MVA2015 IAPR International Conference on Machine Vision Applications, May 18-22, 2015, Tokyo, JAPAN

A Novel Spiral Addressing Scheme for Rectangular Images

Min Jing, Bryan Scotney, Sonya Coleman
University of Ulster
United Kingdom
{m.jing;bw.scotney;sa.coleman}@ulster.ac.uk

Martin McGinnity
Nottingham Trent University
United Kingdom
martin.mcginnity@ntu.ac.uk

Abstract

Spiral architectures have been employed as an efficient addressing scheme in hexagonal image processing (HIP), whereby the image pixel indices can be stored in a one-dimensional vector that enables fast image processing. However, this computational advance of HIP is hindered by the additional time and effort required for conversion of image data to a HIP environment, as existing hardware for image capture and display are based predominantly on traditional rectangular pixels. In this paper, we present a novel spiral image processing framework that develops an efficient spiral addressing scheme for standard square images. We refer to this new framework as "squiral" (square spiral) image processing (SIP). Unlike HIP, conversion to the SIP addressing scheme can be achieved easily using an existing lattice with a Cartesian coordinate system; there is also no need to design special hexagonal image processing operators. Furthermore, we have developed a SIP-based non-overlapping convolution technique by simulating the "eye tremor" phenomenon of the human visual system, which facilitates fast computation. For illustration we have implemented this technique for the purpose of edge detection. The preliminary results demonstrate the efficiency of the SIP framework by comparison with standard 2D convolution and separable 2D convolution.

1 Introduction

Hexagonal image processing (HIP) [5] has attracted attention recently as it has shown positive outcomes for fast processing in edge and corner detection [1,2,4,7,8]. Unlike a square lattice, the points in a hexagonal lattice are difficult to address using Cartesian coordinates. A spiral architecture that enables an efficient addressing scheme for a hexagonal lattice has therefore been developed [9]. As shown in Fig. 1, the spiral addressing scheme originates at the centre of the hexagonal image and spirals out using one-dimensional indexing, with each pixel having six equidistant neighbours. A major advantage of such a spiral addressing scheme is that any location in the image can be represented by a single coordinate value, enabling the spiral image to be stored as a vector.

A major problem affecting the advancement of HIP is that almost all existing hardware for capturing and displaying images is based on a rectangular architecture. Therefore, extra effort is required to convert a square- or rectangular-based image to a hexagonal-based structure before any indexing or processing can take place. In addition, such conversion to a hexagonal image can introduce distortions due to the hexagonal lattice not being aligned in two orthogonal directions; for example, the image may need to be shifted by a half-pixel to simulate capture on a hexagonal lattice.

Inspired by the spiral architecture for HIP, we now propose a novel spiral addressing scheme for standard rectangular pixel based images, to which we refer as "squiral image processing"(SIP). The advantages of SIP are threefold. Firstly, conversion is much easier than for HIP. Whereas converting to HIP from a rectangular pixel-based image requires a process that typically involves pixel sub-division, followed by regrouping and averaging of sub-pixel values for pixel reconstruction, for SIP we can use directly the pixels in a square image. Any conversion involves merely shifting points within the same Cartesian coordinate system. Secondly, SIP is designed for square images, so it can be implemented using existing hardware. Thirdly, an image in SIP can be stored as a vector, thus retaining the computational advantage of HIP, and there is no need to design special hexagonal image processing operators as the kernel for square images can be applied to SIP images too.

In this paper we first explain the proposed SIP addressing scheme and conversion to SIP from the standard Cartesian 2D addressing scheme. We then demonstrate the development of SIP-based non-overlapping convolution by adopting an approach that mimics eye tremor in the human vision system. For illustration we use the application of edge detection, but the technique is applicable to any convolution-based operator or feature extractor.The preliminary results for performance evaluation demonstrate the efficiency of the SIP-based approach.

2 Spiral Architecture for a Square Image

2.1 Spiral Architecture

An illustration of a one-dimensional addressing scheme for HIP is shown in Fig. 1. It is seen that the address starts from the centre pixel (layer-0), then moves from pixel 1 to 6 indexed in a clockwise direction. The cluster of the centre pixel together with its six (layer-0) neighbours is considered as layer-1. It can be noticed that layer-2 is generated by recursive use of layer-1 clusters, such that seven layer-1 clusters are combined to form layer-2. Ultimately, the entire hexagonal image can be considered as a layer-λ cluster comprising 7^{λ} pixels.

