Proceedings of the 12th International Conference “Reliability and Statistics in Transportation and Communication” (RelStat’12), 17–20 October 2012, Riga, Latvia, p. 307-314. ISBN 978-9984-818-49-8
Transport and Telecommunication Institute, Lomonosova 1, LV-1019, Riga, Latvia

LOSSLESS COMPRESSION OF HEXAGONALLY SAMPLED IMAGES USING A MULTIRESOLUTION FILTER BANK

Aleksejs Truhans

Transport and Telecommunication Institute
Lomonosova str. 1, Riga, LV-1019, Latvia
Ph.: +371 29450858. E-mail: aleksejs.truhans@gmail.com

The paper describes development and implementation of a hexagonally sampled image lossless compression method based on a multiresolution filter bank. The work is inspired by successful exploitation of hexagonal lattice advantages in other fields and successful application of wavelet transform to image compression. In addition the work also considers transformation between orthogonal and hexagonal lattices and possibility of applying the method to compress orthogonal images. Efficiency of the method described in this paper is compared with that of PNG and JPEG 2000 Lossless algorithms. The developed method achieves compression ratio comparable with that of the benchmark methods when applied to photographic images. It allows an efficient implementation and is suitable for practical application. There is also a reference software implementation of the method available at https://github.com/sweer/hex-image-compress for use and further research.

Keywords: hexagonal lattice, lossless compression, perfect reconstruction filter bank, multiresolution two-dimensional filtering

1. Introduction

Hexagonal lattice has some advantages over the orthogonal lattice, including higher symmetry degree, equidistant neighbours, compactness and overall better representation of round shapes. Due to existence of especially efficient algorithms, hexagonal image processing is used in bionics, medicine, and geosciences. Compression of such images will reduce transmission and storage costs in practical applications; it will also facilitate further development of the hexagonal image processing field.

Of all existing orthogonal image compression methods two have been selected as benchmarks: Portable Network Graphics (PNG) and JPEG 2000 Lossless [1]. Both standards show similar performance on photographic images, yet differ in other aspects. PNG is relatively simple and widely used, providing a high level of performance on artificial imagery. JPEG 2000 Lossless uses a wavelet transform and a very complex entropy coding process. JPEG 2000 use is very limited, partially due to licensing issues.

A succinct yet comprehensive overview of hexagonal image processing history and practical implementation issues is given in [2]. Several authors, including [3] and [4], have demonstrated the possibility and some ways of creating perfect reconstruction two-dimensional filter banks on a hexagonal grid. The filter bank presented in this paper has been designed independently, though it may appear to belong to one of the filter bank families described in the before mentioned works.

The key considerations concerning hexagonal image compression are provided below:

- The advantages of hexagonal lattice could allow for a compression method surpassing those used nowadays for orthogonal images.
- Image compression is often successfully performed in two steps: first pixel value decorrelation and then entropy coding.
- The decorrelation compression step is sensitive to image content. In particular, a multiresolution transform may perform worse on artificial images.
- Gradual improvement of image quality in the course of receiving image data would be beneficial for many applications.
- There are still many other issues hindering use of hexagonally sampled images, including lack of hexagonal image sensors, displays and processing methods. Moreover, reasoning using the hexagonal grid appears much more complicated to modern humans, than using the orthogonal one.

2. Image Compression Method

Purpose of the work, is to develop a lossless image compression method using multiresolution filtering of hexagonally sampled images. The work first considers transformation between orthogonal and hexagonal grids to (a) generate hexagonally sampled images for experiments, and (b) explore applicability of the compression method to orthogonal images. Secondly, the method of reducing entropy of pixel values is discussed; it is based on iterative application of low-pass and high-pass filters to image pixels. Every iteration takes lower resolution output from the previous iteration and divides it into low and high frequencies very much like a wavelet transform. Pixel values after the transform are compressed using a range coder.

The work deals only with grayscale images for simplicity reasons. Transition from lossless grayscale compression to lossless colour one is a quantitative, rather than qualitative problem, hence colour is omitted in this research.

Images from University of Waterloo Image Repository [5] are used for testing and analysis throughout this work. The images are shown in Table 1 and are referenced further in the work by their respective symbolic names. All the test images are 256 x 256 pixels in size, 8 bits per pixel grayscale luminance levels. Thus, pixel values in the images are in 0 255 range.

