An Application for the Comparison of Lossless Still Image Compression Algorithms

Tomáš Vogeltanz, Pavel Pokorný

***Abstract*—**This paper describes the comparison of results of lossless compression algorithms used for still image processing. In order to create this comparison, an application was developed which allows one to compare the effectiveness and quality of modern lossless image compression algorithms. The first section of this paper describes the evaluation criteria of the compression algorithm effectiveness. The second section briefly summarizes the types of compression algorithms and the graphic file formats that the application supports and tests. The following section describes the architecture of the application and its user interface. The final section contains the comparison of results for JPEG photos, RAW photos, grayscale photos, High-Dynamic-Range (HDR) color photos, High-Dynamic-Range (HDR) grayscale photos, 24-bit images, 8-bit images, 4-bit images, 1-bit images, grayscale images, 1-bit text images, and grayscale text images. The results are described and evaluated.

***Keywords*—**Application, Comparison, Compression Ratio, Compression Time, Decompression Time, Lossless Compression, Lossless Image Compression

I. Introduction

Compression is a process that is used to reduce the physical size of an information block. In other words, the files are coded in such a form that their size is smaller than the original size before compression. Coding means a way of representing data when they are stored in a file, memory, etc.

Each compression algorithm is designed so that it searches for and uses data compression for a given order in the stored data. This procedure can include the repeated character sequence, the frequency of occurrence of individual characters, the identification of large blocks of the same data and more. [2]

Various data characters require a different approach to their compression. It is clear that the character of graphic data will be different to other files (texts, executable files), and that the variance of the data causes the different possibilities of some compression algorithms applicability. [1]

The basic division of compression algorithms is lossless and lossy. The lossless compression is a compression method, in which no information is lost - after decompressing we get the same data as before compression. Sometimes, instead of the term "lossless compression" the term "accurate compression" or "reversible compression" is also used. In general, these methods do not achieve as good compression ratios such as loss methods, but they are also used as auxiliary algorithms for encoding video and audio. The lossy compression is a compression method in which the original data blocks are altered or some less significant values are neglected in order to achieve higher compression ratios. In this case, the decompression algorithm generates different values than before compression. Lossy compression methods are mainly used in the compression of video and audio. [1]

Symmetric and asymmetric is another way of division compression algorithms. Symmetric Compression occurs in the case where the time (and also usually the number and type of operations) required to compress and to decompress data is approximately the same. Asymmetric compression has a different required time to compress and decompress. In data compression, most compression algorithms require one to make more operations. Asymmetric algorithms, which require a longer decompression time, are not so widespread. [3]

The main parameters to compare the performance of compression algorithms are the compression ratio and the compression or decompression time. The compression ratio is usually expressed as the ratio between the size of the compressed and uncompressed data. Under this scheme, if the compression program compresses the file of an original length of 200kB to 50 kB, the compression ratio is 1:4, respectively 25%. The smaller the number of the compression ratio gives the better compression result. The compression time is the time necessary to transform the original information in to the compressed form. The decompression time indicates the reverse process – it is the time needed to extract the compressed file to its original form. [2]

The compression algorithm speeds are also affected by the hardware of the computer and by the software which is running on the computer. The main parameters of the computer that affect its performance are the processor (its clock frequency, number of cores, etc.), the size and access speed of RAM, the access time and the read/write speed of the hard drive, the hard drive status occupation and fragmentation, enabling or disabling the cache memory and the type of the installed operating system. [2]

II. Lossless Compression Algorithms

The theoretical background of the data compression techniques is strong and well established. It dates back to the seminal work of Shannon who, more than half a century ago, gave precise limits on the performance of any lossless compression algorithm; this limit is the entropy of the source we want to compress and it is defined as (1), where p_j is the

probability of symbol j. The entropy of the source (or 0-th order entropy) defines the average amount of information obtained by observing a single source output. [4] [5]

$$H(A) = -\sum_{j=1}^{q} p_j \log_b p_j \quad (1)$$

The source entropy measure gives us the average amount of information per symbol. The information from a source can be represented in fewer bits per symbol; in that case the maximum number of bits/symbol cannot be less than H. If that is true, it is said that the source contains statistical redundancy. [4]

Raster images are characterized by high memory consumption that grows quadratically with their resolution. In contrast to common file compression, raster images have benefits in their own characteristics which usually allow us to get a much smaller size after the compression. In these cases, we get a smaller value of the compression ratio [2].

There are many compression algorithms and even more their modifications. We concentrated our attention on the most common lossless compression algorithms which are also supported by our application. They are the following algorithms (or modifications of these algorithms): RLE, LZ77, LZW, Huffman Coding, JBIG, Integer Wavelet Transformation, and Predictive Coding. [1] [2]

Run-length Encoding (RLE) is a very simple compression algorithm. It is based on the principle where the same data value in the sequences are stored as a single data value and count, rather than as the original run. Run-length encoding and its modifications are well suited to palette-based iconic images. Common graphic formats for run-length encoded data include BMP, PCX, Truevision TGA, and TIFF (the PackBits modification). [1]

Static dictionaries can be used when the behavior of the input source is well known in advance; otherwise a constantly changing, adaptive dictionary is used to give a good compromise between compression efficiency and computational complexity. These algorithms are often called dictionary based methods, or dictionary methods, or Lempel-Ziv methods after the seminal work of Lempel and Ziv. [6] Mainstream methods that are based on dictionaries are almost always adaptive dictionary methods; this adaptive technique is flexible and can be easily used in practical situations. [5]

In practice the textual substitution compression methods are all inspired by one of the two compression approaches presented by Lempel and Ziv. These methods are often called LZ77 and LZ78 or LZ1 and LZ2 respectively in the order in which they have been published. There are many possible variants of LZ1 and LZ2 and they generally differ in the way the pointers in the dictionary are represented and in the limitations on the use of these pointers. [6]