Inspired by the spiral addressing for HIP, we propose a new spiral scheme for square images. Similar to HIP, the SIP image originates at the centre of a square image and spirals out using one-dimensional indexing. An illustration for the proposed SIP addressing scheme is given in Fig. 2. The SIP layer-1 consists of nine pixels (layer-0 clusters), and layer-2 consists of nine layer-1 clusters, i.e, 81 pixels. Higher layers are generated recursively, similarly to HIP but with nine clusters combined each time rather than seven.

This structure facilitates the use of base 9 numbering to address each pixel within the image. For example,

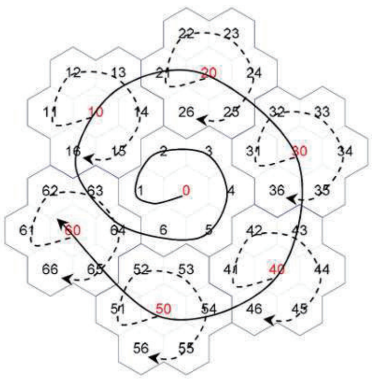


Figure 1. The spiral architecture for a layer-2 hexagonal image.

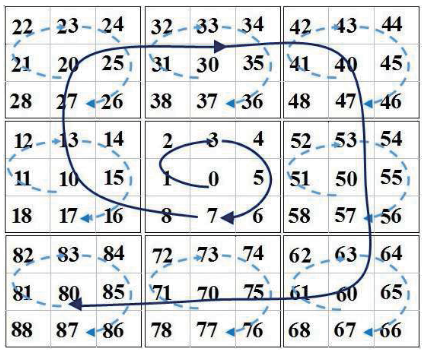
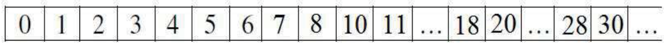


Figure 2. The spiral addressing scheme for a layer-2 SIP-based image.



0	1	2	3	4	5	6	7	8	10	11	...	18	20	...	28	30	...		

Figure 3. One-dimensional address values for SIP image.

the pixels in layer-1 are labelled from 0 to 8, indexed in a clockwise direction. The base 9 indexing continues into each layer, e.g. layer-2 starts from 10, 11, 12, ..., and finishes at 88. Subsequent layers are structured recursively. The SIP image is stored in a one-dimensional vector with addresses as illustrated (in part) in Fig. 3.

2.2 SIP Conversion

HIP conversion relies on use of a re-sampling scheme [3,5] to match the location of points in the square and hexagonal images. As SIP is based on square images, no re-sampling scheme is needed. We need only to convert the lattice of a square image to the new SIP format based on the spiral addressing scheme. The steps are as follow.

For a given image with size $M \times N$, the number of SIP layers λ can be found by $\lambda = (logM + logN)/log(9)$; then the length of the SIP image is 9^{λ}. Because the SIP address scheme is base 9, the conversion between the SIP address and a decimal number can be found by $(a_n a_{n-1}...a_1) = a_n \times 9^{n-1} + a_{n-1} \times 9^{n-2} + ... + a_1$, where the values a_i of a SIP address are $0 \le a_i < 9$. We can adapt the spiral addressing scheme for HIP [9] and the SIP address can be represented as:

$$a_n a_{n-1}...a_1 = \sum_{i=1}^{n} a_i \times 10^{i-1} \qquad (1)$$

where $\sum$ denotes Spiral Addition and $\times$ indicates Spiral Multiplication [5]. For example, the point at SIP address 867 can be located by finding the addresses of 800, 60 and 7. Next, we explain how to locate these SIP addresses in a standard 2D square image.

For a point represented by Cartesian coordinate (x,y), we define the centre point as $L(0) = (0,0)$. Based on the SIP addressing scheme, we can find the points in layer-1: L(1)=(-1,0), L(2)=(-1,1), L(3)=(0,1), L(4)=(1,1), L(5)=(1,0), L(6)=(1,-1), L(7)=(0,-1) and L(8)=(-1,-1). To locate the points in a higher layer, we calculate the number of pixels required to shift the point from the centre to the target point by

$$L(a_i \times 10^{i-1}) = 3^{i-1} \times L(a_i) \qquad (2)$$

For example, based on Eq.(1) and Eq.(2), the point L(87) can be located by $L(87) = L(80) + L(7) = 3 \times L(8) + L(7) = 3 \times (-1,-1) + (0,-1) = (-3,-4)$. To locate a point at L(4536), $L(4536) = L(4000) + L(500) + L(30) + L(6) = 3^3 \times L(4) + 3^2 \times L(5) + 3 \times L(3) + L(6) = 27 \times (1,1) + 9 \times (1,0) + 3 \times (0,1) + (1,-1) = (37,29)$. Hence the point L(4536) can be found by shifting the start point from (0,0) to (37,29). Based on these shifting rules, we can then convert a square image to a SIP image (which is stored as a vector). Next, we explain how such a SIP image can be used in a fast processing approach by developing "eye tremor"-based SIP convolution.