Table 1. Test images

Figure 1. Bird

Figure 2. Bridge

Figure 3. Camera

Figure 4. Circles

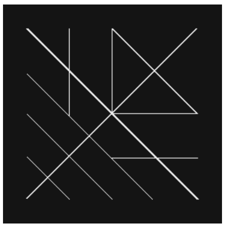

Figure 5. Crosses

Figure 6. Goldhill1

Figure 7. Horiz

Figure 8. Lena1

Figure 9. Montage

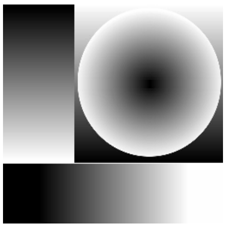

Figure 10. Slope

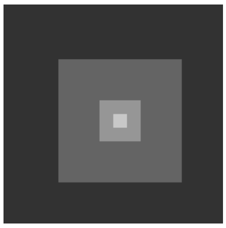

Figure 11. Squares

Figure 12. Text

2.1. Orthogonal-hexagonal conversion

Overwhelming majority of digital images nowadays is orthogonal. Development of a hexagonal image compression method thus requires conversion between orthogonal and hexagonal lattices. In this work the conversion is used to (a) generate hexagonal images to experiment with, and (b) research the possibility of applying the said compression method to orthogonal images. The conversion employed for this purpose is "Reversible Hexagonal-Orthogonal Grid Conversion by 1-D Filtering" (further in the text H2O) [6]. H2O conversion has the following advantages:

- It can be used for conversion in both directions: from orthogonal to hexagonal grid and back, using essentially the same operations;
- It allows for perfect reconstruction of an image converted to another grid, given H2O itself is based on a perfect reconstruction one-dimensional filtering;
- It keeps pixel density intact – after conversion there are as many pixels as there were before.

H2O conversion is performed as three subsequent shearing, each in a "natural" lattice direction. E.g. conversion from orthogonal to hexagonal grid consists of horizontal, then diagonal and then vertical shearing. Each of the shearing consists of one-dimensional interpolation, performed independently for each row, diagonal or column respectively. Such a shearing requires less memory than a two-dimensional resampling operation and can be easily parallelized over multiple computing units.

The one-dimensional interpolation used by H2O algorithm in this work is described by (1).

$$x(t) = \sum_{i=-n}^{2n-1} x[i \bmod n] \cdot \mathrm{sinc}(t-i)\,. \quad (1)$$

It is based on Whittaker-Shannon sinc interpolation. The signal being interpolated is repeated once before and once after itself. The repetition yields better reconstruction of edge pixels and long sequences of the same colour. Increasing number of repetitions makes the reconstruction somewhat better yet takes too much computing time to be feasible. Mirroring the signal instead of repeating it results in much higher pixel errors as shown in Table 2 for some test photographic image.

Table 2. Reconstruction quality example

	Min	1st Quartile	Mean	3rd Quartile	Max
Mirroring	-115.5	-1.295	0.164	1.402	121.5
Repeating	-3.288	-0.3553	0.00009	0.3477	4.363

Table 3 shows the observed reconstruction quality of the repeating sinc interpolator-based H2O using the test images (see Table 1). It features Peak signal-to-noise ratio (PSNR); percentage of pixels having reconstruction error small enough to be cancelled by rounding; some characteristics of error distribution.

Table 3. H2O implementation reconstruction quality

	PSNR, dB	pixels having absolute error < 0.5	90th percentile of absolute error	min error	1st quartile	mean error	3rd quartile	max error
Text	38.207	22.6%	5.295	-20.114	-1.580	-0.001	1.568	16.770
Slope	46.983	45.1%	1.599	-52.812	-0.583	0.000	0.553	79.291
Crosses	48.467	52.8%	1.642	-6.390	-0.463	0.000	0.457	6.521
Montage	49.650	65.3%	1.061	-48.872	-0.336	0.000	0.339	73.066
Circles	49.795	59.5%	1.370	-11.200	-0.362	0.000	0.364	10.552
Goldhill1	50.203	60.0%	1.243	-8.281	-0.385	0.000	0.378	9.125
Bridge	50.544	59.4%	1.215	-9.574	-0.376	0.000	0.391	11.066
Horiz	51.103	65.7%	1.359	-3.261	-0.161	0.000	0.162	2.668
Camera	52.845	71.2%	0.943	-5.207	-0.263	0.000	0.268	5.825
Lena1	55.608	80.1%	0.690	-6.368	-0.227	0.000	0.224	5.795
Bird	58.807	91.9%	0.466	-6.397	-0.171	0.000	0.165	7.858
Squares	73.267	100.0%	0.090	-1.044	-0.028	0.000	0.028	0.935