The LZ77 algorithm achieves compression by replacing repeated occurrences of data with references to a single copy of that data existing earlier in the uncompressed input data sequence. A match is encoded by a pair of numbers, which is called a length-distance pair. The modification of this compression algorithm is used by the PNG (the Deflate modification) and WEBP (the VP8L modification). [2] [7]

Lempel–Ziv–Welch (LZW) is a universal data compression algorithm, which improves implementation of the LZ78 algorithm. It encodes sequences of 8-bit data as fixed-length 12-bit codes. At each stage in compression, input bytes are gathered into a sequence until the next character would make a sequence for which there is no code yet in the dictionary. Common formats for LZW encoded data include the GIF, TIFF, and EXR (ZIP modification) graphic formats. [1] [2]

Huffman coding refers to the use of a variable-length code table for encoding a source symbol (such as a character in a file). There, the code table is derived in a particular way based on the estimated probability of occurrence for each possible value of the source symbol. With the help of this table, each value with a more frequent occurrence has a shorter code assigned at the output. The modification of this compression algorithm is used by the TIFF (the CCIT modifications), JPEG-LS (the LOCO-I modification), PNG (the Deflate modification), and WEBP (the VP8L modification) graphic formats. [2] [3] [7]

The JBIG image compression standard is widely implemented in fax machines. Now that the newer bi-level image compression standard JBIG2 has been released, JBIG is also known as JBIG1. JBIG is based on the probabilities of each bit on the previous bits and the previous lines of the picture. It does not reference future bits in order to allow compress and decompress of images in scanning order. This compression algorithm is used in the JBIG and JBG file formats. [2]

Wavelet transformation transforms signals in the time-frequency field. It can be either lossless or lossy. The lossless variant (i.e. integer wavelet transformation) produces the integer coefficients, from which the image can be fully reconstructed. This compression is used in the JPEG 2000, HDP, and EXR (the PIZ modification) graphic formats. [2]

The oldest IWT is an integer version of the Haar Transform, which is called the S transform. Said and Pearlman came out with an improved method that performs the S transform plus a Prediction step in order to generate a new set of high-pass coefficients. This transform is called the S+P transform. [8]

The Lifting Scheme (LS) is a method to construct biorthogonal wavelets, which are required those latter to obtain a perfect reconstruction of the image. First the input data are split into even and odd samples. The even signal is convolved with a low pass filter, and is added to the odd signal. The roles of the signals are inverted and then we can apply M lifting steps. In the output of the process we will have one high pass signal and a low pass signal. We call the lifting steps that use the low pass signal “prediction steps”, and those, which use the high pass signal, the “update steps”. To reconstruct the original signal, we simply use the same structure but just in the reverse order. [8]

To apply the integer wavelet transform on images, we have

to compute two steps. First a one-level forward IWT is applied on the horizontal dimension. The result of this will be two integer output sub-images. We then use another one-level forward IWT in the vertical dimension. The first operation in the horizontal direction produces one low frequency image L and a high frequency image H. The second operation produces four sub-images; one low-frequency sub-image LL and three high-frequency sub-images LH, HL and HH. The next level of the IWT will process the LL sub-image using the same procedure. [8]

Predictive coding encodes the difference between the current data estimation derived from past data and actual current data to attain more efficient compression. The degree of efficiency depends very much on the accuracy of the estimation as the difference becomes smaller, the information to be encoded becomes smaller as well. One of problems of predictive coding is that it cannot estimate pixel color values very well near edges, boundaries, or when there are sharp transitions of colors. This is because predictive coding relies on the similarities of neighboring pixels for the estimation and, therefore, the dissimilarities of pixel colors near edges or boundaries can adversely affect the accuracy of the estimation of the pixel colors. The JPEG-LS (LOCO-I) is an example of the predictive coding approach which works well on continuous-tone images. The JPEG-LS is not complex and works well with grayscale images. However, its performance is not as impressive when it is applied to indexed color or color-map images. [9]

III. Compression and Graphic Formats

There are a large number of raster graphic formats. The causes of such quantities are historical (they correspond to technical development in computer technology), binding to a specific software (sometimes they contain additional information that cannot be stored in other formats – e.g. PCX, BMP, ICO), some graphic formats have been designed for transferal over the network (PNM , MIFF) or for universal use (GIF, TIFF, PNG, JPEG). Also, many compression algorithms are associated with a specific graphic format. [1]

2D raster files vary greatly in their detail, but they all share the same general structure. In each file we can find the block of bitmap data, which contain own image information. If the appropriate format supports compression, the compression algorithm is used just for this part. Most graphic formats also contain a header, where the most important information like the image resolution or color depth is stored. To add some additional file information, which is not stored in the header, some of files can also use the footer. Depending on the color depth we can also find the color palette block in the bitmap file. Other parts (such as the color correction table) mostly occur in specific files. [1]

Our application supports these following graphic formats for the input: BMP, CR2, EXR, GIF, HDR, J2C, J2K, JP2, JPC, JPEG, JPG, NEF, ORF, PFM, PNG, and RAF. After loading the images, up to 18 test compression algorithms can be performed. The overview of the all supported compression algorithms corresponding to their graphic format is shown in Table I.

