Image Rotation without scaling on Spiral Architecture

Qiang Wu, Xiangjian He, Tom Hintz

Department of Computer Systems
Faculty of Information Technology
University of Technology, Sydney,
PO BOX 123, Broadway Street
Sydney, NSW, 2007
Australia
{wuq, sean, hintz}@it.uts.edu.au

Abstract

Spiral Architecture is a relatively new and powerful approach to general purpose machine vision system. On this novel architecture, image rotation is achieved by Spiral Multiplication. However, the general image rotation on Spiral Architecture has two effects. One is scaling segmentation and the other is rotation. This paper presents an algorithm to achieve image rotation without scaling on Spiral Architecture, which improves the Spiral Architecture's usage in image processing.

Keywords: image rotation, Spiral Architecture, image segmentation, distributed image processing

1. INTRODUCTION

The transformation to be presented in this paper is based on a novel data structure, Spiral Architecture [Sheridan91], which is inspired from anatomical considerations of the primate's vision [Schwartz80]. From the research about the geometry of the cones on the primate's retina we can conclude that the cones' distribution has inherent organization and is featured by its potential powerful computation abilities. The cones with the shape of hexagons are arranged in a spiral clusters. This cluster consists of the organizational units of vision. Each unit is a set of seven-hexagon [Sheridan00] as shown in Figure.

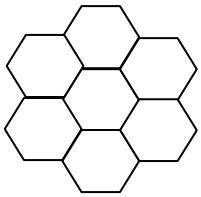

Figure 1. Seven-hexagon unit of vision

In the traditional rectangular image architecture, a set of 3×3 rectangles is used as the unit of vision and each pixel has eight neighbor pixels. In Spiral Architecture any pixel has only six neighbor pixels which have the same distance to the center hexagon of the seven-hexagon unit of vision. So the Spiral Architecture has the possibility to save time for global and local processing. For example, chain coding is a useful way to extract object contour. Spiral Architecture is able to simplify the coding scheme [He99].

A natural data structure that emerges from geometric consideration of the distribution of photo receptors on the primate's retina has been called the Spiral Honeycomb Mosaic (SHM) and is presented in details in [Sheridan99]. SHM is made up of the hexagonal lattices, which are identified by a designated positive number individually. The numbered hexagons form the cluster of size 7^n. The hexagons tile the plane in a recursive modular manner along the spiral direction [Alexander95]. An example of a cluster with size of 7^2 and the corresponding addresses are shown in Figure 2.

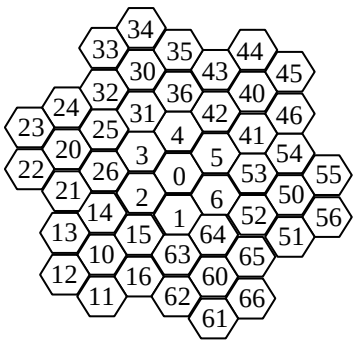

Figure 2. SHM with size of 49

SHM contains very useful geometric and algebraic properties, which can be interpreted in terms of the mathematical object, Euclidean ring. Two

algebraic operations have been defined on SHM: Spiral Addition and Spiral Multiplication. After image is projected onto SHM, each pixel on the image is associated with a particular hexagon and its SHM address. Then these two operations mentioned above can be used to define two transformations on SHM address space respectively, which are translation of image and scaling rotation of image. This paper only concerns about image rotation, which is achieved through Spiral Multiplication generally. In order to simplify our presentation, the relative knowledge about Spiral Multiplication will be explained briefly in Section 2 accordingly.

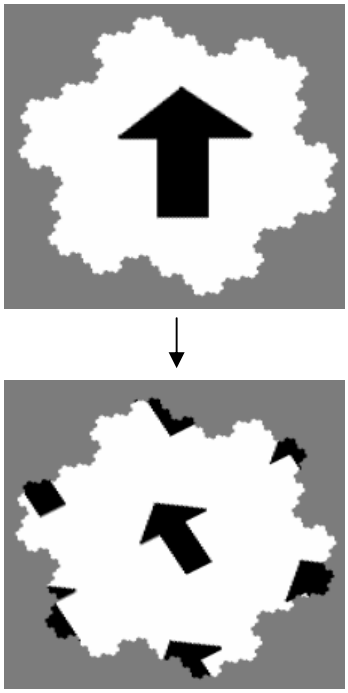

Figure 3. Spiral Multiplication by a common SHM address (56123) not 10^n

There is no doubt that these two algebraic operations light a gateway for the application of Spiral Architecture in image processing. Unfortunately the traditional Spiral Rotation is not a pure image rotation. It has two effects, scaling segmentation and rotation (See Figure 3).

