Practical Analysis of TIFF File Size Reductions Achievable Through Compression

Peter May
British Library
96 Euston Road
London
Peter.May@bl.uk

Kevin Davies
British Library
Boston Spa
West Yorkshire
Kevin.Davies@bl.uk

ABSTRACT

This paper presents results of a practical analysis into the effects of three main lossless TIFF compression algorithms – LZW, ZIP and Group 4 – on the storage requirements for a small set of digitized materials. In particular we are interested in understanding which algorithm achieves a greater reduction in overall storage, and whether there is any variation based on the type of file (e.g. colour depth). We compress 503 files with two software utilities – ImageMagick and LibTiff – and record the resulting file size for comparison against the original uncompressed version. Overall we find that in order to effectively (although not necessarily optimally) reduce total storage, Group 4 compression is most appropriate for 1-bit/pixel images, and ZIP compression is suited to all others. We also find that ImageMagick – which uses the LibTiff library – typically out-performs LibTiff with respect to compressed file sizes, noting that this appears to be the result of setting the "Predictor" tag.

Keywords

TIFF; Compression; LZW; ZIP; Group 4; ImageMagick; LibTiff

1. INTRODUCTION

Tagged Image File Format (TIFF) is considered the de facto format for preservation master image files, providing a simple tagged structure for storing raster image pixel data along with associated metadata. The format itself is stable and well documented, with the specification having not seen a major revision since 1992 [9]. It is also widely adopted, both in terms of graphics software and in terms of Library and Archive adoption. The British Library is no exception to this, having received around 5 million TIFF files through our Endangered Archives Programme alone.

TIFF files can be very large, however, leading to storage cost problems for big collections, and potentially impacting on the long-term preservation of these and other collections for financial reasons; the larger the files, the fewer that can be stored within a defined storage system.

One approach to mitigate this, whilst retaining use of the TIFF format, would be to compress the image payload data. TIFF enables this by supporting a variety of different compression algorithms, such as the lossless Group 4, LZW and ZIP algorithms. Being lossless, these algorithms all enable a TIFF image to be reduced in size, with the image data being fully retrievable at a later stage (through decompression).

From a storage perspective though, it is not clear what impact each of these compression approaches has on the overall size of a stored TIFF collection, particularly for the types of digitized files held by libraries and other memory institutions. Does one algorithm compress the files to a greater extent than another? Are different algorithms suited to different types of file?

In addition to this, compression is applied through the use of a particular software application/library, such as ImageMagick[1] or LibTiff[2]. Does the choice of software impact on the amount of compression achievable?

This paper reports on a practical experiment performed at the British Library analyzing the effects of LZW and TIFF compression on the storage size of a small set (503 files) of digitised material. It focuses on the average file sizes achievable through these compression algorithms, across different image colour depths, and through using two popular and freely available software utilities for performing the compression (the previously mentioned LibTiff and ImageMagick).

We start by briefly outlining the background to TIFF files, their overall structure and details about community recommendations on the use of TIFF files, particularly with respect to compression. Section 3 then describe the experimental methodology applied, covering details about the process, hardware and software, and the dataset used. The results are presented and analysed in Section 4, with discussion about what this means in practice outlined in Section 5.

2. TIFF FILES AND THEIR USE

TIFF is a bitmap image format originally created by the Aldus Corporation in the mid-1980's, but now owned by Adobe after they acquired Aldus in 1994 [9]. It evolved from a bitonal format to encompass grayscale and full-colour image data, as well as support for a variety of compression technologies.

The current specification (revision 6) [9] is split into two parts; part 1 describes baseline TIFF, which covers the core parts essential for TIFF readers to support. Part 2 covers extensions to the baseline, covering features which may not be supported by all TIFF readers.

2.1 Structure and Compression

TIFFs are tag-based in structure. They start with an 8 byte header which contains an offset value to the first Image File Directory (IFD) containing tags (such as height, width, image data location, etc.) and associated values pertaining to the first image within the file. An offset tag to the next IFD allows another sub-image to be included (e.g. the next page, or a thumbnail) in the same manner, and so on. Baseline TIFF readers are not required to read beyond the first IFD however.

In addition to providing the location of the image data within the file, tags also provide details about the compression applied to that data. It should be noted that, as stated in the TIFF specification, "Data compression applies only to raster image data. All other TIFF fields are unaffected" [9].

Baseline rev. 6 TIFF images can be compressed using either the lossless Modified Huffman Compression algorithm for bi-level images, or the lossless PackBits compression algorithm (both are described in the specification). Extensions to the baseline

[1] http://www.imagemagick.org/script/index.php

[2] http://www.libtiff.org/

TIFF define additional compression schemes though: Group 3 and Group 4 for bitonal images (both lossless), as well as LZW (Lempel-Ziv & Welch; lossless) and JPEG (lossy). Compression enhancements for ZIP (Deflate/Inflate; lossless) and 'new style' JPEG were specified in supplementary TIFF Technical Notes [1].

LZW was originally defined as a baseline compression scheme in TIFF version 5, but was moved to the extensions section in TIFF version 6 due to licensing issues surrounding LZW patents. These patents expired in 2003/2004 (US/Europe respectively) [3] effectively removing the need for legal-related restrictions on the use of LZW compression [10].

2.2 Community Guidelines on use of TIFFs

TIFF files are widely used in libraries and archives as master files for digitized still images. Recommendations for their use for this purpose are quite consistent, typically recommending uncompressed or LZW compressed images.

The Succeed Project assessed existing digitization recommendations, providing a summary of these and consolidating them into their own recommendations [5]. TIFF v6 was the recommended master file format for still images, either uncompressed or using LZW compression.

The U.S. National Archives and Records Administration (NARA) Technical Guidelines for Digitizing Archival Materials for Electronics Access suggest LZW or ZIP lossless compression could possibly be used in an actively managed digital repository. JPEG compression should not be used [5].

The same LZW or ZIP compression recommendation is also true for the Federal Agencies Digitization Guidelines Initiative (FADGI) 2010 guidelines for digitizing cultural heritage materials (although uncompressed is preferred) [7]. This is unsurprising given they essentially derive from the NARA guidelines.

Other guidelines are more restrictive on the use of compression, effectively prohibiting it. For example, the National Digital Newspaper Program (NDNP) guidelines state that master page images should be delivered as uncompressed TIFF v6.0 files, and supported by derivative JPEG2000 files for end user access [8].

The British Library's internal digitization guidelines are also consistent with those from the wider community, recommending no compression or LZW compression for TIFF (v6) files.

