EG UK Theory and Practice of Computer Graphics (2007)
Ik Soo Lim, David Duce (Editors)

Advantages of allowing Hexagonal Pixels to be used as a Boundary Description Format

M. J. Turner[†]

Manchester Visualization Centre, Manchester Computing, The University of Manchester, Manchester M13 9PL UK

Abstract

This paper considers some algorithmic extensions and some specific advantages when using a hexagonal pixel array as compared with the usual practice of using a rectangular or square pixel raster array. Discussed are some of the algorithmic changes that are required when using pixels arranged within a hexagonal matrix. The use of a boundary hexagonal description format is considered from the point of view of; hexagonal six-connected contour edge encoding as compared with square four-connected and eight-connected descriptors, efficient probability autocorrelation analysis, direct image manipulation and algorithmic simplification.

Hardware and software conversion techniques for hexagonal pixels are now being seriously considered especially with the potential emergence of hexagonal CCD censor arrangements for cameras; for example this includes a recent patent with Fuji Photo Film Co.Ltd. (US patent number 6882364 Apr 19 2005) incorporating a bidirectional honeycomb pattern.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Bitmap and framebuffer operations E.4 [Coding and Information Theory]: Data compaction and compression

1. Introduction

There has been extensive previous work that has considered region growing algorithms as an efficient image descriptor often considering only square pixel raster arrays [TN92, TW96]. Extensions to these algorithms have considered rectangular pixels or different modes of progressive scan pixel formats. This paper considers the use and some algorithmic improvements that are achievable when employing hexagonal pixel raster arrays for region encoding.

There has also been previous works considering the advantages from a hardware and software point of view of using hexagonal pixels [WS91]. For example various techniques of hexagonal grid representations and their specialist use have been considered for edge detection [MS01], half-toning [Uli87] and triangular grids for improved video colour blending. Commercially there is now renewed interest in new 'honeycomb' arrangements for CCD cameras and other scanner devices as well as some developed prototype display systems [KTA*04]. As higher resolution devices become more endemic the packing advantages of a honeycomb or hexagonal arrangement, over the distortion disadvantages, caused due to artifacts when rendering straight lines, may become more popular that the conventional square raster arrangement [YK93, Tyt00]. For hexagonal raster formats other researchers have proposed advantages in packing density, colour rendition and sharpness characteristics [KTA*04], but these will not be considered directly within this paper.

We are going to present some of the advantages from the point of view as a boundary description format, and we will use a previously described *contour tree* format [TE01, Tur00b] as a basis. This paper will demonstrate both the probabilistic neighbourhood statistics that may be exploitable, as well as some of the advantages in the reduction in complexity of certain algorithms when using hexagonal pixels. Although repeatedly proposed over the last few decades hexagonal pixel representations have not become popular for a few justifiable commercial reasons although we will show that all the principles of region growing al-

[†] Martin.Turner@manchester.ac.uk
http://www.mc.manchester.ac.uk/

gorithms can be applied to hexagons as well as squares and may be more appropriate.

2. Regions, contours and other definitions

An image can be described as a set of areas each having a feature, for example the same intensity value, a functional description or a texture type. From the description of a boundary extraction algorithm (including Freeman's original paper described in the early 60's [Fre61]) we can and need to differentiate between a "region" – a connected area whose contents have this constant feature; and a "contour" – an outside boundary of a region that has this constant feature. For practical purposes we will always consider a contour boundary as being at least one pixel thick.

Given a specific connectivity we can describe any image as a set of connected regions, and encode the complete image from three specific streams of information;

1. a stream of start locations for the contours, defined as the top-left pixel in a raster scan sense for each contour,
2. a stream of descriptors of the constraint feature describing the contour, in this paper that is the intensity or colour value of the contour, and
3. a stream of boundary descriptors for the contours.

The details of four- and eight-connected contour descriptions are to be found in a previous paper [TW96] and we will in the next sections consider some implications of using a six-connected hexagonal contour description.