Like the traditional rectangular image architecture it is very necessary to develop a method to achieve image rotation without scaling on Spiral Architecture, which is surely profitable to improve the Spiral Architecture's usage in image processing.

In this paper, a method called backward walk-jump rotation on Spiral Architecture is presented, which achieves image rotation without scaling on Spiral Architecture.

The organization of this paper is as follows. Spiral Multiplication is briefly reviewed in Section 2. A new way called backward jump-walk image rotation on Spiral Architecture is presented in Section 3. The experiment results are demonstrated in Section 4. Conclusion can be seen in Section 5.

2. SPIRAL MULTIPLICATION

SHM is a subset of the complex plane. Spiral Multiplication is an arithmetic operation with closure properties defined on SHM addressing system so that the resulting product will be SHM address in the same finite set on which the operation is performed [Sheridan96]. In addition, Spiral Multiplication incorporates a special form of modularity.

In order to achieve Spiral Multiplication, a scalar form of Spiral Multiplication is defined in Table 1.

	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6
2	0	2	3	4	5	6	1
3	0	3	4	5	6	1	2
4	0	4	5	6	1	2	3
5	0	5	6	1	2	3	4
6	0	6	1	2	3	4	5

Table 1. Spiral Multiplication Table

Multiplication of address a by the scalar $\alpha(\alpha \in \{0,1,\cdots,6\})$ is obtained by applying scalar multiplication to the components of a according to the above scalar form, and denoted by,

$$\alpha(a) = (\alpha a_n \ \alpha a_{n-1} \ \cdots \ \alpha a_1) \quad \text{where} \quad a = (a_n a_{n-1} \cdots a_1) \text{ for } \forall a_i \in \{0,1,\cdots,6\} \tag{1}$$

If the address in Spiral Multiplication is not a scalar, α, but a common address like,

$$b = (b_n b_{n-1} \cdots b_1) \text{ for } \forall b_i \in \{0,1,\cdots,6\} \tag{2}$$

then

$$a \times b = \sum_{i=1}^{n} a \times b_i \times 10^{i-1} \tag{3}$$

where $\sum$ denotes Spiral Addition and $\times$ denotes Spiral Multiplication. Surely *carry rule* is required in Spiral Addition to handle the addition of numbers composed of more than one digit.

In order to guarantee that all the pixels are still located within the original Spiral area after Spiral Multiplication, a modular multiplication on SHM is defined. Furthermore the transformation through Spiral Multiplication defined on SHM is bijective mapping. That is each pixel in the original image maps one-to-

one to each pixel in the output image after Spiral Multiplication.

Modular Multiplication is shown as follows, Let p be the product of two elements a,b. That is,

$$p = a \times b,\ a,b \in SHM \tag{4}$$

If $p \geq (\text{modulus})$, then
if a is a multiple of 10 map p to

$$\left(p + (p \div (\text{modulus}))\right)\text{mod}(\text{modulus}) \tag{5}$$

Otherwise, map p to

$$p \bmod(\text{modulus}) \text{ where } \text{modulus} = 10^n \tag{6}$$

Here, it is assumed that the number of hexagon in spiral area is 7^n.

3. SPIRAL IMAGE ROTATION WITHOUT SCALING

The Spiral image will rotate for a specific angle after it is multiplied by a specific SHM Address. Unfortunately, it is followed by the scaling down effects (See Figure 3). It limits the Spiral Architecture's usage in image processing while competing with the traditional rectangular architecture. Although this scaling effect will not affect the object recognition processing [He98][He99] when affine integral invariant representation is used in feature extraction, the image restoration is usually required to maintain the original shape and size during image rotation.

After analysis it is found that the rotation by Spiral Multiplication is actually the rotating segmentation rather than the pure rotation. This is further explained in the following.