These guidelines appear to be trying to balance long-term preservation accessibility (though minimizing complications by using no compression) with reduced storage (through lossless compression). In terms of storage reduction however, it is not always clear from the recommendations why a particular algorithm is chosen. More so, if the aim of recommending compression is to reduce storage requirements, is the algorithm choice sufficient?

Gillesse *et al*, at The National Library of the Netherlands (KB) undertook a research project looking at potential alternatives to TIFF Master Image files, comparing LZW compressed TIFF with JPEG2000, PNG and JPEG [2]. They found that based on their two test sets of ~100 originals, "it appears that TIFF LZW in lossless mode can yield a benefit of about 30% compared to an uncompressed file" [2]. This is a useful indication of the amount of storage that can be saved but, being derived from a small test sample, how accurate is it? And what variation, if any, is there based on the type of content tested?

Evidence is not easy to find, and is often embroiled in other investigations and disciplines, particularly medical related [4]. Anecdotal evidence available on the internet[3] suggests that we should expect variation in the compressibility of files based on the amount of detail within the image and the colour depth. However such reports typically only test a handful of files, and provide limited – if any – detail of the methodology taken; hardly conclusive evidence.

3. METHODOLOGY

Figure 1 depicts the overall process used to compress the set of files described below using LZW and ZIP algorithms, interrogate the files to obtain relevant image properties, and compile the results into a CSV file suitable for detailed analysis. This process was automated through a shell script.

3.1 Data Set

503 TIFF images were randomly taken from our Endangered Archive Programme's submissions. These comprised a variety of bit-depth images as detailed in Table 1, and covered a broad range of categories such as books, magazines, microfilm, newspapers and photographs.

Table 1: Sample set details grouped by bit-depth

Bit Depth	File Count	Group 4 Compressed	Total Size
1	56	Yes	5.5 MiB
8	57	-	1231.1 MiB
24	345	-	8512.3 MiB
48	45	-	1636.4 MiB
Total:	*503*		*11385.3 MiB*

3.2 Data Preparation

As can be seen, the sample of TIFF files used were largely uncompressed; the only compressed files were a selection of 1-bit/pixel microfilm records, compressed using the Group 4 algorithm. These files were first decompressed using the 'tiffcp' utility before the main conversion was performed.

3.3 Compression Software

Uncompressed TIFFs are compressed (and subsequently decompressed), as shown in Figure 1, using either the ImageMagick or LibTiff versions mentioned below. These software utilities were chosen as they are commonly used for image file manipulation, particularly on Linux environments. In both cases, standard installations and default settings are used.

Other versions of these utilities, and other graphics software such as Photoshop, have not been investigated.

ImageMagick (6.6.9.7-5ubuntu3.4):

Used to compress and decompress files using ZIP and LZW algorithms. It was also used to obtain image properties such as bit depth, dimensions and number of unique colours.

- convert -compress zip "<inputfile>" "<outputfile>"
- convert -compress lzw "<inputfile>" "<outputfile>"
- convert -compress none "<inputfile>" "<outputfile>"

Note: ImageMagick depends on LibTiff (using the same version as below, in our case) for TIFF manipulation. As we will see, results still vary between standalone LibTiff and ImageMagick.

[3]http://havecamerawilltravel.com/photographer/tiff-image-compression

http://www.scantips.com/basics9t.html

http://videopreservation.conservation-us.org/tjv/index.html

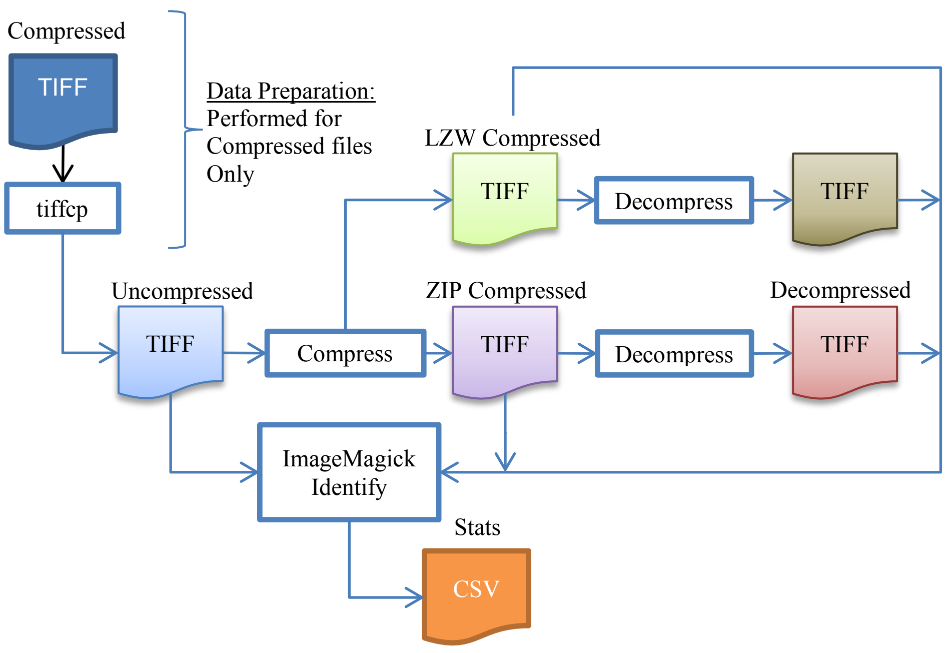

Figure 1: The process used to compare file sizes between the uncompressed, compressed and decompressed TIFFs.

LibTiff (libtiff4-3.9.5-2ubuntu1.9):

LibTiff's 'tiffcp' utility was also used to compress/decompress files, and to remove existing Group 4 compression from files.

- tiffcp -c zip "<inputfile>" "<outputfile>"
- tiffcp -c lzw "<inputfile>" "<outputfile>"
- tiffcp -c none "<inputfile>" "<outputfile>"

Note: Group 4 compressed images were taken as is from the original sample, and not recompressed.

3.4 Hardware

The compression and analysis process was executed on an Ubuntu 12.04.2 64bit (Kernel: 3.5.0-54-generic x86_64) VM running on a HP ProLiant DL385p Gen8 server. The VM was allocated 1 CPU (2.294GHz), ~6GB of RAM and ~500GB storage.

3.5 Entropy Calculations

As part of our analysis we calculated the average image entropy as a measure of how "busy" an image is. To do this we used Fred Weinhaus'[4] 'entropy' script, which uses ImageMagick to extract the histogram for each colour channel of an image, and determine the distribution of each colour value within that histogram. Normalisation of these distributions gives an entropy value of between 0 (a solid plane of colour) and 1 (a uniform gradient of all possible colour) for each channel. The average of the entropy values for all colour channels is used as the average entropy for the image.