2.1. Six-connected Boundary Descriptions

When considering figure 1 there are two main obvious methods for describing the outside boundary of the highlighted small hexagonal region (the four hashed hexagonal pixels). We will consider that all the previous hexagons (those before the one marked with a small white circle) have been previously encoded. In figure 1 these previously defined pixels are shaded a uniform gray level. This means the start hexagon for the next contour to be encoded is always the top-left hexagon of this contour description.

The two methods of tracing a route around the contour are:

6MTC – Six-connected Movement Through Centres The boundary is encoded on a hexagonal grid moving from centre to centre of each hexagon. There are six possible movement directions, see figure 1, giving an unusual Freeman direction code. The code consisting of four movements steps that is required to describe the small contour (hashed pixels) is then: **1, 2, 4, 5**. It is to be noted that a movement step can never go diagonally back on itself; for example after a **2** move the next move can not be a **0** move as this would have meant the path should have taken a **1** move at the initial step. So at every stage there is a maximum of 5 possible moves giving a maximum of $\log_2(5) \approx 2.322$ bits per movement step.

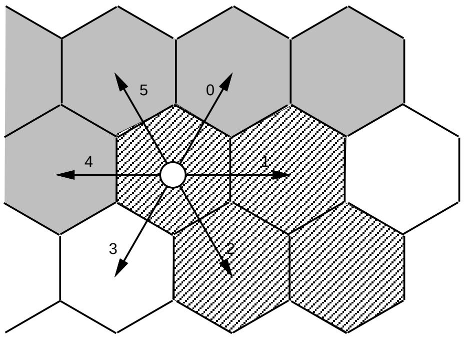


Figure 1: *Hexagonal boundaries. Included are the centre-to-centre coding values: **0**, ..., **5** that are used with the **6MTC** encoding method. The four hexagonal pixels that are diagonally hashed represent the next contour to be encoded and the hexagonal pixel with the white circle is the first (top-left) pixel to be considered when describing the boundary route. The uniform gray shaded pixels are those pixels that have been previously completely defined.*

6MAE – Six-connected Movement Along Edges An alternative is to move along the outside edges of the hexagonal shape. This has the great advantage that at each stage there is only one of two choices; Left **L** or Right **R**. So starting at the top-right vertex from the white circle and ending at the bottom-left vertex, the route is **L, R, R, L, R, R, R, L, R, R, L**. This is a longer code but as there is a binary choice there is only a maximum of one-bit required at each step.

2.2. Restrictions in the Freedom of the Boundary Route

Both of these routes have restrictions due to the clockwise route taken around the contour, and the fact that the path will neither overlap with a previously defined contour, nor incorporate hexagonal pixels prior to the starting location, nor will the path intersect itself. There is also a restriction as the contour is constrained within the borders of the original image dimension. For coding purposes all of these restrictions are relatively easy to implement and can drastically reduce the coding complexity and therefore the final number of bits required to represent a contour. The streams of bits can be sent through an entropy encoder, for example a modified Arithmetic Encoder (see later section) but the **6MAE** method having a binary choice could dispense with this if efficiency is not important. For **6MAE** as there is at each step only a binary decision, if one of the routes is impossible (for example when there is a pixel previously defined) then the other route *must* be taken and that particular step is not required to be encoded.

2.3. Problem of Overlapping Contours for Eight-connected Square Pixels

As is well known a square pixel within a raster array can be either four-connected or eight-connected depending on the number of nearest neighbours it can be connected to. Eight-connected regions allow for a larger set of different areas to be considered as a single region, but have the issue of being able to overlap. This is an extra property that has benefits in reducing the number of regions that need to be encoded but brings some unintuitive problems. There is a set of choices to be made if an algorithm accepts overlapping eight-connected contours, and these have been previously briefly described by the author [TW96]. Figure 2 illustrates the advantage in reducing the small test image from three contours to just two contours. It is possible to create an algorithm that does not allow overlapping contours, but still wishes to use eight-connected contours as a descriptor, this can deterministically be carried out in an algorithm, but leads to another choice as shown in figure 3. Finally figure 4 illustrates a choice in area filling that is in effect arbitrary. The region unfilled, currently in white, on the right, can be filled in with either of the two contour values; light or dark gray.