It is supposed that an object is made up of seven hexagonal pixels on Spiral Architecture as shown in Figure 4. If this image is multiplied by Spiral

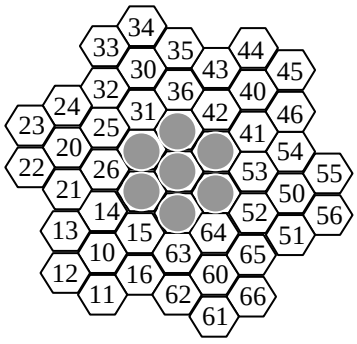

Figure 4. One Supposed Object on Spiral Architecture

address 10, the image segmentation is shown in Figure 5. The arrows on the image show how the

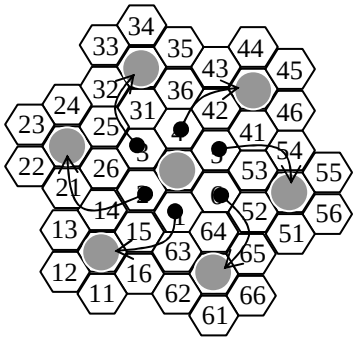

Figure 5. Rotating Segmentation on Spiral Architecture

original pixels are separated. Naturally, if each pixel can be kept on its corresponding circle with the center of Spiral Architecture during image rotation, the size of image will remain. However, if the above proposal is workable, such circle must be defined for each pixel individually and an algorithm must be developed to pull back the pixels to a proper location on its corresponding circle after rotating segmentation.

If it is assumed that that the rotation center is the center of Spiral Architecture, it is not too difficult to define a concentric circle for each pixel, whose radius is the distance of each pixel to the center of Spiral Architecture. However, it costs us much memory to store the information of the circles for all the pixels. Another problem is that it is impossible to get the algorithm to pull back the pixels, which flew out of the original image area, because we are unable to locate the pixels' positions if they are out of the original image area on Spiral Architecture after Spiral Multiplication. In this paper, a novel method called backward walk-jump is developed to avoid the difficulties mentioned above and still achieve the goal, Spiral rotation without scaling.

In order to avoid image segmentation, Spiral Multiplication can't be performed on the original image directly. Before the introduction of the new algorithm, let us see an interesting and meaningful fact in Spiral Multiplication. If the original image is multiplied by Spiral addresses, 1, 2, 3, 4, 5 and 6, the image will rotate by 0°, 60°, 120°, 180°, 240° and 300° respectively without changing the size (See for example, Figure 6). The original object is an up-right arrow as shown in Figure 3.

The rotation angle is determined by vector $\vec{01}$ and vector $\vec{0X}$ (0, 1 and X are SHM address values and X is multiplier. Please refer to Figure 2.). The angle can be separated into two parts properly based on the interesting facts mentioned above. Let us suppose the rotation angle as θ. It is expressed as a sum of two parts as follows,

$$\theta = k \times 60° + \alpha, \text{ where } k \in \{0,1,\cdots,6\} \text{ and } \alpha \in [0°,60°) \quad (7)$$

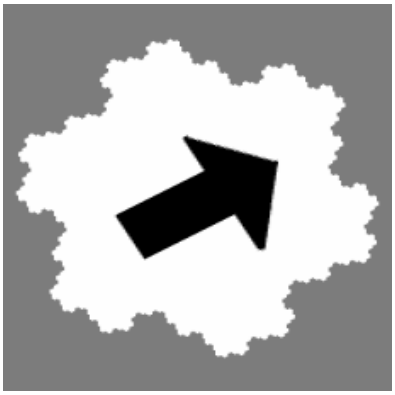

Figure 6. Spiral Multiplication by SHM address, 2

Two steps are developed to realize image rotation without scaling on Spiral Architecture.

Step 1: Walking Operation

Let us focus on the rotation effect in Spiral Multiplication only. Formula (7) means that for any pixel except the center pixel, if it is multiplied by a Spiral address whose corresponding angle is θ, it will walk for α degree clockwise around the center of Spiral Architecture to a new position. In order to prevent the pixels from walking out of the trajectories to result in the distortion, walking is forced to be performed on a circle with the center of Spiral Architecture as its center. So any pixel can find a most approximate position along the circle after rotating/walking for α degree. The pixel on the center of Spiral Architecture remains at the same position. According to formula (7), walking happens only within one pie slice of $60°$, so it won't take a long time.

Step 2: Jumping Operation

In the 2nd step, the point jumps about k pie slices of $60°$ clockwise. This operation can be done by multiplying SHM address of the new position after walking operation with $k+1$ (SHM address) through Spiral Multiplication.

In order to demonstrate the procedure of the above method, this way is used to rotate a single-pixel object as follows.

Example. Suppose a single-pixel object located on SHM address 63 (See Figure 7). Spiral Multiplication with the multiplier 26 (SHM address) will be performed on this image. The result of normal Spiral Multiplication can be seen in Figure 8. It shows that the normal Spiral Multiplication results in the distortion during rotation.