Table I: The list of supported graphic formats

Format	Compression
BMP	RLE
EXR	PIZ
EXR	ZIP
GIF	LZW
HD Photo (WDP)	LOSSLESS WAVELET
JBIG	JBIG
JPEG 2000	LOSSLESS WAVELET
JPEG-LS	LOCO-I
PCX	RLE
PNG	DEFLATE 1
PNG	DEFLATE 9
TGA	RLE
TIFF	LZW
TIFF	PACKBITS
TIFF	CCITT FAX 3
TIFF	CCITT FAX 4
WEBP	VP8L Q0, M0
WEBP	VP8L Q100, M6

IV. The Application

The test application was developed under the Windows 7 professional operating system. We used the C++ language in the CodeLite development environment [10] and the MinGW compiler [11]. In order to easily create the classic windows user environment, we also used wxWidgets library [12].

The user interface contains three main menus. The first menu (File) allows us to work with files (Open, Save), contains the compression setting and the “Start testing”, “Save Test Results” and “Quit” commands. While loading/saving all images, the time of these processes is measured because it is included in the time compression efficiency. The compression setting allows us to set the selected compression algorithm for the graphics formats which support different compressions (the BMP, EXR, PNG, and TIFF formats).

The application supports the multiple graphic format files reading, and also can perform multiple tests of the compression algorithms. While testing, the program window displays the progress bar compression and the results are sequentially saved into the output “Test Results” report.

The second menu (Image) allows us to make base color and geometric transformations, so we can change the color depth of any image. There are also commands to make negative images and horizontal/vertical mirrors. The last menu (Help) contains more information about this application.

V. Test Information

The test was performed on a computer with the following configuration: Intel Core i5-430M 2,26GHz Processor, 4GB DDR3 SDRAM 1066 MHz memory, SATA 5400 rpm with 8192 kB Cache hard drive and the Windows 7 Professional

64-bit operating system.

The comparison's results are stored in the tables. The meaning of the abbreviations in these tables is described in Table II.

Table II: The meaning of the abbreviations in the tables

Abbreviation	Meaning
DT (ms)	decompression time
CT (ms)	compression time
CR_{ϕ}	average compression ratio
CR_{min}	minimum compression ratio
CR_{max}	maximum compression ratio
σ_{CR}	standard deviation of the compression ratio
TCE (%/ms)	Time Compression Efficiency is calculated by using (2)

$$TCE = \frac{1-CR}{CT} \cdot 100 \qquad \left[\frac{\%}{ms}\right] \tag{2}$$

The Time Compression Efficiency expresses the percentage reduction in the file size for one millisecond. Its value varies in the interval (-∞; 100) and we get the negative value if the compression is negative (the file size after compression is higher than before compression). Generally, the higher the value means the better time compression efficiency.

VI. Test Results of Photo Processing

The tested photos were divided into the following categories: JPEG photos, RAW photos, grayscale photos, high dynamic range color photos, and high dynamic range grayscale photos.

A. JPEG Photos

The JPEG graphic format is very popular and uses lossy compression. In total, we tested 2,839 JPEG photos with the average number of around 1.8 million pixels. The results of the measurements and calculations are listed in Table III.

Table III shows that, for these photos, the WEBP and JPEG2000 are useful. However, the long compression time is a disadvantage. Thus, if the compression speed is important, it is preferable to use the HD Photo, JPEG-LS, or PNG (DEFLATE 1) graphic formats.

Table III shows one interesting specialty - DEFLATE 1 can achieve better compression ratio for some files than DEFLATE 9. See the values of the maximum compression ratio of the PNG (DEFLATE) compressions - DEFLATE 9 has the maximum compression ratio of 0.919, whereas DEFLATE 1 has only of 0.882. We can see the same case for the VP8L compression of the WEBP format - Q0,M0 parameters have better value of maximum compression ratio than Q100,M6 parameters. So it is clear that it does not always apply the rule "Bigger compression level means better compression ratio" to these compression algorithms.

Table III: Test results of JPEG Photo processing

Graphic Format (Compression)	DT (ms)	CT (ms)	CR_{ϕ}	CR_{min}	CR_{max}	σ_{CR}	TCE (%/ms)
WEBP (L-Q100,M6)	166	19785	0.300	0.032	0.890	0.128	0.0035
WEBP (L-Q0,M0)	164	1925	0.310	0.033	0.887	0.130	0.0358
JP2 (LW)	1423	2009	0.342	0.037	0.826	0.139	0.0328
HDP (LW)	717	817	0.407	0.073	0.850	0.127	0.0725
PNG (D9)	131	3464	0.438	0.043	0.919	0.165	0.0162
JLS (LOCO-I)	453	383	0.444	0.030	0.927	0.159	0.1452
PNG (D1)	142	402	0.469	0.060	0.882	0.161	0.1321
TIFF (LZW)	85	147	0.534	0.078	1.213	0.194	0.3174
EXR (PIZ)	158	336	0.550	0.067	1.019	0.167	0.1341
TGA (RLE)	20	54	0.822	0.080	1.270	0.238	0.3293
EXR (ZIP)	249	2101	0.857	0.081	1.566	0.315	0.0068
TIFF (PACK)	21	58	0.888	0.082	1.329	0.252	0.1939
PCX (RLE)	125	228	0.889	0.064	1.640	0.274	0.0488

B. RAW Photos

These files are uncompressed photos formats that are created by the appropriate camera. Specifically, they generate the CR2, NEF, ORF, and RAF formats. In total, we tested 90 photos with the average number of around 9.1 million pixels. The test results are shown in Table IV. There we can see that these results are close to the JPEG photos test results.

The JPEG-LS and EXR (PIZ) graphic formats have significant improvement in their compression ratios, while WEBP, TGA and TIFF (Packbits) report the deterioration of compression ratios. If we need quick compression, we should prefer the JPEG-LS graphic format, which compresses about 5 times faster than the JPEG2000 and the WEBP (L-Q0,M0).