3 SIP Convolution via Eye Tremor

3.1 Simulation of Eye Tremor

Standard feature extraction is usually executed via convolution, where typically a gradient-based operator is applied to a pixel and its neighbours. The process of determining these neighbours in a one-dimensional addressing scheme is not straightforward. For HIP, determining a pixel's neighbours requires time consuming special hexagonal and radix-7 addition [5]. To tackle this issue, a biologically inspired framework [8] has been proposed for HIP by modelling eye tremor. The concept of eye tremor, rhythmic oscillations of the eye, has been exploited in image processing [6]. The traditional approach to feature extraction based on overlapping convolution operators does not closely represent the human visual system. Furthermore, the human visual system does not process single static images, but instead a series of temporal images that are slightly off-set due to involuntary eye movements. Thus [8] proposed a non-overlapping convolution that can be implemented by simulating eye tremor.

Inspired by [8], we develop the eye tremor framework for the SIP case. In Fig. 4, consider I_0 as a "base" SIP image, and eight additional images $I_j, j = 1, 2, ..., 8$ can be obtained by shifting I_0 by one pixel in the image plane along the proposed spiral addressing scheme. The "centre" of each image I_j is located at a pixel within the layer-1 neighbourhood centred at image I_0. Each image is stored as a vector after being converted from the 2D image structure, which enables us to achieve fast and efficient processing for feature extraction.

I_2	I_3	I_4
I_1	I_0	I_5
I_8	I_7	I_6

Figure 4. Illustration of the 9 SIP image centres in the eye tremor approach.

3.2 SIP based Convolution

Feature detection operators are often based on first derivative approximations. For hexagonal image feature detection, x- and y-components of a hexagonal operator need to be designed accordingly to compute the derivative [1]. This is unnecessary for SIP as SIP is designed for square images, and so any standard operator (e.g., size 3×3 or 9×9) can be converted directly to a SIP vector. For example, for the Sobel operator in the x- and y-directions:

$$Sobel_x = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}, Sobel_y = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix} \quad (3)$$

to perform convolution with a SIP image, we first flip the kernel matrix and then convert it into a layer-1 SIP vector, generating $Sobel_x = [0, 2, 1, 0, -1, -2, -1, 0, 1]$ and $Sobel_y = [0, 0, 1, 2, 1, 0, -1, -2, -1]$. For a given image I_0, convolution of a Sobel operator (denoted as H_1) across the entire image plane is achieved by applying the operator sparsely to each of the nine images $I_j, j = 0, ..., 8$ and then combining the resultant outputs. In the eye tremor framework, in each of the images I_j we apply the operator H_1 only when centred at those pixels with spiral address 0(mod 9), hence achieving non-overlapping convolution. In its general form, convolution of the SIP image I_j with an operator H_λ can be defined as

$$G_\lambda^j(s_0) = \sum_{s \in N_\lambda(s_0)} H_\lambda(s) \times I_j(s), \quad (4)$$

where $\forall s_0 \in \{s | s = 0(mod 9))\}$ and $N_\lambda(s_0)$ denotes the λ-neighbourhood centred on the pixel with spiral address s_0 in image I_j. For the Sobel example, the matrix implementation of convolution of I_0 with H_1, based on Eq. (4), can be written as:

$$\begin{pmatrix} G_1^0(0) \\ G_1^0(10) \\ \vdots \\ G_1^0(k) \end{pmatrix} = \begin{pmatrix} I_0(0) & I_0(1) & ... & I_0(8) \\ I_0(10) & I_0(11) & ... & I_0(18) \\ \vdots & \vdots & \ddots & \vdots \\ I_0(k) & I_0(k+1) & ... & I_0(k+8) \end{pmatrix} \begin{pmatrix} H_1(0) \\ H_1(1) \\ \vdots \\ H_1(8) \end{pmatrix} \quad (5)$$

where $k = 0, 10, 20, 30, ...$ We can apply the same process to the remaining eight images $I_j, j = 1, ..., 8$; each is a SIP image created by shifting the origin by one pixel from I_0. The outcome can be obtained by assembling the results of G_1^j as,

$$\begin{pmatrix} G_1^0(0) & G_1^1(0) & \cdots & G_1^8(0) \\ G_1^0(10) & G_1^1(10) & \cdots & G_1^8(10) \\ \vdots & \vdots & \ddots & \vdots \\ G_1^0(k) & G_1^1(k) & \cdots & G_1^8(k) \end{pmatrix} \quad (6)$$