2.2. Orthogonal image compression using hexagonal grid challenges

Having found a superior hexagonal image compression method it will be tempting to use it for orthogonal image compression. It can be achieved converting the orthogonal image to the hexagonal grid before compression. The decompression will involve conversion back to the orthogonal grid respectively. However, there are several practical issues to be considered. The orthogonal-hexagonal conversions introduce noise to the signal. The noise can be computed as the difference between the original and reconstructed orthogonal images, compressed and stored alongside the compressed image. Experiments with H2O and range coder-based entropy encoding have shown the noise to take too much space. Most of it is contributed by rounding: orthogonal to hexagonal lattice conversion methods produce real (floating point) values for hexagonal pixels. It maximizes the hexagonal image sampling precision, yet after decorrelating the pixel values one has to round them up to feed to fit into a rather small alphabet of an entropy coder. The subsequent decompression restores then rounded values and reverse decorrelation multiplies the rounding error “echoing” it over surrounding point.

There are two possible solutions to the problem. First is to have a perfectly reversible integer-to-integer orthogonal to hexagonal grid conversion. Such a choice will sacrifice some of the hexagonal grid advantages. The image on the hexagonal grid will then be a very rough approximation of what a proper resampling to a hexagonal grid would look like. Nevertheless, this approximation might still appear more compressible than orthogonal representation of the image.

The second option would be to abandon the lossless orthogonal image compression on a hexagonal grid idea as infeasible and concentrate on lossy compression instead. The error introduced by converting an orthogonal image to hexagonal and then back to orthogonal might not be noticeable for the human eye.

With all these considerations the work concentrates solely on hexagonal image compression. Practical applications of the compression to orthogonal images require separate research and are as such out of the scope of this work.

2.3. The decorrelation filter bank

The idea of the filter bank used in this work belongs to Anatoly Ressin. The bank consists of four two-dimensional filters: one low-pass and three high-passes, applied sequentially at each resolution level. When processing at the very first level, the low-pass filter is applied, storing low frequency value in every second pixel in every second row (see ***Figure 13***).

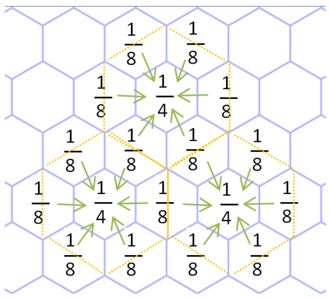

Figure 13. Low-pass filtering step

The value is computed as a weighted average of the pixel itself and its surrounding six neighbours. Let’s call them “centre” and “petals” for easier referencing. The ratio of the centre and petal weights is 2:1. Sum of the weight coefficients shall be equal to 1 in order to get an average in the same scale as the source values, hence the weight coefficients 1/8 for each of six petals and 1/4 for the centre.

After low-pass filtering the high-pass filtering is performed for all the petals. Each petal has two adjacent centres. Average of the centres is used as a prediction of the petal value. The average is subtracted from the petal value thus leaving the high frequency component in the petal. The neighbourhood used in the high frequency filtering of the point marked with X is shown in ***Figure 14***. It follows from the fact that the two centre values (dark green) have been computed using surrounding pixels. Depending on petal position the filter will be rotated by 60° or 120°. The filter bank therefore has 3-fold rotational and axial symmetry.

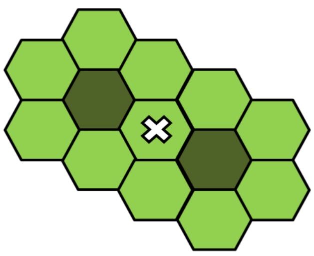

Figure 14. High-pass filter form

Thus, each pixel is updated with new value, either average or difference. The $1/4^{th}$ of the original pixels holding the low-pass component (the averages) is then processed at the next, second, resolution level, using the same algorithm (see ***Figure 15***). Again every second pixel in the every second row of the level becomes a centre.

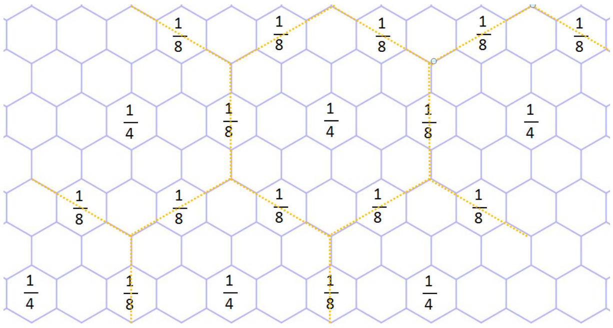

Figure 15. Second level filtering

After filtering each level the pixels containing low-pass filter output contain a blurred version of the original image featuring properties of the image appropriate for the current resolution level. After processing all levels there remains one "average" pixel, while all other pixels contain the high frequency components. Denoting i^{th} pixel value at level k as $x_{i,k}$, the filtering can be expressed in a lifting scheme-like way: the first centre values are computed as shown in (2),

$$x_{i,k+1} = \frac{x_{i,k}}{4} + \frac{1}{8} \sum_{j \in neighbours\ of\ x_{i,k}} x_{j,k}, \tag{2}$$

then values of petals as shown in (3).