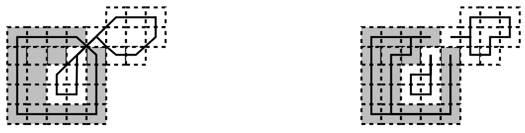


Figure 2: *Example of overlapping contours: part 1. The first image shows two overlapping eight connected contours, while the second image shows the same image having to be represented as three four-connected contours.*

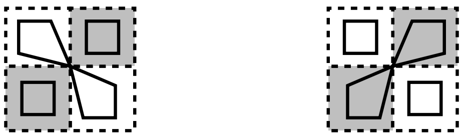


Figure 3: *Example of overlapping contours: part 2. These two images illustrate the standard conundrum as to which order eight-connected contours should be connected if overlapping contours are not allowed.*

Overlapping contours also have interesting behaviours when you consider creating a hierarchical description for all the contours in an image. Then specific contours may have multiple parents at different levels of the tree structure. This again causes non-intuitive behaviour especially noticeable when viewing interactively on a graphical display. We will describe in section 2.5 the probability advantage that eight-connected contours potentially have, but this is let down in implementation by the extra complexity required. Six-connected contours for hexagonal pixels are of a similar property to four-connected contours for square pixels

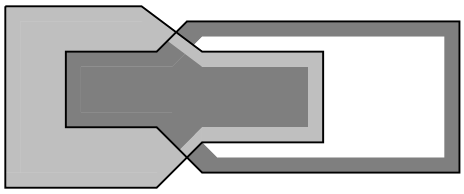


Figure 4: *Example of overlapping contours: part 3. This image illustrates the problem of allowing eight-connected overlapping contours as there is no obvious answer as to how to fill-in the white region on the right. Both options of light gray and dark gray are justifiable according to the rules.*

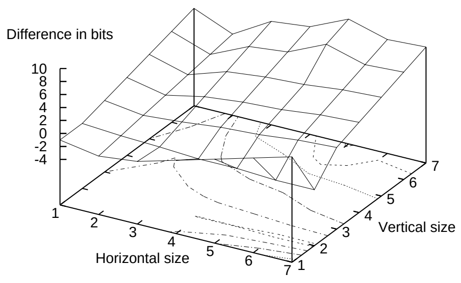
in that they are impossible to overlap so have none of these encoding and representational disadvantages.

2.4. Movement Through Centres vs Movement Along Edges

The advantage **6MTC** codes has over **6MAE** codes is that the number of coding steps required is likely to be fewer for a particular contour, but each step is likely to be more complex. As contours become large it can be demonstrated that **6MAE** becomes more efficient, but this is not true in all cases including certain small contours. To consider this the hexagonal semi-rectangular contours whose sides are less than eight (49 in total) were individually encoded. Table 1 shows the numerical differences when encoding each of the contours with the two methods. Apart from an unusual anomaly concerning vertical $2 \times n$ contours, only for very small contours is **6MTC** a better option. The few cases that have negative values in table 1 are highlighted in bold. This indicates that except for rare images there is a coding advantage in using an **6MAE** method.