The above process is illustrated in Fig. 5. The final outcome in a SIP format is obtained by rearranging each row of the matrix (Eq.(6)) into a vector,

$$[G_1^0(0)G_1^1(0)...G_1^8(0)G_1^0(10)G_1^1(10)...G_1^8(10)...G_1^0(k)...G_1^8(k)] \quad (7)$$

Based on the process above, SIP based approach achieves exactly the same outcome as standard 2D convolution. In 2D convolution using a kernel of size $A \times B$ requires $A \times B$ multiplications for each sample. For an image of size $M \times N$ four loops are needed to complete $M \times N \times (A \times B)$ multiplications. For SIP-based convolution, each sample still involves $A \times B$ multiplications, but since the SIP image is stored as a vector, all of the processes above can be executed in a more efficient manner.

4 Performance Evaluation

We evaluate the proposed SIP approach using edge detection as the application, and we use the Sobel edge detector for illustration. We have used three well-known test images: lena, peppers and coins, with image sizes 100×100, 384×520 and 738×900 respectively. Each image was converted into a SIP image, and in which the number of SIP layers for each image are: layer-4(81×81), layer-5(243×243) and layer-6 (729×729) respectively. We compare the SIP approach to both the standard 2D convolution and to the Matlab built-in function *conv2.m* (optimised by separable convolution). Implementation of standard 2D convolution involves use of four *for* loops that moves the flipped kernel along each row and column of the input image and then computes the weighted sum over the neighbourhood. For separable convolution, the 2D convolution is performed by two one-dimensional convolutions (assuming that the 2D filter is separable), so is much faster than the standard 2D convolution approach.

The results of Sobel edge detection are shown in Fig. 6, in which the left column shows the results from the original images by standard 2D convolution and in the right column are those from SIP images. The extent of the region of SIP images is also highlighted within the original images in Fig. 6(left column). It is clearly shown that the edge detection results from SIP are the same as those from the standard approach, except that the size of the SIP image is slightly smaller. The correlation of edge results between SIP-convolution and standard convolution confirms that both approaches provide identical results (Table.2). Note that the original images are used for visual comparison here (there is no other difference between the standard and SIP convolution results). In the following comparison of run-times, the edge detection was performed for the same image size as the SIP image in each case.

For SIP conversion, we pre-calculated the coordinate shifting table up to layer-6 (total 533441 pixels). This look-up table needs to be calculated only once and then saved for use in any SIP conversion. The average time for conversion of a square image to one layer-5 SIP (243×243) is 0.065 sec. Because the focus of this evaluation is on the convolution, the time required for SIP conversion was not considered for the comparative evaluations. The run-times include the time for calculating the gradient in the x- and y- directions and

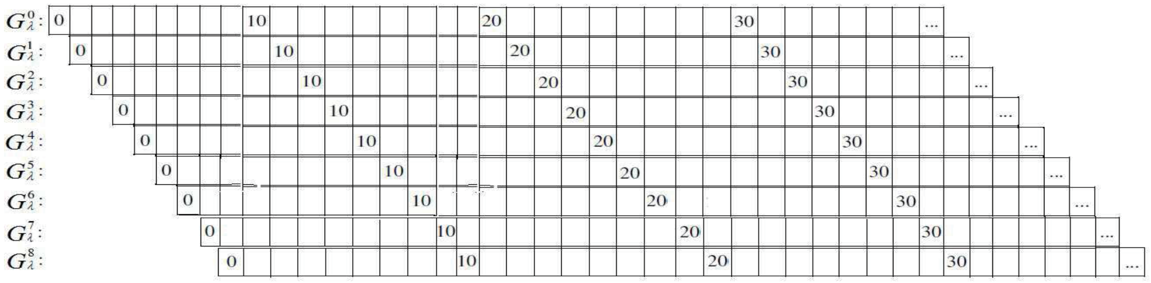


Figure 5. Illustration of assembling the one-dimensional vectors for SIP based convolution.