$$x_{i,k+1} = x_{i,k} - \frac{1}{2} \sum_{j \in adjacent\ centers\ of\ x_{i,k}} x_{j,k+1}. \tag{3}$$

At the edge of the picture some surrounding hexagons may be missing, especially if it is not a picture from a hexagonal sensor containing precisely 7^N hexagons. During the low-pass filtering values of the missing petals are assumed equal to the average of petals which are present. In the high-pass filtering if one of the adjacent centres is missing the one that is present is used as petal's predictor.

The reconstruction process is an exact reverse of the filtering. It starts with the highest level. At each level it first restores petal values from values of the centres and high frequency remainders in the petals. Then the value of each centre is restored using values of surrounding petals and the weighted average stored in the centre. The process is then repeated on the previous levels until the very first level.

Thus the proposed method is indeed a perfect reconstruction multiresolution filter bank. Possible implementation of the filter bank and experimental results it produces are described below.

2.4. Decorrelation in integers

For optimal use of storage and processing resources it is best to implement the entire compression pipeline using integer operations only. In this section the integer version of the filter bank is described. The integer low-pass filter is described by (4)–(6).

$$s = x_{i,k} + \left\lfloor \frac{1}{2} \sum_{j \in neighbours\ of\ x_{i,k}} x_{j,k} \right\rfloor, \tag{4}$$

$$x_{i,k+1} = \left\lfloor \frac{s}{4} \right\rfloor, \tag{5}$$

$$b_{i,k+1} = s \bmod 4. \tag{6}$$

The computed value s is potentially 10 bits long. The most significant 8 bits comprising the rounded weighted average are stored in the centre, while the least significant 2 bits are stored in a separate bit plane b to be used at the reconstruction step.

Petal value computation is described in (7).

$$x_{i,k+1} = \left(x_{i,k} - \left\lfloor \frac{1}{2} \sum_{j \in adjacent\ centers\ of\ x_{i,k}} x_{j,k+1} \right\rfloor \right) \bmod 256. \tag{7}$$

Reconstruction formulae are shown in (8) and (9) respectively.

$$x_{i,k} = \left(x_{i,k+1} + \left\lfloor \frac{1}{2} \sum_{j \in adjacent\ centers\ of\ x_{i,k}} x_{j,k+1} \right\rfloor \right) \bmod 256. \tag{8}$$

$$x_{i,k} = 4x_{i,k+1} + b_{i,k+1} - \left\lfloor \frac{1}{2} \sum_{j \in neighbours\ of\ x_{i,k}} x_{j,k} \right\rfloor. \tag{9}$$

The filtering described requires very little computing resources. Encoding each pixel takes 1.5 bitwise shift operations, 2 integer sum and 1 bitwise AND operation on average. Total number of pixel encoding necessary to filter all levels is 4*N/3, where N is number of pixels. Reconstruction has similar resource requirements. Moreover, filtering of each level can be performed in place and parallelized over several computing units, which makes it very attractive for implementation on modern hardware.

2.5. Compression performance

In order to estimate compression performance achieved by the suggested compression method a software reference implementation has been created using Java programming language and Test Driven Development approach. The software source code is available at https://github.com/sweer/hex-image-compress licensed under GNU Public License. The software can be used for subsequent development and experiments in the field of hexagonal image processing. The compression pipeline implemented consists of three stages:

- Orthogonal-hexagonal transform;
- Integer-to-integer decorrelation using the filter bank as described in chapter 0. The decorrelation is applied only when it gives a positive effect, that is, reduces the compressed image total size;
- Entropy coding with a range coder.

Compression test results are shown in Table 4. Overall, each image has been successfully compressed.

The orthogonal-hexagonal transform for photographic images is performed using “repeating sinc”-based H2O described in chapter 2.1, with pixel values rescaled into the 0–255 range and rounded afterwards. This produces an approximation of a natural image taken with a hexagonal sensor. For artificial images, however, the nearest neighbour sampling method (denoted by NN in the table) is used in the experiments to create test hexagonal images. This choice avoids the wave-like artefacts band limited sinc-interpolation introduces around steep edges and long constant sequences encountered in artificial images.