2.5. Hexagonal Pixels Compared with Square Pixels

When considering a hexagonal pixel raster array as a replacement for the standard square raster display format the layout to be considered is fairly straightforward with each raster line being displaced by half a pixel width and stretched to take into account the fact that hexagons are taller than they are wide (or wider than they are tall depending on how they are arranged). This stretching ratio is T/D as will be defined specifically later. We will assume the hexagon area ΔH is the same value as the area of a square pixel ΔS, and let N be the distance between the centres of two four-connected square pixels (that is the width of a pixel). We are considering only the scenario of square pixels within a raster display which is often considered as a *de facto* standard for computer graphics post-processing of an image. The issues of mimicking and converting a rectangular array of pixels, with an hexagonal structure can be derived from the same standard formulas. This hexagonal structure described is illustrated in Figure 5 and the following mathematical properties



	1	2	3	4	5	6	7
1	**-1.00**	**-1.00**	0.00	1.00	2.00	3.00	4.00
2	**-2.00**	**-1.42**	**-0.83**	**-0.25**	0.34	0.92	1.51
3	**-1.36**	**-1.68**	**-0.36**	0.97	2.29	3.61	4.93
4	0.29	**-2.03**	0.29	1.61	2.93	4.25	5.58
5	1.93	**-2.03**	0.61	1.93	3.25	6.90	8.22
6	3.58	**-2.39**	1.25	2.58	3.90	5.22	6.54
7	5.22	**-2.39**	1.58	2.90	4.22	5.54	6.86

Table 1: *Difference values between the number of bits calculated by simple entropy to encode a **6MAE** boundary description compared with a **6MTC** boundary description, for all the 49 small semi-rectangular hexagonal contours.*

hold that relate hexagonal distance values to N. From the area preserving equality property we have the following

$$\Delta H = \frac{3}{2} D^2 \tan\left(\frac{\pi}{6}\right) = \frac{3}{2\sqrt{3}} D^2 = N^2 = \Delta S \qquad (1)$$

and then we can calculate the approximate values (to 4dp) for the geometric hexagonal constants.

$$D = N\sqrt{\left(\frac{2\sqrt{3}}{3}\right)} \approx 1.0747N \qquad (2)$$
$$L = \frac{D}{2} \approx 0.5373N \qquad (3)$$
$$R = D\frac{\sqrt{3}}{2} \approx 0.9306N \qquad (4)$$
$$T = 2\left(R - \frac{L}{2}\right) \approx 1.3239N \qquad (5)$$
$$H = D\tan\left(\frac{\pi}{6}\right) = \frac{D}{\sqrt{3}} \approx 0.6204N \qquad (6)$$

and now we can calculate the ratio

$$T/D \approx 1.2321 \qquad (7)$$

There is of course the alternative hexagonal arrangement, derived by rotating figure 5 by 90 degrees, leading to a shift of half a pixel vertically in every pixel column.

What the formula estimates tells us is that R the line-to-line distance for each raster line is smaller than the equivalent in a normal raster display, but the centre-to-centre distance D is slightly larger. We now assume that there exists a function $q(\ldots)$ which when given a distance will return a probability that two points, this distance away from each other, will have the same digitized value. Now we can consider the probability P_n that a pixel is the same value and therefore connected to one of its neighbours depending on how n-connected the pixel is. Square pixels can be either four- or eight-connected and hexagonal pixels can be six-connected. So we have,

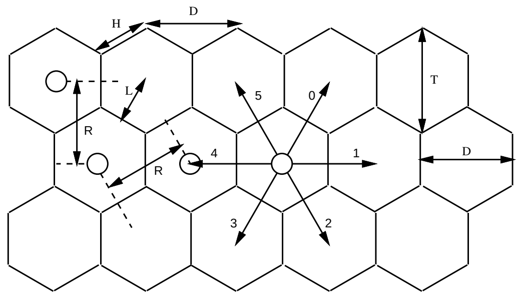


Figure 5: *Hexagonal Layout for the proposed pixel display arrangement. This illustrates all the key distances between the hexagonal pixels.*

$$P_4 = \max(1, 4q(N)) \qquad (8)$$
$$P_6 = \max(1, 6q(1.0747N)) \qquad (9)$$
$$P_8 = \max\left(1, 4q(N) + 4q(\sqrt{2}N)\right) \qquad (10)$$

Figure 6 shows the normalised autocorrelation graph (see appendix A) for the standard mandrill image, and although this is an extreme example it is representative of certain image types, and indicates that the correlation between pixels reduces as the distance between them increases. Below this in figure 7 is the histogram of the same image, but carried out for pixel-pairs resulting in a 256×256 bin surface plot. This numerically indicates again a strong correlation between close pixels for this image and of course similar image types. For the mandrill image we can calculate from this pixel-pair histogram the normalised correlation function, $\rho = 0.987$.