Table IV: Test results of RAW photo processing

Graphic Format (Compression)	DT (ms)	CT (ms)	CR_{ϕ}	CR_{min}	CR_{max}	σ_{CR}	TCE (%/ms)
JP2 (LW)	5383	8492	0.347	0.207	0.598	0.095	0.0077
WEBP (L-Q100,M6)	876	121219	0.351	0.207	0.630	0.105	0.0005
WEBP (L-Q0,M0)	880	8803	0.357	0.211	0.629	0.104	0.0073
JLS (LOCO-I)	2067	1745	0.375	0.190	0.736	0.118	0.0358
HDP (LW)	3437	3881	0.404	0.303	0.634	0.081	0.0154
PNG (D9)	678	34073	0.419	0.243	0.773	0.111	0.0017
EXR (PIZ)	743	1670	0.455	0.257	0.812	0.120	0.0326
PNG (D1)	723	1933	0.484	0.314	0.784	0.105	0.0267
TIFF (LZW)	362	661	0.523	0.267	0.967	0.158	0.0721
EXR (ZIP)	1318	13775	0.847	0.542	1.300	0.188	0.0011
PCX (RLE)	619	1017	0.894	0.589	1.137	0.107	0.0104
TGA (RLE)	77	278	0.942	0.746	1.002	0.060	0.0210
TIFF (PACK)	96	286	0.994	0.826	1.009	0.030	0.0022

C. Grayscale Photos

We selected 665 photos with the average number of around 1.9 million pixels to test. Because the images are in the grayscale, it is possible to save these images into more graphic formats, so the table with the results (Table V) is larger than in the previous cases.

Table V shows that the JPEG-LS graphic format offers the best compression ratio and great compression time. Also the JPEG 2000 has the good compression ratio, but the compression and decompression times of JPEG 2000 are much worse than of JPEG-LS.

Some photos were saved into some files with negative compression after processing. This concerned the cases of the TIFF (LZW), GIF (LZW), TIFF (Packbits), BMP (RLE), TGA (RLE), EXR (ZIP), and PCX (RLE) graphic formats.

Table V: Test results of grayscale photo processing

Graphic Format (Compression)	DT (ms)	CT (ms)	CR_ϕ	CR_{min}	CR_{max}	σ_{CR}	TCE (%/ms)
JLS (LOCO-I)	125	110	0.483	0.007	0.888	0.139	0.4691
JP2 (LW)	426	683	0.496	0.008	0.898	0.139	0.0737
PNG (D9)	38	1170	0.552	0.004	0.906	0.139	0.0383
HDP (LW)	257	283	0.558	0.134	0.920	0.125	0.1561
JBIG (JBIG)	620	599	0.566	0.006	0.985	0.159	0.0724
PNG (D1)	41	152	0.586	0.011	0.913	0.131	0.2720
EXR (PIZ)	75	140	0.621	0.066	0.985	0.151	0.2709
WEBP (L-Q100,M6)	57	14727	0.626	0.004	0.929	0.147	0.0025
WEBP (L-Q0,M0)	61	662	0.699	0.006	0.976	0.153	0.0454
TIFF (LZW)	29	69	0.711	0.029	1.267	0.196	0.4199
GIF (LZW)	158	218	0.837	0.026	1.306	0.195	0.0750
TIFF (PACK)	10	33	0.898	0.025	1.008	0.141	0.3099
BMP (RLE)	23	18	0.919	0.020	1.023	0.137	0.4500
TGA (RLE)	17	50	0.937	0.026	1.077	0.150	0.1273
EXR (ZIP)	78	532	0.938	0.023	1.539	0.252	0.0117
PCX (RLE)	42	16	0.938	0.038	1.750	0.163	0.3775

D. HDR Color Photos

Only the EXR, HD Photo and TIFF graphic formats support a high dynamic range color therefore we only processed these files with the corresponding compression algorithms in this test.

In total, we tested 66 photos with the average number of around 815 thousand pixels. The test results are shown in Table VI.

We can see that the EXR (ZIP) and EXR (PIZ) compressions are the best. But if it is necessary to have fast compression, we should prefer the PIZ compression algorithm.

Table VI: Test results of HDR color photo processing

Graphic Format (Compression)	DT (ms)	CT (ms)	CR_ϕ	CR_{min}	CR_{max}	σ_{CR}	TCE (%/ms)
EXR (ZIP)	105	566	0.191	0.003	0.398	0.109	0.143
EXR (PIZ)	94	177	0.194	0.011	0.398	0.096	0.457
HDP (LW)	371	397	0.250	0.071	0.413	0.082	0.189
TIFF (DEFLATE)	100	1134	0.328	0.008	0.588	0.172	0.059
TIFF (LZW)	104	224	0.400	0.070	0.675	0.177	0.268
TIFF (PACK)	24	99	0.849	0.317	1.020	0.199	0.152

E. HDR Grayscale Photos

As well as high dynamic range color photos, high dynamic range grayscale photos are supported by the EXR, HD Photo and TIFF graphic formats only.

We tested 72 photos with the average number of around 788 thousand pixels.

As we can see in Table VII, the compression ratio results are similar to the high dynamic range color photos. The only difference is in time-processing because in this case, compression and decompression times are considerably shorter. Also, the best choice is to use the EXR graphic format with the ZIP or PIZ compression algorithm.

