IMAGE EDGE DETECTION IN A MIMIC SPIRAL ARCHITECTURE

Qiang Wu, Tom Hintz and Xiangjian He

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

ABSTRACT

Detection of edge points of 3-dimensional physical objects in a 2-dimensional image is one of the main research areas of computer vision. Object contour detection and object recognition rely heavily on edge detection. In this paper, we present an edge detection scheme using Gaussian Multi-resolution Theory based on a mimic Spiral Architecture. The Spiral Architecture has been described in many papers. Although it has many advantages such as powerful computational features in image processing especially in image edge detection, there is no available image capture device yet to support this structure. Hence, we mimic the Spiral Architecture from the existing image structure. This mimic structure inherits all computational features of the Spiral Architecture. The Gaussian Multi-resolution Theory is used to reduce noise and unnecessary details of the image.

Keywords: Edge Detection, Computer Vision, Spiral Architecture, Image Processing

1 INTRODUCTION

Edge detection plays a key role in computer vision, image processing and related areas. It is a process which detects the significant features that appear as large delta values in light intensities. At an early stage of computation in a large scale computer vision application, an edge map is detected from the original image. It contains major image information and only needs a relatively small amount of memory space for storage. If needed, a replica image can be reconstructed from its edge map.

In the past, various edge detection algorithms were proposed (e.g. [Bergholm87], [Tian00] and [Zhang94]). In this paper, we present a method for edge detection. Our image algebra is established on a mimic Spiral Architecture and the detection algorithm is based on Gaussian Multi-scale Theory [Lindeberg94].

Spiral Architecture described by Sheridan [Sheridan96] is a relatively new data structure for computer vision. The image is represented by a collection of hexagons of the same size (in contrast with the traditional rectangular representation) as displayed in Figure 1. The importance of the hexagonal representation is that it possesses special computational features that are pertinent to the vision process.

Although the Spiral Architecture has many advantages in image processing and computer vision, it is not yet supported by any available image capture device. Hence, it becomes necessary to construct or mimic the

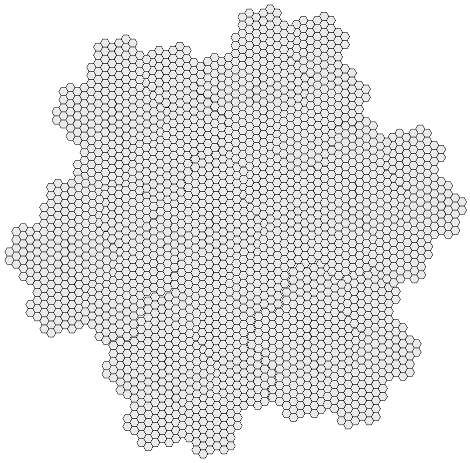

Figure 1: Collection of hexagonal cells.

Spiral Architecture from the existing image structure, on which the traditional image representation is based. In this paper, we will present the mimic Spiral Architecture using the rectangular pixels.

The Gaussian Multi-scale theory introduced by Koenderink [Koenderink84] is a tool to remove image noise. The image brightness function is parameterized. A large change in image brightness over a short spatial distance indicates the presence of an edge. The image is blurred and noise is removed when the parameter is positive. The change in image brightness is described by the derivatives of the brightness function in the gradient directions. The derivatives and the computation of gradient vectors in the mimic Spiral Architecture will also be proposed in this paper.

The content of this paper is arranged as follows. We mimic the Spiral Architecture in Section 2. In Section 3, an approach to the Gaussian multi-scale theory for edge detection including edge definition in the mimic Spiral Architecture is presented. This is followed by an edge detection algorithm in the mimic structure in Section 4. We compare our results in this paper with the previous results derived by He [He99] in Section 5. We conclude in Section 6.

2 THE MIMIC SPIRAL ARCHITECTURE

Traditionally, an image is considered as a collection of rectangular pixels of the same size. Since the late 1990s, edge detection within a relatively new data structure, called the Spiral Architecture has been considered by He, Hintz and Szewcow in their papers [HeH98] and [HeHS98]. This significantly extends and simultaneously makes practical the Spiral image structure. In the Spiral Architecture, an image is represented as a collection of hexagonal picture elements as displayed in Figure 1. The distribution of cones on the retina (see Figure 2) provides the basis of the Spiral Architecture. In the case of the human eye,

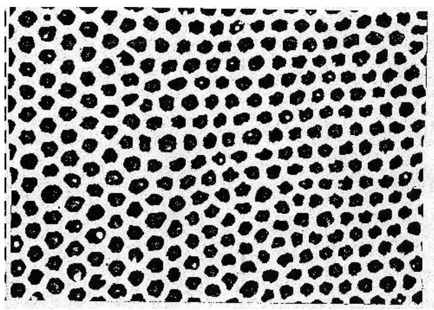

Figure 2: Distribution of cones on the retina.

these elements would represent the relative positions of the rods and cones on the retina.