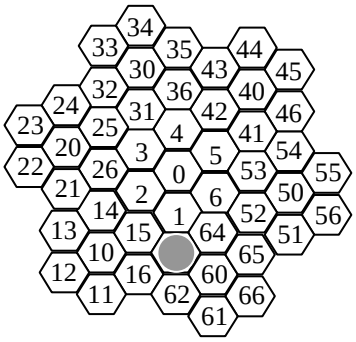

Figure 7. Single-Pixel Object

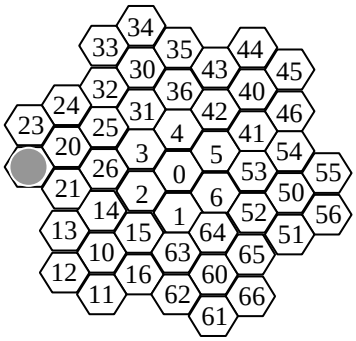

Figure 8. Normal Spiral Multiplication: 63×26

Now walk-jump algorithm is used to achieve image rotation on Spiral Architecture. According to formula (7), the initial rotation angle can be presented as,

$90° = 1 \times 60° + 30°$.

That means, in formula (7),

$\theta = 90°, k = 1, \alpha = 30°$.

So walking operation moves this object from position 63 to 15 (See Figure 9).

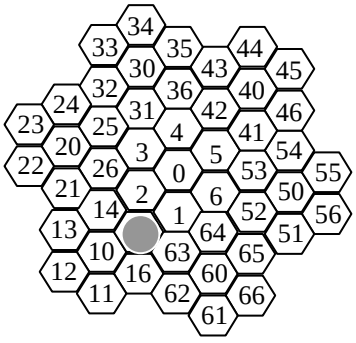

Figure 9. Single-Pixel Object walking to a new position

In the next step, jumping operation is achieved through Spiral Multiplication with SHM address, $k+1=2$. Figure 10 shows the final result of image rotation without scaling to this single-pixel object.

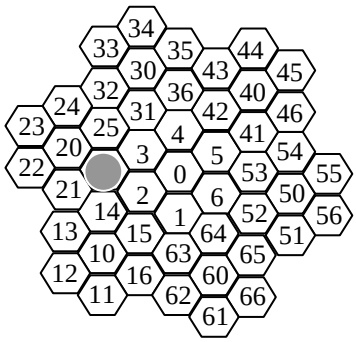

Figure 10. Final result of image rotation without scaling to single-pixel object

In fact, the above walk-jump rotation is a forward projection from the input image to the

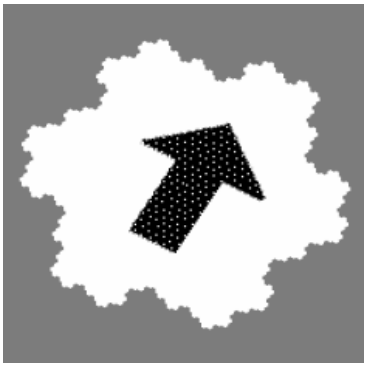

Figure 11. Results of forward walk-jump rotation

output image. It is obvious that jumping operation by Spiral Multiplication is a bijective mapping but walking operation is not. In other words, many positions are nil positions, which have no corresponding positions in the input image. So forward walk-jump rotation results in many gaps in the output image (See Figure 11). Original image is an up-right arrow.

To guarantee that each pixel in the output image has its corresponding value in the input image, backward mapping approach replaces forward mapping approach. That means on the output image the pixel positions are checked one by one to find the nearest position on the input image according walk-jump rotation. The gray value of the corresponding pixel on the input image is regarded as the gray value for the pixel on the output image. The backward mapping has two differences from the formal forward mapping approach:

1) Walking anti-clockwise in walking operation on the output image;
2) Jumping k regions anti-clockwise or multiplying Spiral address of the new position after walking operation with the inverse value of $k+1$ (SHM address) based on Spiral Multiplication principle on the output image (Refer to Section 2 and [Sheridan96]).

4. EXPERIMENTAL RESULTS

As a simplified illustration of our algorithm without loss of generality, an image containing an up-right arrow and an image containing a toy duck (See Figure 12) are used here. There are totally 16807 (7^5) hexagonal pixels in the Spiral Architecture area.