Table VII: Test results of HDR grayscale photo processing

Graphic Format (Compression)	DT (ms)	CT (ms)	CR_ϕ	CR_{min}	CR_{max}	σ_{CR}	TCE (%/ms)
EXR (ZIP)	32	153	0.227	0.007	0.429	0.110	0.505
EXR (PIZ)	53	91	0.237	0.033	0.448	0.101	0.838
HDP (LW)	129	139	0.288	0.101	0.401	0.075	0.512
TIFF (DEFLATE)	37	298	0.379	0.014	0.589	0.165	0.208
TIFF (LZW)	37	101	0.459	0.052	0.672	0.181	0.535
TIFF (PACK)	9	61	0.874	0.337	1.023	0.191	0.206

VII. Test Results of Image Processing

In this section, images, especially created using any computer software were tested. For example, images created in the graphical editor or game screenshots.

The tested images were divided into the following categories: 24-bit images, 8-bit images, 4-bit images, 1-bit images, and grayscale images.

A. 24-bit images

We tested 1,936 images with the average number of around 1.3 million pixels. The test results are shown in Table VIII.

The WEBP is the best graphic format for this category because of the lowest compression ratios. If we need fast compression, we can use the JPEG-LS or PNG (DEFLATE 1).

The WEBP, JPEG 2000, PNG, JPEG-LS, and HD Photo do not achieve negative compression ratio in this test.

Table VIII: Test results of 24-bit image processing

Graphic Format (Compression)	DT (ms)	CT (ms)	CR_ϕ	CR_{min}	CR_{max}	σ_{CR}	TCE (%/ms)
WEBP (L-Q100,M6)	126	15621	0.246	0.0005	0.831	0.119	0.005
WEBP (L-Q0,M0)	123	1950	0.258	0.0005	0.873	0.124	0.038
JP2 (LW)	769	1257	0.308	0.0011	0.898	0.122	0.055
PNG (D9)	84	3649	0.350	0.0020	0.921	0.162	0.018
JLS (LOCO-I)	304	256	0.378	0.0022	0.932	0.145	0.243
PNG (D1)	91	251	0.395	0.0062	0.885	0.155	0.241
HDP (LW)	494	568	0.397	0.038	0.987	0.125	0.106
TIFF (LZW)	59	108	0.451	0.022	1.515	0.188	0.509
EXR (PIZ)	116	242	0.485	0.016	1.004	0.155	0.213
TGA (RLE)	15	43	0.758	0.010	1.388	0.224	0.564
EXR (ZIP)	174	1467	0.776	0.011	1.601	0.283	0.015
PCX (RLE)	91	151	0.792	0.013	3.858	0.281	0.137
TIFF (PACK)	16	52	0.931	0.021	1.469	0.182	0.133

B. 8-bit images

We tested 1,512 images with the average number of around 1.2 million pixels. The test results are shown in Table IX.

Table IX: Test results of 8-bit image processing

Graphic Format (Compression)	DT (ms)	CT (ms)	CR_ϕ	CR_{min}	CR_{max}	σ_{CR}	TCE (%/ms)
WEBP (L-Q100,M6)	35	8410	0.423	0.0033	0.822	0.170	0.007
WEBP (L-Q0,M0)	41	590	0.476	0.012	0.874	0.182	0.089
PNG (D9)	14	318	0.503	0.0073	0.946	0.212	0.156
PNG (D1)	16	54	0.526	0.015	0.947	0.199	0.879
GIF (LZW)	89	123	0.593	0.0081	1.298	0.236	0.331
TIFF (LZW)	17	44	0.624	0.028	1.606	0.236	0.860
TIFF (PACK)	8	25	0.782	0.028	2.017	0.252	0.883
TGA (RLE)	13	33	0.792	0.025	1.075	0.253	0.632
PCX (RLE)	29	11	0.794	0.040	1.299	0.269	1.806
BMP (RLE)	23	14	0.797	0.025	1.044	0.230	1.477
JP2 (LW)	834	1370	1.312	0.014	2.709	0.528	-0.023
JLS (LOCO-I)	247	201	1.352	0.031	2.756	0.573	-0.175
HDP (LW)	458	520	1.536	0.124	2.748	0.468	-0.103
EXR (PIZ)	103	209	1.577	0.069	2.846	0.557	-0.276
EXR (ZIP)	150	1094	2.180	0.043	4.464	1.006	-0.108

In this category, the WEBP (L-Q100,M6) has the best compression ratios. But also the WEBP (L-Q0,M0) has great compression ratios and is more than 14 times faster than the WEBP (L-Q100, M6).

If we need fast compression, it is the best to choose the PNG (DEFLATE), which is up to 6 times faster than the WEBP (L-Q0,M0). And also, we can use the PNG (DEFLATE) for fast decompression.

The WEBP and PNG are only ones, which do not achieve negative compression ratios. Negative compression ratios most often appear for the JPEG 2000, JPEG-LS, HD Photo, and EXR - This is most because we have to change the bit depth to a higher value than it is needed. Those four graphic formats should not be used for 8-bit images.

C. 4-bit images

We tested 635 images with the average number of around 800 thousand pixels. The test results are shown in Table X.

In this category, the WEBP (L-Q100, M6) is the best compression in terms of compression ratios. However, the WEBP (L-Q100, M6) has much longer compression time than the other compressions on top of the Table X. If the longest compression time of the WEBP (L-Q100, M6) is a problem, the better option is to use the PNG (DEFLATE 9) or the WEBP (L-Q0,M0). The better option can be the WEBP (L-Q0,M0), because only the VP8L compression of the WEBP graphic format do not achieve negative compression ratios. But on the other hand, the PNG (DEFLATE 9) has faster compression and decompression than the WEBP (L-Q0,M0).

If we need fast compression, it is appropriate to use the GIF (LZW) or the PNG (DEFLATE 1). The GIF (LZW) achieves the third best maximum compression ratio and the fourth best average compression ratio. On the other hand, the PNG (DEFLATE 1) is even three times faster than the GIF (LZW) and its compression ratio is worse only by 1%.