3.6 Uncompressed vs. Decompressed Pixels

Pixel data from the original uncompressed TIFF files and the decompressed files were compared using ImageMagick's 'compare' command with '–metric ae' option, which measures the number of pixels differing between the two images. In all cases, pixel data in the original and decompressed TIFFs was identical.

3.7 Reported File Sizes

Compression of a TIFF file is applied to the image payload only, however changes will often occur within other areas of the file (i.e. the tags) to describe this compression. Furthermore, software libraries applying the compression may affect, for example remove, other metadata within the file. File sizes reported in this paper are for the complete file, encompassing all changes made by the application of compression, as this best reflects the total storage requirements. From a preservation perspective however, all changes caused by compression should be considered.

[4]http://www.fmwconcepts.com/imagemagick/entropy/index.php

4. RESULTS

This section presents the results from experimentation on the specified sample of collection material, with an analysis of the main findings.

The results are organized in a logical order following the process diagram shown in Figure 1 evaluating:

- Original Group 4 compressed files compared to their uncompressed "original" form.
- LZW/ZIP compressed files compared to their uncompressed "original" form.
- LZW/ZIP compressed files compared to their Group 4 compressed form (for 1-bit files).

4.1 Group 4 Compressed vs. Uncompressed File Sizes

Of the original sample of files, all 56 of the 1-bit TIFFs were found to be compressed with Group 4 compression. The initial step in our process decompressed these to present a uniform, uncompressed sample of files.

Table 2 shows the mean average ratio in file sizes between the Group 4 compressed files and their uncompressed counterparts. LibTiff's "tiffcp" utility was always used to decompress originally compressed TIFF files, and so no comparable results are available for the ImageMagick tool.

Table 2: Minimum, mean and maximum ratio of Group 4 file sizes with respect to their uncompressed size (to 1 d.p.)

Min	Mean	S.D.	Max
0.05%	12.68%	13.86%	47.15%

As can be seen, and as to be expected, the 1-bit Group 4 compressed TIFFs are smaller than their uncompressed counterparts, averaging ~13% of the uncompressed size. At most, the least compressed file is still over 50% smaller than its uncompressed form.

Summary:

- Group 4 compressed files appear to be at least half the size of their uncompressed form.

4.2 LZW/ZIP Compressed vs. Uncompressed File Sizes

With all 56 Group 4 files de-compressed, the 503 uncompressed files become the base sample for further compression analysis. These are compressed using either LibTiff's 'tiffcp' command or ImageMagick's 'convert' command, and the file sizes recorded. Table 4 shows, for both software libraries, the minimum, maximum and mean average file sizes (in MiB[5]) for the original uncompressed files, and the resulting LZW or ZIP compressed files.

4.2.1 Effect of Compression Algorithm

With respect to compression algorithm, Table 4 shows three things. Firstly, irrespective of bit-depth and software utility, both ZIP and LZW compression generate compressed files with a mean average size smaller than the original uncompressed files.

This is also highlighted in Table 3, which indicates that LZW files are an average of ~51% or ~70% the size of the uncompressed original (for ImageMagick and LibTiff respectively), and ZIP files are an average of ~44% or 58% (respectively).

Ratios are calculated on a file-by-file basis across the entire 503 uncompressed sample files before averaging.

Table 3: Ratio of LZW/ZIP compressed file sizes as a percentage of the original uncompressed files (to 1 d.p.)

Library	Alg.	Min	Mean	S.D.	Max
ImageMagick	LZW	2.3%	51.2%	26.8%	133.8%
	ZIP	0.5%	43.9%	21.1%	99.4%
LibTiff	LZW	2.0%	69.8%	27.6%	130.0%
	ZIP	0.5%	58.2%	24.3%	98.5%

Secondly, generating smaller files from the use of compression is not guaranteed. A maximum compressed-to-uncompressed ratio being greater than 100%, as seen in Table 3, indicates that there are incidents where applying LZW compression using either software utility actually increases the file size. This predominantly affects 24-bit images in our sample, as summarised in Table 5, with 28 compressed images being larger than the original when using ImageMagick, compared to 68 when using LibTiff.

Table 4: Count of files, per bit depth and software library, whose LZW compressed size is greater than their original uncompressed size

Bit Depth	File Count	
	ImageMagick	LibTiff
8	-	1
24	28	68

Why should LZW compression increase the file size however? The original, and common, choice for LZW code table is to store 4096 entries, requiring 12-bits to encode every entry (2^12=4096). The initial 256 entries are reserved for the single byte values 0-255, while the remaining entries correspond to multi-byte sequences. Savings are made when sequences of bytes in the original file can be encoded by one 12-bit code; however, if this is not possible, then the 12-bit code for individual bytes is used instead, adding a 50% overhead to each byte. This is a simplified example, but illustrates the point of how LZW could create larger files.

Thirdly, these results highlight that, again irrespective of bit-depth and software utility, ZIP compression generates an average compressed file size smaller than that produced with LZW compression. This appears to be consistently true for our tested sample. Comparing the ratio of ZIP to LZW compressed file sizes on a file-by-file basis (shown in Table 6), ZIP compressed files are between ~22% and ~96% the size of LZW compressed files, with an average of ~84%. No individual ZIP file has therefore exceeded the size of the corresponding LZW file; if it had, the maximum ratio would have been larger than 100%.

Table 5: Minimum, maximum and mean average file sizes[5] for each colour depth grouping (to 1 d.p.; † to 1 s.f.).

		ImageMagick				LibTiff			
Bit Depth	TIFF*	Min (MiB)	Mean (MiB)	S.D. (MiB)	Max (MiB)	Min (MiB)	Mean (MiB)	S.D. (MiB)	Max (MiB)
1	Original	0.7	0.9	0.6	3.9	0.7	0.9	0.6	3.9
	LZW	0.03†	0.1	0.06†	0.3	0.02†	0.1	0.06†	0.3
	ZIP	0.006†	0.1	0.05†	0.2	0.006†	0.1	0.05†	0.2
8	Original	1.4	21.6	18.2	63.3	1.4	21.6	18.2	63.3
	LZW	0.4	13.3	15.5	55.3	0.5	17.0	17.8	62.6
	ZIP	0.4	11.4	12.7	45.1	0.5	15.2	15.2	52.8
24	Original	8.6	24.7	12.8	54.4	8.6	24.7	12.8	54.4
	LZW	2.4	16.4	14.2	60.5	4.5	21.2	14.0	57.1
	ZIP	2.0	13.9	11.1	45.0	3.4	17.5	11.6	43.3
48	Original	25.9	36.4	15.0	57.3	25.9	36.4	15.0	57.3
	LZW	6.8	9.5	2.6	14.9	12.7	16.8	5.0	25.8
	ZIP	5.6	7.6	2.0	11.9	9.9	12.8	3.1	19.2

* "Original" TIFF refers to the original uncompressed image.