Suppose the original image is multiplied by Spiral address 21. The corresponding angle of this multiplier is $79.11°$. So from formula (7) we get $k=1$ and $\alpha=19.11°$. The result after walking operation can be seen from Figure 13.

Then all the pixels jump 1 region or multiplied by the Spiral address 2. Figure 14 shows the final result of rotation without scaling on Spiral Architecture.

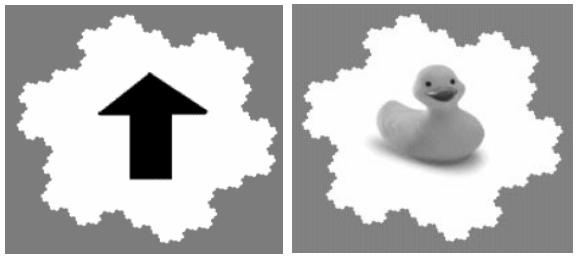

Figure 12. Up-right arrow and toy duck

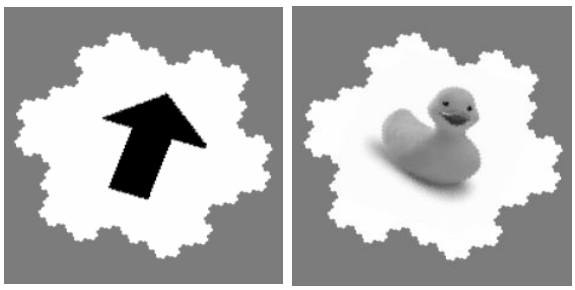

Figure 13. Result after walking *19.11°*

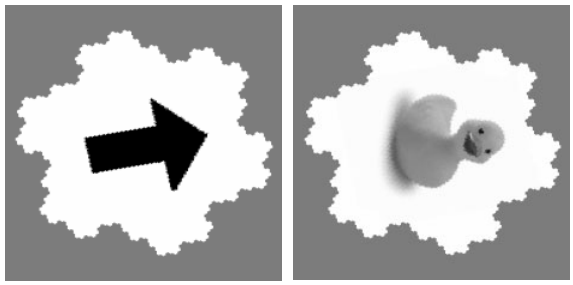

Figure 14. Result of rotation without scaling on Spiral Architecture (The original image rotates *79.11°*.)

Remark In order to make Spiral Architecture practically workable on the existing image capture device, mimic Spiral Architecture is used in the research work [He99]. Due to a few differences between real Spiral Architecture and mimic Spiral Architecture, a little distortion is introduced into the image during image rotation (See Figure 13, Figure 14). But it does not affect the theoretical research about Spiral Architecture. Surely, this distortion will be resolved with the development of image capture hardware device.

5. CONCLUSION

This paper presents the deep research work about image rotation on Spiral Architecture. A new algorithm is developed to realize image rotation without scaling on Spiral Architecture. From the experimental results we see the objective is achieved successfully. It successfully improves the Spiral Architecture's usage in image processing.

This algorithm has two steps. Most of processing time is taken by the first step, walking operation. During the research work, it is found that if polar coordinates operations are introduced into Spiral Architecture, it will save much processing time but it needs more memory to store the information for mapping Spiral Architecture to polar coordinates.

REFERENCE

[Sheridan91] Sheridan P., Hintz T., and Moore W., Spiral Architecture in Machine Vision, In T. Bossamier, Editor, Australian Occam and Transputer Conference, IOS Press, Amesterdam, 1991

[Schwartz80] Schwartz E., Computation natomy and functional architecture of striate cortex: a spatial mapping approach to perceptual coding, Vision Research 20, pp645-669, 1980

[Sheridan00] Sheridan P., Hintz T., and Alexander D., Pseudo-invariant image transformations on a hexagonal lattice, Image and Vision Computing 18 (11), pp907-917, 2000

[Sheridan99] Sheridan P. and Hintz T., Primitive image transformations on a hexagonal lattice, Technical report, Charles Sturt University, Bathurst, NSW, December 1999

[Alexander95] Alexander D., Recursively modular artificial neural network, PhD thesis, Macquire University, Australia, 1995

[Sheridan96] Sheridan P., Spiral Architecture for machine vision, PhD thesis, University of Technology, Sydney, 1996

[He98] He X. and Hintz T., , Affine integral invariants and object recognition, Proc. 3rd High Performance Computing Aisa Conference and Exhibition (Singapore), pp419-423, 1998

[He99] He X., 2D-object recognition with Spiral Architecture, PhD thesis, University of Technology, Sydney, Australia, 1999