The hypothesis states that when we have the property that $P_6 > P_8$ then hexagonal pixels of the same area as square pixels are likely to yield more connected regions than eight-connected square pixels would. It is also likely that in most cases $P_6 > P_4$ which would mean we have an improved representation and have none of the overlapping problems of eight-connected regions. These points will be individually highlighted and discussed later.

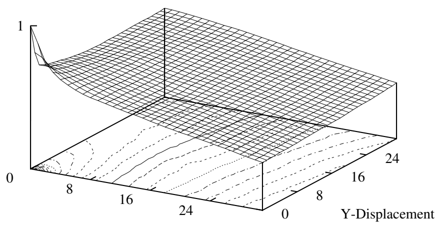


Figure 6: *Normalised autocorrelation function for the mandrill test image. In images due to content and alignment there is often an increased correlation along one of the axis with respect to the other, which means autocorrelation functions are rarely $x-y$ symmetric.*

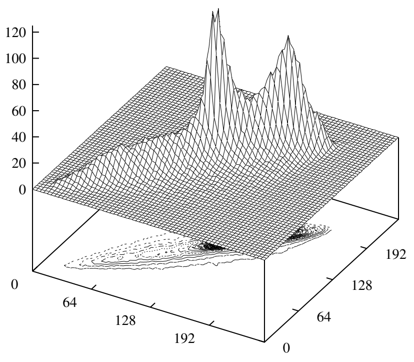


Figure 7: *Pixel pair histogram – mapped with a two dimensional contour for the mandrill test image. A standard 1D histogram can be obtained by summing the bin values along either the x or y direction. A key feature to note is the spread away from the main diagonal $x = y$, which indicates how correlations vary depending on specific intensity values.*

So there are two key advantages of using a hexagonal description;

1. correlations between neighbouring pixel values could be higher than both square four-connected and square eight-connected pixels, resulting in increased coding efficiency, and
2. there are fewer special cases including the problem of dealing with overlapping contours that can occur when you consider eight-connected square pixel regions.

2.6. Single Pixel Contours

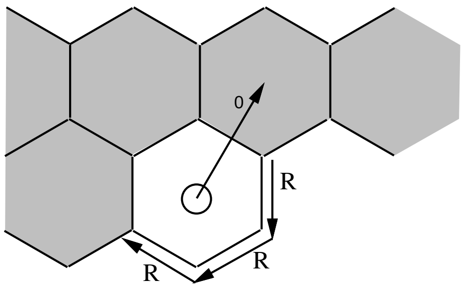


Figure 8: *Single pixel contour being encoded by both **6MTC** and **6MAE** methods.*

The **6MAE** method can encode a single pixel contour very simply with the three step path **R, R, R** but the **6MTC** method has no code to describe "go nowhere". A solution, illustrated in figure 8 is to encode a single pixel contour as an initial **0** move as this move is impossible because it would result in moving to a previously defined pixel – that is one on a previous raster line which we have already assumed as being completely define (marked as a uniform gray colour).

2.7. Comma-free Boundary Codes

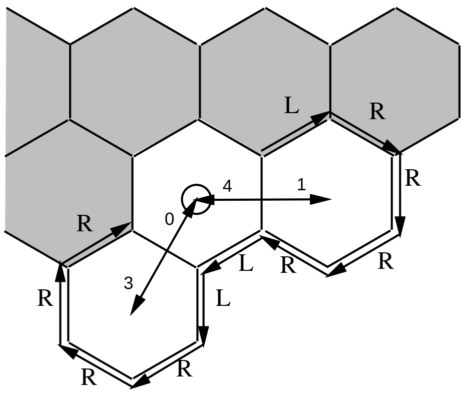
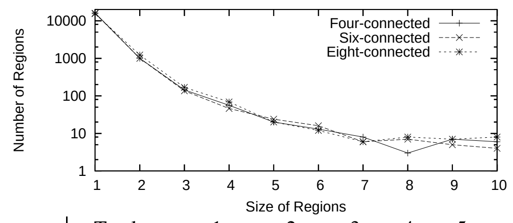