[5] File size results are expressed in IEC 80000-13 binary prefixes; 1 KiB (kibibyte) = 1024 Bytes, 1MiB (mibibyte) = 1024 KiB.

Table 6: Minimum, mean and maximum ratio of ZIP to LZW compressed file sizes for each software library (to 1 d.p.)

Library	Min	Mean	S.D.	Max
ImageMagick	22.4%	86.0%	7.5%	95.6%
LibTiff	25.9%	82.6%	9.9%	95.5%
All	22.4%	84.3%	9.0%	95.6%

Summary with respect to Compression Algorithm:

- Either algorithm generates an *average* compressed file size smaller than the original, uncompressed average file size
- ZIP generates compressed files smaller than that produced with LZW.
- The LZW algorithm is capable of increasing the file size, rather than decreasing it.
- The ZIP algorithm has not, for this sample, increased the file size.

4.2.2 Effect of Bit-Depth

The ratios of compressed to original file sizes shown in Table 3 can be examined further based on the bit-depth of the original image. Results from this bit-depth analysis are shown below in Table 7.

These results are clearer at showing for which bit-depth LZW compressed files are not guaranteed to be smaller than their uncompressed originals (specifically, 8 and 24-bit).

They also reinforce, at each bit-depth level, the previously mentioned findings that the average ZIP compressed files are smaller than LZW compressed files. As per Table 6, Table 8 confirms this on a file-by-file basis, with ZIP compressed files being at most ~96% the size of the LZW compressed files.

Table 7: Ratio of LZW/ZIP compressed file sizes as a percentage of the original uncompressed file sizes for each bit-depth (to 1 d.p.)

	Alg.	Bit Depth	Min	Mean	S.D.	Max
ImageMagick	LZW	1	2.3%	17.1%	8.9%	34.2%
		8	24.6%	48.5%	19.0%	87.4%
		24	22.9%	60.3%	25.0%	133.8%
		48	17.5%	27.5%	4.4%	34.0%
	ZIP	1	0.5%	14.1%	7.5%	27.4%
		8	22.6%	42.4%	15.2%	71.2%
		24	19.0%	51.8%	18.2%	99.4%
		48	13.3%	22.2%	3.8%	27.7%
LibTiff	LZW	1	2.0%	16.7%	8.8%	33.8%
		8	32.6%	64.0%	22.6%	102.2%
		24	39.8%	82.2%	18.2%	130.0%
		48	35.0%	48.3%	5.7%	54.6%
	ZIP	1	0.5%	14.5%	7.7%	28.0%
		8	30.4%	57.9%	19.8%	89.2%
		24	22.2%	68.0%	18.0%	98.5%
		48	21.6%	37.9%	7.2%	46.6%

Table 7 shows that the average compression achieved varies with bit-depth, lessening as the bit-depth increases. For example, the average LZW compressed file size produced by ImageMagick is ~17% (of the uncompressed size) for 1-bit, ~48% for 8-bit, and ~60% for 24-bit. Interestingly though, 48-bit images appear to achieve substantially more compression than 8 and 24-bit images, with average compressed file sizes ranging between 22% and 48% of the original uncompressed size. Sample sizes should always be borne in mind, however if considered representative of a larger population value, then this indicates better compression performance on the larger sized image payloads afforded by the 48-bit colour depth.

Table 8: Minimum, mean and maximum ratio of ZIP to LZW compressed file sizes for each software library (to 1 d.p.)

	Bit Depth	Min	Mean	S.D.	Max
ImageMagick	1	22.4%	78.1%	15.5%	90.1%
	8	81.5%	88.4%	3.0%	92.2%
	24	74.1%	87.6%	4.9%	95.6%
	48	75.0%	80.4%	1.6%	82.7%
LibTiff	1	25.9%	82.9%	15.2%	95.1%
	8	84.2%	90.7%	3.1%	95.5%
	24	50.3%	81.9%	9.2%	91.8%
	48	61.6%	77.8%	6.6%	85.8%

Table 7 also clearly illustrates that 1-bit images are capable of being heavily compressed, more so than the other bit-depths. ZIP especially, is able to reduce these bitonal files to 0.5% of their original uncompressed size. Analysis of these 1-bit files shows that the heavily compressed ones have lower average image entropies (described in Section 3.5) than the less compressed files – see Figure 2.

Entropy is a quality indicating how "busy" an image is. Low entropy images – such as a photograph of the night sky, or a page of plain black text against a white background - contain large swathes of pixels with the same or similar colour values. Higher entropy images – such as a photograph of a crowd - have a great deal of contrast between neighbouring pixels.

The theory is that high entropy images have a greater variety of information (e.g., more variation in colour) to encode than lower entropy images, and therefore should be more difficult to compress effectively. These results support this theory for 1-bit images – the ability to Group 4 compress a 1-bit image appears to degrade as image entropy increases.

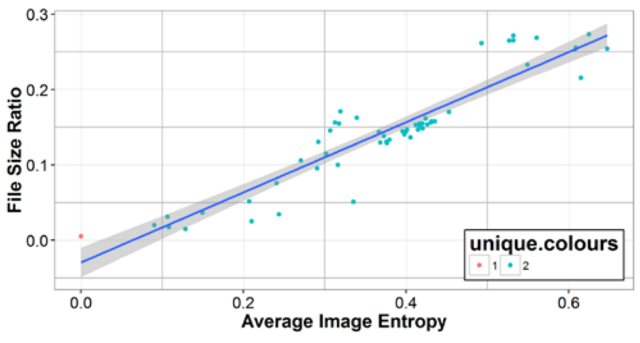

Figure 2: ImageMagick's ZIP to Uncompressed file size ratio for 1-bit TIFFs vs. the average image entropy

Finally, comparing ZIP compressed file sizes relative to LZW compressed sizes (rather than with respect to the uncompressed file size) – as shown in Table 8 – then we actually see that ZIP files are, on average, approximately 80-90% the size of LZW compressed files across the four bit-depth levels. It would appear then, that the ZIP algorithm is achieving similar compression improvements over LZW regardless of bit-depth.