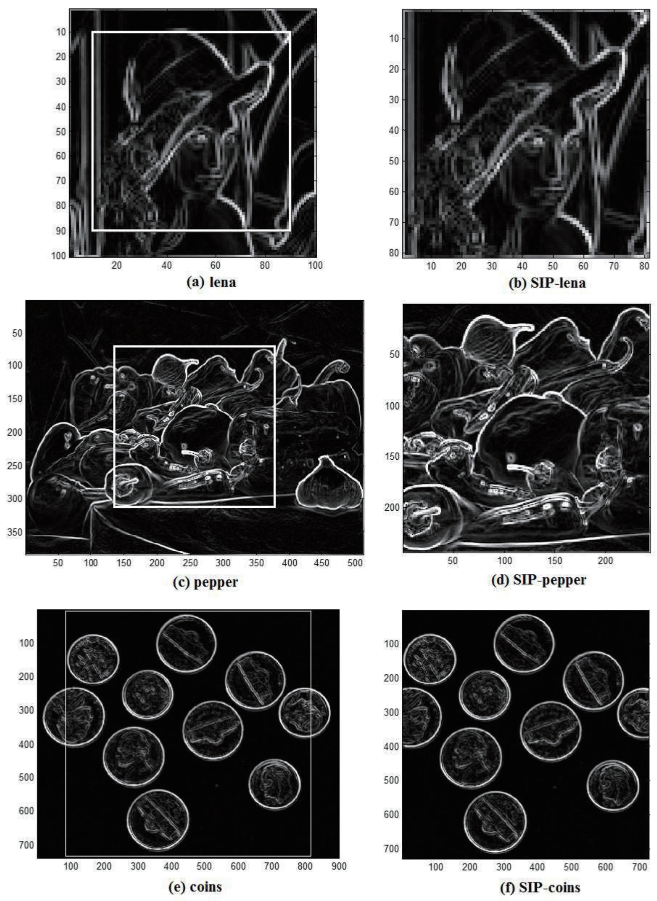


Figure 6. Sobel edge detection by standard 2D convolution (left column) and SIP convolution (right column). The SIP image size is also highlighted in the original images (left).

magnitude of both. The run-times based on an average of 100 runs are presented in Table 1. It can be seen that the SIP approach is not only much faster than standard 2D convolution, but also comparable to the separable convolution, demonstrating the efficiency of the proposed SIP framework.

5 Conclusion

We have presented a novel spiral image processing framework for use with standard rectangular pixel-based images. This framework enables fast feature detection by use of a spiral architecture in conjunction with eye tremor and non-overlapping convolution. In the SIP framework, images can be converted easily from their existing typically rectangular coordinate system. Therefore SIP can be incorporated directly with existing hardware systems and state-of-the-art image processing methods for square images. The preliminary results demonstrated for edge detection show the computational efficiency of the SIP approach. Further development will consider the application of SIP for interest point detection and efficient feature representation in real-time image and video retrieval.

Table 1. Average Run-times (Seconds)

Images	Lena	Pepper	Coins
Standard	0.006826	0.061737	0.605872
SIP	0.000282	0.001130	0.016618
Separable	0.000284	0.001305	0.014501

Table 2. Correlation with Standard Convolution

Images	Lena	Pepper	Coins
SIP	1.0	1.0	1.0

References

[1] S. Coleman, B. Scotney and B. Gardiner,:" A Biologically Inspired Approach for Fast Image Processing," In IAPR Proc. Machine Vision Applications, 2013.

[2] X. He, et al.:"An Approach to Edge Detection on a Virtual Hexagonal Structure," Digital Image Computing Techniques and Applications, pp. 340-345, 2007.

[3] I. Her:"Geometric transformations on the hexagonal grid," IEEE Transactions on Image Processing, 4(9), pp. 1213- 1222, 1995.

[4] S. J. Liu, S. Coleman, D. Kerr, et al,: "Corner Detection on Hexagonal Pixel based Images," In Proc IEEE ICIP, 2011.

[5] L. Middleton and J. Sivaswamy,:"Hexagonal Image Processing; A Practical Approach," Springer 2005.

[6] A. Roka, et al.: "Edge Detection Model Based on Involuntary Eye Movements of the Eye-Retina System," Acta Polytechnica Hungarica, 4(1), pp. 31-46, 2007.

[7] D. Kerr, S. Coleman, M. McGinnity et al,:"Biologically Inspired Edge Detection," In Proc. IEEE ISDA, 2011.

[8] B. Scotney, S. Coleman and B. Gardiner, :"Biologically Motivated Feature Extraction Using the Spiral Architecture", In Proc. IEEE ICIP, 2011.

[9] P. Sheridan, T. Hintz and D. Alexander:"Pseudo-invariant Image Transformations on a Hexagonal Lattice," Image and Vision Computing, vol. 18, pp. 907-917, 2000.