To construct the mimic Spiral Architecture, we start with a collection of seven hexagonal pixels as show in Figure 3. These seven

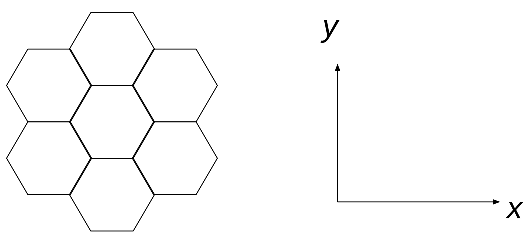

Figure 3: Cartesian coordinates of a cluster of 7 hexagons.

hexagonal pixels are mimiced by twenty-eight (4×7) rectangular (square) pixels, as arranged and shown in Figure 4. A set of four

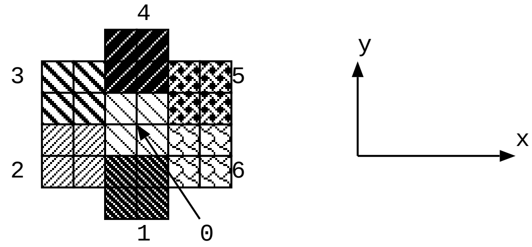

Figure 4: Distribution of 7 pixels constructed from rectangular pixels.

rectangular pixels which are adjacent to each other is used to mimic a hexagonal pixel. The seven mimic hexagonal pixels are numbered from 0 to 6 as shown in Figure 4. These numbers are also called *Spiral Addresses* of (mimic) hexagonal pixels according to Sheridan [Sheridan96]. The grey level (or value) at each mimic hexagonal pixel is computed as the average of the grey values at the four hexagonal pixels, which together form the mimic hexagonal pixel. Figure 5 shows a duck image represented in a mimic Spiral Architecture with 7^5 pixels.

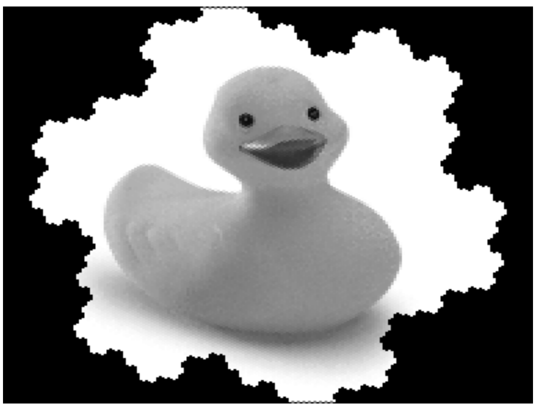

Figure 5: Sample image of 'the duck' in mimic spiral space.

It is obvious that the mapping from a group of four square pixels to a hexagonal pixel as shown in Figure 4 is a one-to-one map. It is easy to see that our mimic is consistent with the important property of hexagon distribution that each such pixel has exactly six surrounding pixels. This mimic Spiral Architecture inherits all computational features of the Spiral Architecture including the computation of *Spiral Addition* and *Spiral Multiplication* which was proposed by Sheridan in [Sheridan96] and then demonstrated to be very powerful in image processing and computer vision.

3 GAUSSIAN THEORY IN THE MIMIC SPIRAL ARCHITECTURE

The Gaussian Scale-space Theory[1] was proposed by Lindeberg [Lindeberg94] to explain how certain aspects of image information can be represented and analysed at the earliest processing stages of a computer vision system. This theory is one of the best understood multi-resolution techniques available to the computer vision and image analysis community [Sporring97]. Gaussian multi-scale theory is used for our edge detection algorithms as a tool to remove image noise. In the following, the image brightness function will be parameterized. A large change in image brightness over a short spatial distance indicates the presence of an edge. The image is blurred and noise is removed when the parameter is positive.

Let $f : \Re^2 \to \Re$ be a brightness function of an image which maps the coordinates of a pixel, (x, y) to a value in light intensities. The scale-space representation $L : \Re^2 \times [0, +\infty) \to \Re$ is defined such that the representation at 'zero scale' is equal to the original signal, i.e.,

$$L(\cdot; 0) = f(\cdot), \tag{1}$$

and the representation at 'coarser scales' is the convolution

$$L(\cdot; t) = g(\cdot; t) * f(\cdot), \tag{2}$$

where $g : \Re^2 \times (0, +\infty) \to \Re$ is the Gaussian kernel

$$g(x, y; t) = \frac{1}{2\pi t} e^{-\frac{x^2+y^2}{2t}}. \tag{3}$$

Lindeberg defined edges from the continuous grey-level image function $L : \Re^2 \times [0, +\infty) \to$

[1] It is a multi-scale theory.