Summary with respect to Bit-Depth:

- The average compressed image file size appears to vary with bit-depth, with compression rates decreasing as bit-depth increases (for 1, 8 and 24-bit images).
- 48-bit images appear to achieve better compression than 8 and 24-bit images, with compressed file sizes between 22% and 48% (of the uncompressed size).
- 1-bit images are capable of being heavily compressed down to 0.5% the uncompressed file size using ZIP. The amount of compression achieved appears correlated to the amount of average image entropy in the file.
- ZIP compressed files are approximately ~84% the size of LZW compressed files, across all bit-depth levels.

4.2.3 Effect of Software Library

As previously mentioned, both LibTiff and ImageMagick generate average compressed file sizes smaller than the original uncompressed file, regardless of compression algorithm applied. However, as can be seen in Table 7, there is variation between the compression performance of the software utilities for similar bit-depths and compression algorithms. Notably, with the exception of 1-bit LZW compressed images (which is in itself a tiny percentage difference anyway), ImageMagick generates smaller average compressed file sizes than LibTiff. Such an effect is more predominant across the 8 to 48-bit colour depths, irrespective of the compression algorithm used. It is also somewhat true when considered on a file-by-file basis.

Table 9 shows the count of files compressed using ImageMagick which have a file size smaller, larger or the same as those compressed using LibTiff. Specifically these results highlight that ImageMagick generally generates smaller files than LibTiff across all bit-depths – approximately 85% of ImageMagick's LZW files are smaller than LibTiff's; and ~97% of its ZIP files are smaller too.

Table 9: Number of ImageMagick files which are smaller, larger or the same size as those compressed with LibTiff

Alg.	Bit Depth	# Smaller	# Larger	Equal Size
LZW	1	0	56	0
	8	57	0	0
	24	327	18	0
	48	45	0	0
	Total	*429 (85.3%)*	*74 (14.7%)*	*0*
ZIP	1	51	5	0
	8	57	0	0
	24	333	12	0
	48	45	0	0
	Total	*486 (96.6%)*	*17 (3.4%)*	*0*

However there are also occasions when LibTiff fares better and ImageMagick results in larger files; for our sample, this is mainly for 1-bit LZW compressed files, although there are also 5 1-bit (ZIP compressed) images and 30 24-bit (LZW and ZIP) images which are larger.

Analysing the 24-bit "larger" images shows they all come from the same sub-collection of content which have a very large number of unique colours compared to the rest of the sample. Figure 3 plots the difference in compressed file sizes between ImageMagick and LibTiff for 24-bit images. Points above the 0MiB difference line indicate files where the ImageMagick version is larger; points below the 0MiB line indicate files where the LibTiff version is larger. For 24-bit images at least, this plot hints at a (non-linear) correlation between the number of unique colours in an image and the compression performance of ImageMagick (with respect to LibTiff).

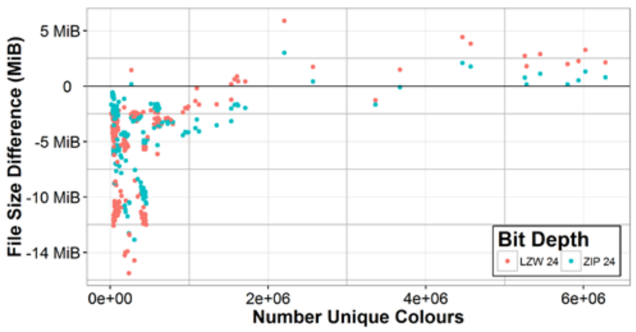

Figure 3: Difference in compressed file size between LibTiff and ImageMagick, with respect to number of unique colours

This all raises an interesting question – if ImageMagick uses the LibTiff library, why should these results differ? Evaluation of the source code shows that for ZIP and LZW compression, ImageMagick uses a slightly higher quality value[6] and sets the TIFF "Predictor" Tag to "horizontal" for RGB and 8/16 bits/sample bilevel/greyscale images[7]. This tag is not set (by default) when using LibTiff directly[8] and requires the user to manually specify the value when compressing an image[9].

The Predictor tag invokes a pre-compression operation which aims to improve compression. The premise for this operation is that subsequent pixels will often be similar in value to their predecessor, especially for continuous-tone images, and so the information content can be reduced by subtracting information already contained in the predecessor pixels. The pixels essentially become the difference from the pixel at the beginning of the row, with many being 0 for continuous-tone images. Compression applied after this can take advantage of lower information content.

ImageMagick's use of this Predictor tag is consistent with our results. With the 24-bit images, the majority of ImageMagick's compressed files are smaller than LibTiff's; where they are larger are for the files which have large numbers of unique colours and high entropy (suggestive of low-continuous-tone). By virtue of how the Predictor differencing works, using this pre-compression operation is unlikely to be as helpful for such files.

These results seemingly contrast with the ZIP to LZW compression ratios shown in Table 8, which show LibTiff offers, on average, slightly better ZIP compression ratio for 24 and 48-bit images than ImageMagick. ZIP compressed 24-bit TIFFs are ~82% (the size of LZW files) for LibTiff versus ~88% for ImageMagick; similarly, 48-bit TIFFs are ~78% versus 80% respectively. Although tempting to think this means LibTiff should offer smaller ZIP files than ImageMagick for these bit-depths, it should be remembered that LibTiff's LZW compression algorithm generates a larger compressed file than ImageMagick's and so the higher compression rates alluded to in Table 8 (with respect to LZW) do not translate to smaller ZIP images. Ultimately ImageMagick generates a smaller average ZIP compressed file than LibTiff, regardless of bit-depth (Table 7).

[6] ImageMagick uses a default quality value of 7, compared to LibTiff's 6; the higher the value, the better the compression.

[7] ImageMagick 6.6.9-5: coders/tiff.c, line 2903 and 2929.

[8] LibTiff 3.9.5: tools/tiffcp.c, line 693

[9] E.g. 'tiffcp –c lzw:2' sets the Predictor tag to 2 (Horizontal)

Finally, whilst the evidence suggests both libraries generate average file sizes less than the original, it also shows that both libraries exhibit cases where LZW compressed files are actually larger than their uncompressed counterparts (see Table 5). Specifically, for our sample LibTiff has over double the occurrences of "larger than original" LZW compressed files than ImageMagick. As previously explained, this is most likely due to limitations with the dictionary based encoding approach used in LZW; however it is also suggestive of implementation differences between the LibTiff and ImageMagick LZW algorithms, such as from the use of the Predictor tag (which favours ImageMagick).