Table X: Test results of 4-bit image processing

Graphic Format (Compression)	DT (ms)	CT (ms)	CR_ϕ	CR_{min}	CR_{max}	σ_{CR}	TCE (%/ms)
WEBP (L-Q100,M6)	14	1805	0.376	0.0091	0.847	0.165	0.035
PNG (D9)	5	293	0.407	0.013	1.153	0.179	0.202
WEBP (L-Q0,M0)	15	449	0.413	0.011	0.882	0.176	0.131
GIF (LZW)	27	33	0.431	0.027	1.015	0.178	1.734
PNG (D1)	6	13	0.441	0.024	1.153	0.175	4.233
TIFF (LZW)	5	23	0.487	0.044	2.831	0.247	2.234
TIFF (PACK)	4	13	0.695	0.104	2.723	0.255	2.396
PCX (RLE)	10	4	0.700	0.109	1.226	0.255	8.221
TGA (RLE)	8	19	1.240	0.105	6.385	0.537	-1.257
BMP (RLE)	20	11	1.322	0.106	8.538	0.640	-2.985
EXR (PIZ)	65	128	2.409	0.233	6.550	0.852	-1.103
JLS (LOCO-I)	114	91	2.722	0.126	5.498	1.037	-1.895
EXR (ZIP)	86	448	2.760	0.086	5.811	1.330	-0.393
JP2 (LW)	594	984	2.969	0.203	5.993	1.081	-0.200
HDP (LW)	285	326	3.746	0.452	10.324	1.058	-0.843

D. 1-bit images

We tested 1,915 images with the average number of around 1.0 million pixels. The test results are shown in Table XI.

The JBIG is clearly the best compression in this category. Also the WEBP is worth mentioning – the WEBP do not achieve, same as the JBIG, negative compression ratio and minimum compression ratio of the WEBP is even 6 times better than of the JBIG.

It is hard to say whether or not we should use the PNG (DEFLATE 1) if we need fast compression. The PNG (DEFLATE 1) has 6 times faster compression algorithm than the JBIG, but compression ratio of the PNG (DEFLATE 1) is worse by about 17% than of the JBIG.

It is positive surprise that the JPEG-LS and the EXR (ZIP) achieve a non-negative average compression ratio, although they need to increase the bit depth.

It is negative surprise that that the TIFF (CCITT3) and TIFF (CCITT4) compressions achieve negative average compression ratio. But it is necessary to take into account of the fact that these compression algorithms were mainly created for the compression of the text content of an image.

Table XI: Test results of 1-bit image processing

Graphic Format (Compression)	DT (ms)	CT (ms)	CR_ϕ	CR_{min}	CR_{max}	σ_{CR}	TCE (%/ms)
JBIG (JBIG)	29	32	0.317	0.0028	0.985	0.226	2.164
WEBP (L-Q100,M6)	12	701	0.390	0.00046	0.988	0.224	0.087
WEBP (L-Q0,M0)	12	406	0.412	0.00046	0.988	0.227	0.145
PNG (D9)	3	77	0.452	0.0076	1.945	0.281	0.710
PNG (D1)	3	5	0.482	0.012	1.945	0.269	10.821
GIF (LZW)	21	27	0.498	0.025	1.176	0.275	1.876
TIFF (LZW)	3	19	0.583	0.043	3.405	0.402	2.246
JLS (LOCO-I)	21	20	0.653	0.015	1.741	0.456	1.760
TIFF (PACK)	2	10	0.705	0.051	3.243	0.372	2.851
PCX (RLE)	3	2	0.717	0.058	1.811	0.324	18.214
EXR (ZIP)	34	100	0.745	0.046	4.284	0.479	0.255
HDP (LW)	86	97	0.978	0.031	3.063	0.595	0.023
TIFF (CCITT4)	7	22	1.011	0.0062	3.529	0.929	-0.051
TIFF (CCITT3)	6	20	1.038	0.018	3.486	0.899	-0.189
EXR (PIZ)	42	72	1.230	0.122	6.892	0.911	-0.320
TGA (RLE)	9	21	2.881	0.137	11.216	2.507	-8.991
BMP (RLE)	21	10	3.340	0.083	15.000	2.419	-23.288
JP2 (LW)	266	444	3.985	0.090	7.998	2.471	-0.673

E. Grayscale images

We tested 436 images with the average number of around 3.6 million pixels. The test results are shown in Table XII.

The best graphic formats for the grayscale images are the PNG, JPEG-LS, and WEBP. For fast compression we can choose the JPEG-LS and the PNG (DEFLATE 1). The compression ratios of the first four compressions in Table XII are not significantly different.

The PNG, JPEG-LS, WEBP, JPEG 2000, and HD Photo graphic formats do not achieve negative compression ratio. But the HD Photo is not recommended to use for this kind of images because average compression ratio is not too good.

Table XII: Test results of grayscale image processing

Graphic Format (Compression)	DT (ms)	CT (ms)	CR_ϕ	CR_{min}	CR_{max}	σ_{CR}	TCE (%/ms)
PNG (D9)	59	1441	0.367	0.0042	0.932	0.254	0.044
JLS (LOCO-I)	167	148	0.376	0.014	0.929	0.220	0.421
WEBP (L-Q100,M6)	67	10815	0.378	0.0014	0.952	0.293	0.0058
PNG (D1)	61	223	0.392	0.0093	0.935	0.255	0.272
WEBP (L-Q0,M0)	74	700	0.411	0.0037	0.983	0.318	0.084
JBIG (JBIG)	971	915	0.413	0.0062	1.036	0.265	0.064
JP2 (LW)	787	1291	0.440	0.014	0.957	0.211	0.043
TIFF (LZW)	46	96	0.495	0.035	1.330	0.347	0.526
EXR (PIZ)	117	221	0.529	0.039	1.114	0.248	0.213
GIF (LZW)	195	283	0.548	0.016	1.332	0.372	0.160
HDP (LW)	446	498	0.565	0.115	0.981	0.198	0.087
TIFF (PACK)	14	40	0.588	0.032	1.008	0.354	1.030
EXR (ZIP)	138	755	0.633	0.023	1.654	0.419	0.049
TGA (RLE)	27	83	0.634	0.032	1.071	0.361	0.439
BMP (RLE)	33	25	0.648	0.027	1.125	0.345	1.401
PCX (RLE)	72	25	0.682	0.047	1.816	0.405	1.276

VIII. Test Results of Text Image Processing

In this test, images always contain text information. In some few cases, images contain not only a text, but also a picture.