Figure 9: *Simple three pixel contour encoded with a non comma-free code using **6MTC** and a comma-free code using **6MAE**.*

A comma-free code is one where the end of the code is completely deterministic without any extra information, whereas a non-comma free code may need to transmit an explicit end instruction. For boundary codes it appears that the end is always deterministic, which is whenever we reach the start pixel again, but this is not the case for certain contours using the **6MTC** coding strategy. Consider a route **1,**

4 that moves to the right and then returns to the start pixel. This code is actually a pre-fix code to many possible longer codes for example **1, 4, 3, 0** which defines a three pixel contour. The start pixel is this case is traveled through twice. In fact the start pixel can be traveled through either once or twice in any contour boundary route so an extra code may be required to be included to terminate the route.

When considering the **6MAE** boundary code there are no special cases and the code is always comma-free, which simplifies the implementation. The route for this simple example is then **L, *R*, R, R, R, *L*, L, R, R, R, *R***. While looking at this apparently long code three of the entries have been italicised that indicate routes that are deterministic and do not need to be encoded and also it is noted that in any **6MAE** there will be more **R** moves leading to a bias in probabilities to aid coding, which leads onto the next section.



	Total	1	2	3	4	5	>5
Four	17510	16225	1005	144	56	20	60
Six	17239	15978	996	136	46	24	59
Eight	17038	15490	1227	169	69	20	62

Table 2: *Number of connected regions by size for the resampled test image. It can be seen that the graph shows a near power law relationship for small region size values.*

2.8. Arithmetic Coding of the Stream Probabilities

Encoding square pixels using four-connected and eight-connected movement steps when using **MTC** codes could be a simple task as they can be compactly represented in “two” (four combinations) or “three” (eight combinations) bits per movement step. Encoding square pixels using a **MAE** code is in fact complex as there are three possible movement routes; $\log_2(3) \approx 1.5850$ representing Left **L**, Right **R** and Forwards **F**. For hexagonal pixel layouts as has been previously described this is reversed and in binary notation, the hexagonal **6MAE** technique is simpler to implement as it only requires one bit per movement step – two possible options; **L** and **R**.

In efficient implementations to obtain maximum compression ratio, none of this is a great issue as modern entropy encoding techniques for example variations within the range of Arithmetic Coders [MNW98] can efficiently encode a small set of symbols with arbitrary probabilities. One advantage can be applied though to the binary **6MAE** method in the use of specialist Arithmetic Coding tools derived through the family of coders based upon the IBM Q-Coder [PMLA88,HV92]. These are designed specifically for very fast operations when there are only two possible symbols **0** and **1** to be encoded.

3. Image Processing Operations using a Hexagonal Contour Description

Due to the potential packing and colour blending benefits that an hexagonal layout allows, the first use will be for image and video capture. With modifications and approximations all image processing and data compression algorithms can be applied to a hexagonal pixel structure. Over correspondence with one of the reviewers it is noted that efficient direct affine transformation operations may result in larger sampling errors, and be potentially slower.

A possible solution is to apply image processing operations directly to the individual contour description, allowing faster transformations to occur, for example a translation or rotation can be applied to the boundary description stream rather than to all the pixels individually. More complex contour operations can be and have been considered including; contour error diffusion, texture repainting methods, edge smoothing techniques as well as simplification merging of multiple contours [Tur00a, TE01].

An image processing extension for future work is to consider the packing arrangements in the next dimension; to accommodate both 3D representations and moving images, which will require more complex arrangements than the equivalent current voxel arrangements [Tur04].