Summary with respect to Software Library

- ImageMagick generates a smaller *average* compressed file size than LibTiff, regardless of compression algorithm.
- Across all bit-depths, our results suggest that:
    - ~85% of ImageMagick's LZW compressed files are smaller than LibTiff's; and,
    - ~97% of ImageMagick's ZIP compressed files are smaller than LibTiff's.
- Some evidence to suggest a correlation between the number of unique colours and whether ImageMagick's compressed files are larger. Further investigation is needed though.
- ImageMagick sets the TIFF "Predictor" tag for RGB and 8/16 bits/sample greyscale images, which could explain its superior compression performance on more continuous-tone images. Further investigation is needed.
- LibTiff appears to offer a slightly better ZIP to LZW compression ratio for 24 and 48-bit images, compared to ImageMagick.
- LibTiff appears more likely to generate LZW compressed files which are larger than the uncompressed file, compared to ImageMagick.

4.3 LZW/ZIP Compressed vs. Group 4 Compressed File Sizes

Whilst the focus of this paper is on the application of ZIP and LZW compression to TIFF files, given we have a subset of files initially Group 4 compressed, it is worth considering how these compare to ZIP and LZW compression. Throughout this discussion it should be kept in mind that Group 4 compression applies to bitonal (1-bit) images only – as such, ImageMagick will not set the Predictor tag.

Table 10 shows the ratio of LZW and ZIP compressed file sizes to the original Group 4 compressed file sizes. As can be seen, compared against TIFFs already compressed using the Group 4 algorithm, LZW and ZIP compressed files are on average, overwhelmingly larger than the originals, with LZW files averaging more than 3 times – and up to 50 times – the Group 4 size, and ZIP files averaging over twice the size – and up to 11 times the size – of the original Group 4 TIFF.

Table 10: Ratio of LZW/ZIP compressed TIFF file sizes as a percentage of the original Group 4 compressed TIFF file sizes (to 1 d.p.)

Library	Alg.	Min	Mean	S.D.	Max
ImageMagick	LZW	69.1%	403.9%	906.6%	5012.2%
	ZIP	55.5%	207.9%	190.8%	1122.5%
LibTiff	LZW	68.2%	366.2%	759.5%	4219.6%
	ZIP	54.6%	212.7%	185.2%	1092.3%

Oddly, these results indicate a difference in resulting file sizes despite the fact that both software libraries do not set the Predictor tag. ImageMagick does use a slightly higher quality setting, which may possibly account for the slightly better ZIP compression (208% vs 213%), however this is not shared in the LZW results (404% vs 366% respectively). It is possible other changes, such as additional tags/metadata, may cause more significant variations in the file sizes seen; further investigation is required.

Table 11: Number of LZW/ZIP compressed files with sizes less than or greater than Group 4 compressed

Library	Alg.	No. LZW/ZIP File size < Group 4	No. LZW/ZIP File size > Group 4
ImageMagick	LZW	11 (19.6%)	45 (80.4%)
	ZIP	12 (21.4%)	44 (78.6%)
LibTiff	LZW	12 (21.4%)	44 (78.6%)
	ZIP	12 (21.4%)	44 (78.6%)

Minimum ratios in Table 10 show that some LZW and ZIP files do compress better than Group 4. Table 11 indicates this is ~20% of such files, consistent across both libraries and algorithms; however a larger sample ideally needs to be tested.

In an effort to understand why certain files compress better and others worse, the LZW/ZIP to Group 4 compressed file size ratio was plotted against the average image entropy – see Figure 4[10]. Recall that entropy is a quality indicating how "busy" an image is, and that higher entropy images should, in theory, be more difficult to compress effectively. Figure 4 suggests that for 1-bit images those with higher entropy are more readily compressible with LZW and ZIP than with Group 4 compression; put another way, Group 4 compression degrades as image entropy increases, which is exactly the result seen earlier in Figure 2.

For our sample there is an outlier file around the 0.25 entropy value (compressing better with ZIP) which goes against our observation, and so further analysis on a larger sample is ideally required before a definitive correlation can be determined.

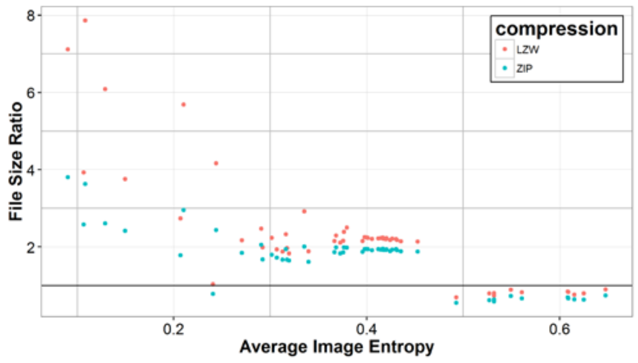

Figure 4: Ratio of LZW/ZIP file size with respect to the Group 4 compressed image file size against the average image entropy (points for 0 entropy not shown)

Summary:

- Group 4 compression applied to 1-bit TIFFs is, on average, at least 2x more effective than ZIP compression, and at least 3x more effective than LZW compression (in terms of generated file size)
- Approximately 20% of 1-bit files compress better with LZW or ZIP algorithms over Group 4.
- There may be a correlation between files with higher entropy values and their ability to be more effectively compressed with LZW or ZIP rather than Group 4 compression.

[10] note: the points for 0 entropy are not shown as they skew the y-axis making the results hard to interpret; these points have a ratio >0

5. WHAT DOES THIS MEAN IN PRACTICE?

This section aims to give practical advice for the application of LZW or ZIP compression based on the previously mentioned findings. The driver for this advice is the reduction of associated file storage, leading to the ability to store more within a defined storage system. Other long-term preservation issues – such as arising from the use of non-baseline TIFF tags or compression algorithms – are not, but should be considered before application of this advice in a production preservation system. Readers should also bear in mind that our observations are obtained from a relatively small sample set, and ideally require further testing.

5.1 For 1-bit TIFFs

Should 1-bit TIFFs be compressed with Group 4, LZW or ZIP to reduce total storage?

Should existing Group 4 compressed TIFFs be decompressed and recompressed with LZW or ZIP to better utilise existing space?

Our analysis indicates that for those TIFFs in the sample which were originally Group 4 compressed (1 bit/pixel, bitonal TIFFs), when decompressed and then recompressed using LZW or ZIP, the original compression was on average more effective than either LZW or ZIP.

