratushnyak, A., Smirnov, M., & Yukin, V. (2003). *Metody szhatiya dannyh. Ustroystvo arhivatorov, szhatie izobrazheniy i video [Data compression methods. The archivers internal structure, images and videos compression].* Moscow, Russia: DIALOG-MIFI, 2003.
2. Sayood, K. (2012). Lossless image compression. In *Introduction to data compression* (4th ed.) (pp. 183–215). University of Nebraska. doi:10.1016/B978-0-12-415796-5.00007-7
3. Karam L. J. (2009). Lossless image compression. In A. Bovik (Ed.) *The essential guide to image processing* (2nd Ed.) (pp. 385–419). Academic Press, Elsevier Inc. doi:10.1016/B978-0-12-374457-9.00016-0
4. Li, Z.-N., Drew, Mark, S., & Liu, J. Lossless compression algorithms (2014). In *Fundamentals of multimedia* (pp. 185–224). Springer, Cham. doi:10.1007/978-3-319-05290-8_7
5. Pennebaker, W. B., & Mitchell, J. L. (1992). *JPEG: Still Image Data Compression Standard.* Springer Science & Business Media.
6. Taubman, D., & Marcellin, M. (2001). *JPEG2000 Image Compression Fundamentals, Standards and Practice.* Springer Science & Business Media.

7. Smirnov, V., & Korobeynikov, A. (2016). Ordering the numeric sequence of image pixels at lossless compression. In *Instrumentation Engineering, Electronics and Telecommunications – 2015: Paper book of the I International Forum IEET-2015 held within the framework of the XI International Scientific-Technical Conference "Instrumentation Engineering in the XXI Century Integration of Science, Education and Production"*. Izhevsk, Russia: Publishing House of Kalashnikov ISTU. 175–180. Retrieved from http://pribor21.istu.ru/proceedings/IEET-2015/IEET-2015_Smirnov.pdf
8. Smirnov V. S., & Korobeynikov A. V. (2012). Opredelenie optimal'nogo obhoda s ispol'zovaniem kodovoy knigi pri szhatii izobrazheniy bez poter' [Optimal Bypass Definition with Code Book Application at Images Lossless Compression]. *ISTU bulletin, 2012*(2). 143–144.
9. Korobeynikov A. V., & Smirnov, V. S. (2012). Kaskadnoe razbienie izobrazheniya na fragmenty pri szhatii bez poter' na osnove optimizacii obhoda [Cascade Image Splitting into Fragments at Lossless Compression on Basis of Image Bypass Optimization]. *ISTU bulletin, 2012*(3). 114–115.
10. The USC-SIPI Image Database. (n.a.). Los Angeles, USA: University of Southern California. Retrieved from http://sipi.usc.edu/database/database.php?volume=misc