The tested images were divided into the following categories: 1-bit text images and grayscale text images.

A. 1-bit text images

We tested 357 images with the average number of around 2.9 million pixels. The test results are shown in Table XIII.

It is the best to use the JBIG compression for this category. This compression achieves the best average, minimum and maximum compression ratio.

If fast compression is requested, it is better to use the TIFF (CCITT4) compression that is more than three times faster than the JBIG. On the other hand the JBIG compression is not negative. However, the TIFF (CCITT4) and TIFF (CCITT3) formats achieves negative compression ratio only for text images with pictures.

Another option for fast compression is the PNG (DEFLATE 1), which is 2 times faster than the TIFF (CCITT4) and do not achieve negative compression in this test.

Table XIII: Test results of 1-bit text image processing

Graphic Format (Compression)	DT (ms)	CT (ms)	CR_ϕ	CR_{min}	CR_{max}	σ_{CR}	TCE (%/ms)
JBIG (JBIG)	58	67	0.126	0.017	0.552	0.054	1.306
PNG (D9)	5	147	0.197	0.032	0.627	0.072	0.545
TIFF (CCITT4)	7	20	0.198	0.024	1.219	0.112	4.024
WEBP (L-Q100,M6)	23	997	0.199	0.026	0.641	0.076	0.080
WEBP (L-Q0,M0)	25	272	0.206	0.029	0.641	0.075	0.291
PNG (D1)	4	9	0.228	0.041	0.662	0.076	8.945
TIFF (LZW)	4	21	0.247	0.057	0.792	0.085	3.549
GIF (LZW)	45	63	0.249	0.044	0.723	0.084	1.198
TIFF (CCITT3)	7	19	0.257	0.052	1.324	0.123	3.864
JLS (LOCO-I)	25	23	0.277	0.043	0.933	0.104	3.203
EXR (ZIP)	94	211	0.328	0.082	1.002	0.110	0.319
TIFF (PACK)	2	11	0.334	0.076	0.740	0.103	5.919
PCX (RLE)	6	2	0.440	0.097	0.946	0.139	23.229
HDP (LW)	220	248	0.538	0.085	1.707	0.199	0.186
EXR (PIZ)	103	177	0.560	0.182	1.486	0.173	0.248
TGA (RLE)	20	55	0.929	0.211	3.845	0.396	0.129
BMP (RLE)	37	15	1.232	0.166	4.167	0.525	-1.573
JP2 (LW)	617	1080	2.083	0.276	6.846	0.776	-0.100

B. Grayscale text images

We tested 319 images with the average number of around 2.9 million pixels. The test results are shown in Table XIV.

Table XIV: Test results of grayscale text image processing

Graphic Format (Compression)	DT (ms)	CT (ms)	CR_ϕ	CR_{min}	CR_{max}	σ_{CR}	TCE (%/ms)
JLS (LOCO-I)	212	190	0.377	0.140	0.623	0.081	0.328
JP2 (LW)	783	1269	0.433	0.176	0.622	0.078	0.045
JBIG (JBIG)	865	833	0.449	0.146	0.716	0.092	0.066
PNG (D9)	67	3034	0.459	0.173	0.697	0.085	0.018
WEBP (L-Q100,M6)	113	12340	0.508	0.157	0.751	0.108	0.004
PNG (D1)	72	249	0.512	0.200	0.718	0.076	0.196
HDP (LW)	405	444	0.514	0.281	0.702	0.070	0.109
EXR (PIZ)	134	232	0.534	0.220	0.735	0.085	0.201
TIFF (LZW)	47	90	0.561	0.205	0.934	0.121	0.486
WEBP (L-Q0,M0)	128	1377	0.564	0.162	0.817	0.101	0.032
GIF (LZW)	206	304	0.567	0.171	0.931	0.123	0.142
EXR (ZIP)	115	672	0.638	0.283	1.027	0.139	0.054
TIFF (PACK)	17	39	0.891	0.284	1.005	0.074	0.277
TGA (RLE)	27	77	0.930	0.298	1.060	0.081	0.091
BMP (RLE)	46	27	0.945	0.303	1.007	0.061	0.204
PCX (RLE)	70	28	1.219	0.398	1.751	0.199	-0.769

In this test, the JPEG-LS graphic format far exceeds all other compression algorithms in both quality and compression time.

The EXR (ZIP), TIFF (PACKBITS), TGA, BMP, and PCX formats achieve negative compression ratio at least in one case. It is clear in terms of the compression ratio of these formats that it is not appropriate to use them for images tested in this category.

IX. Conclusion

The application, which was developed at our faculty last year, compares the results of lossless compression algorithms. In order to find the suitable lossless compression algorithm and corresponding graphic file, we created some tests for photos, images and text images.