This can also be seen in Table 12 below, which compares the mean average and aggregate file sizes for all 1-bit files in our sample in their uncompressed, Group 4 compressed, LZW compressed and ZIP compressed forms. Specifically it indicates that the total space required for the Group 4 compressed sample of 1-bit files is less than for the other compression techniques, regardless of library.

Table 12: Mean average and total size of 1-bit compressed files, by compression algorithm (all to 1 d.p.)

	Alg.	Mean File Size	Total Sample Size
Original	None	940.7 KiB	52677.6 KiB
	Group4	101.4 KiB	5676.5 KiB
ImageMagick	LZW	142.1 KiB	7957.8 KiB
	ZIP	115.1 KiB	6446.1 KiB
LibTiff	LZW	138.9 KiB	7776.2 KiB
	ZIP	118.8 KiB	6653.2 KiB

In essence, the data from our sample suggests that existing 1-bit Group 4 compressed images should be left alone in order to utilize storage space efficiently; uncompressed 1-bit images should be compressed to Group 4.

We did find that approximately 20% of 1-bit files did compress better with LZW or ZIP, hinting at the possibility of selectively encoding 1-bit TIFFs with the most appropriate algorithm to achieve further aggregate space savings. Without conclusive evidence on how to select the "most appropriate" algorithm however, this will probably result in a trial and error approach. It would also diversify the compression profile of TIFFs in a collection.

These results are based on the subset of 1-bit/pixel TIFF images which were originally Group 4 compressed. It is likely that these findings may be extended to other bitonal images, however it should be noted that we have not performed Group 4 compression, and so it is unclear if, and how, these results will be affected through performing such compression with either ImageMagick or LibTiff; further testing would be required in order to formally confirm these conclusion.

5.2 For 8, 24 and 48-bit TIFF images

Should 8, 24 and 48-bit TIFFs be compressed with LZW or ZIP to most effectively reduce total storage?

Which software utility should be used to reduce total storage?

Examination of the file compression ratios has shown that, for our sample at least, ZIP compression is uniformly superior to LZW compression in terms of the degree of file size reduction, irrespective of bit-depth or software library.

Table 13 documents the total space requirements for each bit-depth of our tested sample compressed with each algorithm by each library. This illustrates, particular for 24-bit images, the storage savings achievable through use of ZIP compression over LZW. It also illustrates a preference for using ImageMagick over LibTiff.

While there are some cases where the difference in effectiveness between LZW and ZIP compression is small, there are no examples in this analysis where the ZIP compressed file was larger than the corresponding LZW compressed file. Furthermore, ZIP compression has not caused an overall increase in file size for any images in this sample, which is the case for LZW compression, particularly on 24-bit images.

From a practical perspective, the data from our sample suggests that 8, 24 and 48-bit TIFF images should ZIP compressed using ImageMagick, in order to reduce overall storage space.

Table 13: Mean average and total size of 8-, 24- and 48- bit compressed files, by compression algorithm (all to 1 d.p.)

	Alg.	Bit Depth	Mean File Size	Total Sample Size
Original	None	8	21.6 MiB	1.2 GiB
		24	24.7 MiB	8.3 GiB
		48	36.4 MiB	1.6 GiB
ImageMagick	LZW	8	13.3 MiB	0.7 GiB
		24	16.4 MiB	5.5 GiB
		48	9.5 MiB	0.4 GiB
	ZIP	8	11.4 MiB	0.6 GiB
		24	13.9 MiB	4.7 GiB
		48	7.6 MiB	0.3 GiB
LibTiff	LZW	8	17.0 MiB	0.9 GiB
		24	21.2 MiB	7.2 GiB
		48	16.8 MiB	0.7 GiB
	ZIP	8	15.2 MiB	0.8 GiB
		24	17.5 MiB	5.9 GiB
		48	12.8 MiB	0.6 GiB

Through plotting, we did find that the number of unique colours in 24-bit images appears to suggest a correlation with whether ImageMagick compressed files were larger (than LibTiff's). Although further investigation is needed, this may present a mechanism for selectively encoding 24-bit images using the appropriate library, in order to achieve optimal storage reductions.

5.3 Optimal vs. Recommended Compression

How much total disk space could be saved by using the most efficient compression per file compared to the recommended?

Sections 5.1 and 5.2 presented recommendations, based on evidence from our tested sample, for which compression to apply and what software to use, for each bit-depth of image. Namely:

- 1-bit images: Group 4 compress
- All others: ZIP compress with ImageMagick

It was acknowledged however, that for some files the recommendations were suboptimal (within the bounds of our analysis) with respect to compressed file size. Specifically, some 1-bit images compressed better with ZIP/LZW than Group 4, and some 24-bit images compressed better with LibTiff rather than ImageMagick. If it were feasible to be selective over the compression approach – library and algorithm combination – how much storage would be saved?

Table 14 shows the total storage requirements for the original sample set (i.e. Group 4 compressed 1-bit images; all others uncompressed), plus the storage needs if the recommended or optimal compression approaches were used. It also includes storage figures for an alternative compression approach using Group 4 for 1-bit images and LibTiff ZIP compression (with default settings) for all others.

Table 14: Total sample sizes (in MiB) achieved from original, optimal, recommended and alternative approaches (to 1 d.p.)

	Sample Sizes			
Bit Depth	Original	Optimal	Recommended	Alternative
1	5.5	4.3	5.5	5.5
8	1231.1	651.7	651.7	865.7
24	8512.3	4789.8	4801.5	6039.1
48	1636.4	341.1	341.1	578.0
Total	***11385.3***	***5787.0***	***5799.8***	***7488.3***

In total, there is approximately 13MiB saved from using the optimal approach as opposed to the recommended. Considering the average compressed file sizes presented in Table 13, this equates roughly to an ability to store 1 extra compressed image (out of the ~500 sample).

More generally, the recommended approach has led to an approximate 50% reduction in total file size over the original sample.

As way of example of how the software library employed can have an effect, the alternative compression approach – which use LibTiff ZIP compression instead of ImageMagick's – requires nearly 30% more storage than the recommended approach.

6. CONCLUSIONS

This paper focused on comparing the relative effectiveness of two lossless compression algorithms - LZW and ZIP - on a collection of TIFF files, with the aim of reducing the overall storage needs for the collection. Two software utilities were tested (using default settings) – ImageMagick and LibTiff – to investigate the impact the software choice has on achievable file sizes.

Group 4 compression was found, on average, to be superior to either LZW or ZIP compression when applied to 1-bit bitonal images by at least a factor of 2. Despite this, approximately 20% of our sample of 1-bit images did compress better (on an individual level) with LZW and ZIP. Investigation found some evidence to suggest that the effectiveness of Group 4 correlates (inversely) with the amount of entropy in an image – i.e. "busier" images appear to compress less. However, with only 56 1-bit images in the sample, testing of a larger set would be needed to confirm this.