4. Enlargement Image Test Experiment: Image Creation

Finally we wish to consider a synthetic but practical test to demonstrate the probability distributions for different square and hexagonal raster based images.

To create a pair of test images a very high-resolution image was resampled to create both a hexagonal pixel and a square pixel image, both having the same spatial sampling. A test image whose original size is 3236×4001 resulted in a 119×172 hexagonal pixel image and a 129×160 square pixel image. This used an even area sampling reconstructor. Table 2 shows the number of regions at each size for the three cases of four-connected, and eight-connected regions for the square pixel image; and six-connected regions for the hexagonal pixel image.

As can be seen the actual numbers are very similar, but we can calculate the normalised correlation, ρ for each of the three cases. This can be defined for neighbouring pixels, as

$$\rho = \frac{E\left(f_{i+1}f_i\right)}{E\left(f_i^2\right)} = \frac{\sum_i f_{i+1}f_i}{\sum_i f_i^2} \tag{11}$$

where f_i are the pixel values and $E(\ldots)$ is the expected value. We then have for a four-connected pixel, $\rho(1) \approx 0.965$, a six-connected pixel, $\rho(1.0747) \approx 0.961$, and for a diagonal-connected pixel, $\rho(\sqrt{2}) \approx 0.950$. This is a simple indicator to highlight the reduction in correlation. According to certain studies [WCAG07] for a "correctly" scanned image ρ should lie above 0.85.

This also goes someway to demonstrate the properties that are hoped for when we defined equations 8-10 and the required inequalities for these probability functions P_n that will be needed to convince its wide spread use.

5. Conclusions

We have demonstrated a set of important properties that will allow future 'honeycomb' or hexagonal image descriptors to be efficiently and easily represented.

- Hexagonal six-connected pixels almost certainly will have a higher connectivity ratio when compared to four-connected pixel regions.
- Hexagonal six-connected pixels have none of the overlapping-problems that exist when considering eight-connected pixel regions.
- A **6MAE** boundary description of a hexagonal six-connected pixel region is a comma-free and efficient binary coding system with no special cases. It has the properties that a) except for unusual hexagonal raster images it is more efficient than a **MTC** description and b) it is easier to implement than the equivalent square pixel **MAE** method.

If maximum connectivity is required then eight-connected is an option, but due to all the special issues and more importantly the arbitrary choices that need to be made when implementing either an overlapping or non-overlapping eight-connected contour coder creating a consistent standard will be hard. As such the obvious conclusion is to implement six-connected contour coding and use a hexagonal raster array instead of a square raster array. This advantage has been demonstrated both by probability and algorithmic complexity studies. Unfortunately, we will still have to convince manufacturers and wait for the wide-spread deployment of hexagonal inspired cameras, scanners and displays.

Hexagonality for pixel layout is the future – it just still may take some time to arrive.

Appendix A: Efficient way to Calculate the Normalised Autocorrelation Function

The two dimensional normalised autocorrelation function R' is simple to define but can resulting in very large intermediary values.

$$R_{i,j} = \sum_{x,y=-\infty}^{\infty} f_{x,y} f_{x+i,y+j} \tag{12}$$

$$R'_{i,j} = \frac{R_{i,j}}{R_{0,0}} \tag{13}$$

It is possible to calculated this function from the normalised difference correlation function D',

$$\begin{aligned}
D_{i,j} &= \sum_{x,y=-\infty}^{\infty} \left(f_{x,y} - f_{x+i,y+j}\right)^2 && (14)\\
&= \sum_{x,y=-\infty}^{\infty} \left(f_{x,y}^2 - 2f_{x,y}f_{x+i,y+j} + f_{x+i,y+j}^2\right) && (15)\\
&= \sum_{x,y=-\infty}^{\infty} 2\left(f_{x,y}^2 - f_{x,y}f_{x+i,y+j}\right) && (16)\\
&= 2\left(R_{0,0} - R_{i,j}\right) && (17)\\
D'_{i,j} &= \frac{D_{i,j}}{2R_{0,0}} && (18)\\
&= 1 - R'_{i,j} && (19)
\end{aligned}$$

In the bounded image case there is a possibility of a slight error due to edge conditions, but final computational accuracy is likely to be higher.