The WEBP format with the VP8L compression is the best choice for photos, 24-bit, 8-bit, and 4-bit images. The VP8L has the excellent compression ratio, whose values are often the best of all tested cases. The WEBP does not achieve too good compression ratios for grayscale photos and grayscale text images. The only flaw of the WEBP is the compression time for the Q100,M6 parameters which belongs to the longest in all tests. Because of that, it is better to use the Q0,M0 parameters instead.

We cannot completely support the statement: "WEBP lossless images are 26% smaller in size compared to PNGs" [7]. We can only say that this statement applies mostly in JPEG photos and 24-bit images. For RAW photos, 8-bit images, and 1-bit images the WEBP is smaller in size compared to the PNG but by less than 26%. For 4-bit images it depends to settings of the compression parameters of the WEBP and of the PNG, but generally their compression ratios are almost identical. For grayscale photos, grayscale images, 1-bit text images, and grayscale text images the PNG has better results than the WEBP.

We failed to demonstrate the statement: "WEBP decoding speeds faster than PNG have been demonstrated" [13]. On the contrary the PNG is faster than the WEBP in our tests. But the WEBP is still in development, so it can be more optimized or changed in other way in future. [7]

The test results show that the JPEG 2000 graphic format with the integer wavelet transform algorithm is the great choice for photos or 24-bit images and good choice for grayscale photos or grayscale images. This transformer has the excellent compression ratio and the only drawback is the relatively long compression time.

For high dynamic photo processing, it is suitable to use the EXR graphic format with the PIZ or ZIP compression algorithm. Both of them have the approximately equal compression ratio, however, the PIZ algorithm is slightly faster.

The HD Photo achieves good results for any kind of photos including high dynamic range photos.

If both quality and fast compression is requested, it is the

best to use the LOCO-I compression of the JPEG-LS format. Although this compression achieves good results for photos and 24-bit images, quality of this compression is manifested mainly in greyscale images - irrespective of whether they are photos, images created using any computer software or images containing the text.

The PNG (DEFLATE) is the most appropriate to use for grayscale images created using any computer software and 4-bit images. DEFLATE compression levels are not much different, so a user himself can make decision about choosing the compression level. The PNG is the most universal format of all - the compression algorithm of this format belongs to one of the best in the compression ratio, decompression time, and compression time (valid only for small compression level) in the all tested categories.

The LZW compression in the GIF and TIFF formats should be used only for images with the bit depth less than 8. The LZW compression is neither bad nor great for other types of images. Generally we can say that it is better to prefer the GIF (LZW) compression, which can reduce the file size more than the TIFF (LZW). It is better to use the TIFF (LZW) than the GIF (LZW) only for grayscale images (with or without text).

Although the JBIG compression supports greyscale images, these images are not its priority. The excellent compression ratio is achieved with 1-bit images - irrespective of whether it is purely images, text images or both.

CCITT4 and CCITT3 compression results confirmed that their primary use is 1-bit images containing a text, especially the CCITT4 compression makes very good values. However, if the text is not in the image, they often produce negative compression. The compression and decompression times are identical for both.

The JPEG 2000, as well as HD Photo, EXR, JPEG-LS are not suitable for use on color images with the bit depth of 8 or less.

The RLE compression of the PCX, TGA, BMP and the PACKBITS compression of the TIFF achieve the poor compression ratio in the all categories and therefore they are not recommended to use.

REFERENCES

[1] J. D. Murray and W. Vanrypper, *Encyclopedia of Graphics File Formats: The Complete Reference*, 2nd edition, O'Reilly Media, 1996, 1152 p., ISBN 1565921615.

[2] D. Salomon, *Handbook of Data Compression*, 5th edition, Springer, 2010, 1383 p., ISBN 1848829027.

[3] K. Sayood, *Lossless Compression Handbook*, 1st. edition, Academic Press, 2003, 454 p., ISBN 0126208611.

[4] F.Alexa, V. Gui, C. Caleanu and C. Botoca, "Lossless Data Compression Using Neural Networks", Proceedings of the 7th WSEAS International Conference on CIRCUITS, SYSTEMS, ELECTRONICS, CONTROL and SIGNAL PROCESSING, Canary Islands, 2008, pp.128-132.

[5] B. Carpentieri, "Interactive compression of books", (2010). WSEAS Transactions on Computers, 9 (3), pp. 278-287.

[6] B. Carpentieri, "Dictionary Based Compression for Images", *International Journal of Computers*, Issue 3, Volume 6, 2012, pp. 187-195

[7] Google, Inc. (2013, September 26). WebP: A new image format for the Web [Online]. Available: https://developers.google.com/speed/webp/

[8] S.Atek, T.Vladimirova. "A New Lossless Compression Method for Small Satellite On-Board Imaging" Proceeding of the 3rd WSEAS International Conference on Applied and Theoretical Mathematics, Miedzyzdroje, Poland, September 1-5, 2002, ed. N.E.Mastorakis, pp. 1871-1876.

[9] F. PENSIRI and S. AUWATANAMONGKOL, "A lossless image compression algorithm using predictive coding based on quantized colors", in *WSEAS TRANSACTIONS on SIGNAL PROCESSING*, Issue 2, Volume 8, April 2012, pp. 43-53

[10] CodeLite contributors. (2013). CodeLite – an Open-Source, cross-platform IDE for C/C++. [Online]. Available: http://codelite.org

[11] MinGW contributors. (2013). MinGW – Minimalist GNU for Windows. [Online]. Available: http://www.MinGW.org

[12] wxWidgets contributors. (2013). wxWidgets – Cross-Platform GUI Library. [Online]. Available: http://wxWidgets.org

[13] J. Alakuijala. (2012, August 16). Lossless and Transparency Encoding in WebP [Online]. Available: https://developers.google.com/speed/webp/docs/webp_lossless_alpha_study