The ZIP algorithm was found to be superior in effectiveness to LZW for all images in the sample, always generating compressed files smaller than the uncompressed and the LZW TIFFs. In contrast, LZW compression, when applied to the 8-bit and 24-bit images in the sample, occasionally resulted in an increase in file size (from the uncompressed form). This occurred more often when using LibTiff.

The effectiveness of both ZIP and LZW compression algorithms varied with image colour depth, with compression rates decreasing as bit-depth increased (up to 24-bit). 48-bit images seem to buck this trend, achieving better compression rates than 8 and 24-bit images. For specific compression rates see Table 7.

ImageMagick was found to generate smaller average compressed files than LibTiff, with ~85% of its LZW and ~97% of its ZIP compressed files being smaller. Analysis showed that ImageMagick uses the same LibTiff libraries, prompting questions as to why the results should vary so much. Deeper investigation indicated that ImageMagick sets the TIFF 'Predictor' extension tag which enhances LZW/ZIP compression for certain images, offering a probable explanation for the difference, but one that requires further analysis. Theoretically, similar levels of compression should be achievable using LibTiff by setting this tag (no analysis has been performed to confirm this); however based on these results, ImageMagick will perform better by default.

Taking these observations into account, in order to reduce storage space effectively, the following recommendations are suggested:

- For 1-bit images, compress with Group 4
- For all others, ZIP compress with ImageMagick.

For our tested sample, these recommendations result in an approximate storage saving of 50% across the entire collection. It may be possible to reduce the overall storage for a collection further by selecting the most appropriate compression approach on a file-by-file basis; however there is no clear guidance on how to select the best compression approach for any given file, and the overall storage reduction across a collection appears minimal.

6.1 Caveats and Future Work

The figures in this paper should be interpreted with the size of the sample in mind. How these results compare to those obtained from much larger samples remains to be seen, and would be useful further work. In particular, it would be useful to test on a sample set that encompasses larger sub-collections, i.e. a sample with larger numbers of 1-bit, 8-bit and 48-bit images.

Given the connection and results variation between software utilities shown, evaluating the performance of these libraries using the same settings would be of benefit, for example, comparing LibTiff with the Predictor tag set to 'Horizontal' and a quality level of 7 (as per ImageMagick).

It should also be borne in mind that no Group 4 compression was undertaken. 1-bit files were already Group 4 compressed in the sample, and these were used as is. An obvious enhancement to these experiments would be to start with uncompressed TIFFs and Group 4 compress them using LibTiff and ImageMagick.

Compression of a TIFF file is applied to the image payload only. Whilst it might be expected that this would be the only source of change in a file when compressing, additional changes also occur in the tagged metadata portions of the file to describe the compression. Furthermore, additional metadata, particularly that associated with the Adobe Photoshop's "Image Source Data" tag (# 37724), which captures layering information, appears to be removed during compression. Such changes to the tagged metadata are included in the file sizes presented in this paper. Therefore the change in file size represents the complete change to the file, and not just the change to the image pixel data. Consideration should be given as to whether this presents vital information that must be preserved, and therefore whether compression is appropriate.

Similarly, this paper does not address other long-term preservation issues with the use of TIFFs, non-baseline tags and compression. Robustness of compressed TIFF formats towards bit-errors is not examined, although perhaps mitigated through bit-level preservation. LZW and ZIP compression are both TIFF extensions which do not have to be supported by TIFF readers, as is the Predictor tag. Subsequently, there is a small possibility that compressing TIFFs may make them difficult to render with baseline-compliant-only TIFF readers. Whilst there is currently software (e.g. LibTiff) able to decompress such files, consideration needs to be given to the appropriate preservation practices and documentation required for the software and algorithms involved.

7. REFERENCES

[1] Adobe Photoshop® TIFF Technical Notes. 22 March 2002. http://partners.adobe.com/public/developer/en/tiff/TIFFphotoshop.pdf [Online; cited: 24 Apr 2016]

[2] Gillesse, R., Rog, J., Verheusen, A. 2008. *Life Beyond Uncompressed TIFF: Alternative File Formats for the Storage of Master Images Files*. In: Proceedings of the IS&T Archiving Conference, Bern, Switzerland

[3] LZW Patent Information. Unisys. 02 June 2009. https://web.archive.org/web/20090602212118/http://www.unisys.com/about__unisys/lzw [Online; cited: 24 Apr 2016]

[4] Mateika, D. Martavicius, R. 2006. *Analysis of the Compression Ratio and Quality in Medical Images*. ISSN 1392-124X. Information Technology and Control, vol. 35, No. 4

[5] Puglia, S., Reed, J., Rhodes, E. 2004. T*echnical Guidelines for Digitizing Archival Materials for Electronic Access: Creation of Production Master Files – Raster Images*. U.S. National Archives and Records Administration (NARA). http://www.archives.gov/preservation/technical/guidelines.pdf [Online; cited: 12 Apr 2016].

[6] Succeed Project, 2014, *D4.1 Recommendations for metadata and data formats for online availability and long-term preservation*. http://www.succeed-project.eu/sites/default/files/deliverables/Succeed_600555_WP4_D4.1_RecommendationsOnFormatsAndStandards_v1.1.pdf [Online; cited: 12 Apr 2016].

[7] *Technical Guidelines for Digitizing Cultural Heritage Materials: Creation of Raster Image Master Files*. 2010. Federal Agencies Digitization Guidelines Initiative (FADGI) – Still Image Working Group. http://www.digitizationguidelines.gov/guidelines/FADGI_Still_Image-Tech_Guidelines_2010-08-24.pdf [Online; cited: 12 Apr 2016].

[8] The National Digital Newspaper Program (NDNP) Technical Guidelines for Applicants. 2015. Library of Congress. https://www.loc.gov/ndnp/guidelines/NDNP_201618TechNotes.pdf [Online; cited: 12 Apr 2016]

[9] TIFF Revision 6.0. [Online] 3 June 1992. https://partners.adobe.com/public/developer/en/tiff/TIFF6.pdf. [Online; cited: 24 Apr 2016]

[10] Wheatley, P., May, P., Pennock, M., *et al.* 2015. *TIFF Format Preservation Assessment*. http://wiki.dpconline.org/images/6/64/TIFF_Assessment_v1.3.pdf [Online; cited: 24 Apr 2016]