References

[Fre61] FREEMAN H.: On the encoding of arbitrary geometric configurations. *IRE Transactions on Electronic Computers EC-10* (June 1961), 260–268.

[HV92] HOWARD P. G., VITTER J. S.: Analysis of arithmetic coding for data compression. *Information Processing and Management 28*, 6 (November 1992), 749–763.

[KTA*04] KUBO N., TAKEMURA K., ADACHI K., NISHIMURA T., TAMAYAMA H., IWABE K., YAMADA T.: *Super Dynamic Range Image Processing System Using a New Structure CCD.* Tech. rep., Fujifilm Research and Development, 2004. No 49 UDC 621.383.7+621.397.422.

[MNW98] MOFFAT A., NEIL R., WITTEN I. H.: Arithmetic coding revisited. *ACM Transactions on Information Systems 16*, 3 (July 1998), 256–294.

[MS01] MIDDLETON L., SIVASWAMY J.: Edge detection in a hexagonal-image processing framework. *Image and Vision Computing 19*, 14 (December 2001), 1071–1081.

[PMLA88] PENNEBAKER W. B., MITCHELL J. L., LANGDON, JR G. G., ARPS R. B.: An overview of the basic principles of the q-coder adaptive binary arithmetic coder. *IBM Journal of Research and Development 32*, 6 (1988), 717.

[TE01] TURNER M., EVANS A.: Analysis of edge based smoothing techniques for facsimile images. In *Image Processing III: Third IMA Conference on Image Processing* (2001), Ellis Horwood, pp. 152–178.

[TN92] TILTON J. C., NOVIK D. A.: Adjustable lossless image compression based on a natural splitting of an image into drawing, shading and fine grained components. In *Space and Earth Science Data Compression Workshop* (March 1992), Tilton J. C., (Ed.), NASA Conference Publication 3183, pp. 17–25.

[Tur00a] TURNER M.: Design of entropy conscious and constrained image operations using a contour tree image definition. In *Image Processing II: Third IMA Conference on Image Processing* (2000), Ellis Horwood, pp. 22–25.

[Tur00b] TURNER M.: Optimised black-and-white contour coding. In *Machine Graphics and Vision* (May 2000), vol. 9, Institute of Computer Science Polish Academy of Sciences 01-237 Warsaw, Ordona 21, Poland (in cooperation with Association for Image Processing, Poland), pp. 397–402.

[Tur04] TURNER M.: Shell creation, representation and visualisation from an arbitrary $f(x,y,z) = 0$ polynomial shape description. In *Eurographics UK Chapter* (June 2004), IEEE Computer Society, pp. 196–172.

[TW96] TURNER M. J., WISEMAN N. E.: Efficient lossless image contour coding. *Computer Graphics Forum 15*, 2 (June 1996), 107–117.

[Tyt00] TYTKOWSKI K.: Hexagonal raster for computer graphics. In *Information Visualization* (2000), IEEE International Conference, pp. 69–73.

[Uli87] ULICHNEY R.: *Digital Halftoning*. MIT Press, 1987.

[WCAG07] WOODS R. E., CZITROM D. J., ARMITAGE S., GONZALEZ R. C.: *Digital Image Processing Third Edition*. Prentice Hall, 2007.

[WS91] WÜTHRICH C. A., STUCKI P.: An algorithmic comparison between square- and hexagonal-based grids. *CVGIP: Graphical Models and Image Processing 53*, 4 (July 1991), 324–339.

[YK93] YONG-KUI L.: The generation of straight lines on hexagonal grids. *Computer Graphics Forum 12*, 1 (1993), 27–31.