Contribution of the decorrelation filter bank is measured for each image comparing the compression degree achieved with and without the decorrelation step. Thus cases where filtering increases the entropy have “filter contribution” column value less than 1. In such cases the source image is fed into the entropy coding skipping the decorrelation to achieve better compression.

Table 4. Compression performance

	H2O or NN	filter contribution ratio	Compression ratio
Bird	H2O	1.51	1.59
Camera	H2O	1.25	1.44
Lena1	H2O	1.30	1.36
Goldhill1	H2O	1.20	1.27
Bridge	H2O	1.16	1.22
Crosses	NN	0.15	30.05
Horiz	NN	0.42	15.24
Text	NN	0.13	13.97
Squares	NN	1.30	9.00
Circles	NN	1.35	5.80
Slope	NN	2.38	2.52
Montage	NN	1.40	1.68

Comparison with the compression performance of JPEG 2000 Lossless and PNG is provided in Table 5. The achieved compression ratio for photographic images is in the range of 72%–92% of the benchmark lossless orthogonal image compression methods. With artificial images it is not so definite. There are also significant differences in artificial image compression ratio between PNG and JPEG 2000 Lossless. This may be a manifestation of some fundamental multiresolution transform property.

PNG compression was performed using GIMP version 2.8 with maximal compression setting. JPEG 2000 Lossless compression was performed using OpenJPEG software.

Table 5. Comparison with existing methods

	Developed method compression ratio	JPEG 2000 lossless compression ratio	PNG compression ratio	Performance relative to JPEG	Performance relative to PNG
Text	13.97	1.90	35.56	736%	39%
Crosses	30.05	7.53	35.56	399%	85%
Bridge	1.22	1.33	1.35	92%	90%
Goldhill1	1.27	1.45	1.46	88%	87%
Camera	1.44	1.75	1.71	82%	84%
Lena1	1.36	1.68	1.59	81%	86%
Bird	1.59	2.21	2.02	72%	79%
Montage	1.68	2.69	2.74	62%	61%
Slope	2.52	5.98	5.71	42%	44%
Circles	5.80	8.65	45.71	67%	13%
Horiz	15.24	33.68	128.00	45%	12%
Squares	9.00	49.23	128.00	18%	7%

Conclusions

In this work a novel, fast and efficient lossless compression of hexagonally sampled photographic images method is presented. Possible applications of the method include image transmission in a machine vision system, storing images in a medical image archive, etc. The developed method is efficient enough both in terms of consumed computational resources and in terms of achieved compression ratio. For the photographic test images set used in this work the method's compression ratio reaches 72%–92% of that of the benchmark lossless orthogonal image compression methods: PNG and JPEG 2000 Lossless. Performing grayscale image encoding takes just 6 integer operations per pixel, can be done in place and parallelized over multiple processing units.

There are two possible directions for the further research. The first one leads towards a similar lossy compression method using some kind of orthogonal-hexagonal lattice transform and thus applicable to orthogonal images. The second one is inspired by PNG relative simplicity and superior performance, as compared to JPEG 2000 lossless. It could be possible to develop a hexagonal grid-based PNG-like image compression algorithm.

Acknowledgements

The author is very grateful to Anatoly Ressin for the idea of the filter bank usage in the study. The author thanks the anonymous reviewers for making many valuable suggestions that significantly improve the presentation of this paper.

References

1. Christopoulos, C. A., Skodras, A. N., Ebrahimi, T. (Nov. 2000). The JPEG2000 Still Image Coding System: An Overview. *IEEE Trans. on Consumer Electronics*, 46(4), 1103–1127.
2. Middleton, L., Sivaswamy, J. (2001). Edge Detection in a Hexagonal-Image Processing Framework. *Image and Vision Computing*. 19, 1071–1081.
3. Cohen, A., Schlenker, J.-M. (1993). Compactly Supported Bidimensional Wavelet Bases with Hexagonal Symmetry. *Constructive Approximation,* 9, 209–236.
4. Jiang, Q. (Sep. 2008). FIR Filter Banks for Hexagonal Data Processing. *IEEE Trans. Image Proc.*, 17(9), 1512–1521.
5. *University of Waterloo Image Repository*. Retrieved December 15, 2009 from http://links.uwaterloo.ca/Repository.html
6. Condat, L., Forster-Heinlein, B., Van De Ville, D. (2007). Reversible Hexagonal-Orthogonal Grid Conversion by 1-D Filtering. In: *IEEE ICIP 2007. Vol. II* (pp. 73–76). USA: IEEE.