$\Re$ as the set of points for which the gradient magnitude assumes a maximum in the gradient direction [Lindeberg94]. This can be further described as follows.

Let $\bar{v}$ be the gradient of $L(x,y;t)$ at (x,y) for a given t, and $L_x(x,y;t)$ and $L_y(x,y;t)$ be the derivatives of $L(x,y;t)$ with respect to x and y. Denote $L_x(x,y;t)$ and $L_y(x,y;t)$ L_x and L_y respectively. Then $\bar{v}$ is parallel to (L_x, L_y). Furthermore, the derivative of $L(x,y;t)$ in gradient direction at (x,y) is $\sqrt{L_x^2 + L_y^2}$. We denote this derivative by $L_{\bar{v}}$ i.e.,

$$L_{\bar{v}} = \sqrt{L_x^2 + L_y^2}. \tag{4}$$

Hence, by Lindeberg's definition, (x,y) is an edge point (or edge pixel) if and only if $L_{\bar{v}}$ assumes a maximum at (x,y).

Lindeberg's work assumed a continuous space. In this section, we give a discrete approach within the mimic Spiral Architecture.

3.1 Approach to Gaussian operator

Given discrete data in the mimic Spiral Architecture, if we assume that the distance between centres of two neighbouring rectangle pixels on the same row or colum is 1 and the Cartesian coordinates of the mimic hexagon with Spiral Address 0 is (x,y) (Figure 4), the hexagons with the spiral addresses 1, 2, 3, 4, 5 and 6 have Cartesian coordinates (x, y-2), (x-2, y-1), (x-2, y+1), (x, y+2), (x+2, y+1) and (x+2, y-1) respectively. Let us denote these points a_0, a_1, a_2, a_3, a_4, a_5 and a_6 respectively as shown in Figure 6.

Note that the distance square or the value of x^2+y^2 in Equation 3 is 4 between a_0 and a_1 (or a_4), 5 between a_0 and a_2 (or a_3 or a_5 or a_6). Hence, for a given t, we implement the Gaussian convolution at a_0 by

$$\begin{aligned} & L(a_0;t) = L(x,y;t) \\ = \; & \frac{1}{2\pi t}\{L(a_0;0) \\ + \; & e^{\frac{-4}{2t}}[L(a_1;0) + L(a_4;0)] \\ + \; & e^{\frac{-5}{2t}}[L(a_2;0) + L(a_3;0) \\ + \; & L(a_5;0) + L(a_6;0)]\}. \end{aligned} \tag{5}$$

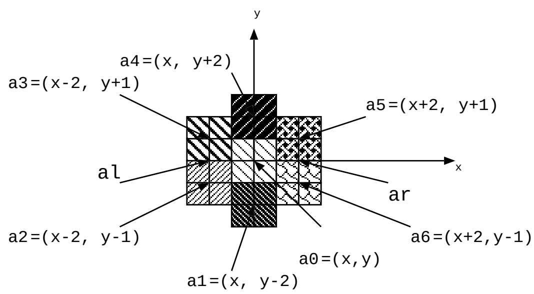

Figure 6: Cartesian coordinates of a cluster of 7 hexagons.

In order to involve more neighbours of a_0 in the convolution, a more general implementation of the convolution is introduced as follows.

1. Choose an integer denoted by J as the number of iterations of the following loop;
2. For $(j = 0; j < J; j++)\{$

$$\begin{aligned} & L(a_0;t) \\ \Leftarrow \; & \frac{1}{2\pi t}\{L(a_0;0) \\ + \; & e^{\frac{-4}{2t}}[L(a_1;0) + L(a_4;0)] \\ + \; & e^{\frac{-5}{2t}}[L(a_2;0) + L(a_3;0) \\ + \; & L(a_5;0) + L(a_6;0)]\}; \\ & L(a_i;0) \Leftarrow L(a_i;t); \end{aligned} \tag{6}$$

$\}$

The above procedure means that

1. Use the simple convolution (Equation 5) to get a convolved value $L(\cdot;t)$ for every hexagon.
2. Use $L(\cdot;t)$ as a new value of $L(\cdot;0)$ at every hexagon.
3. Repeat Step 1 above J times.
4. The newest value of $L(\cdot;t)$ is recorded as the Gaussian convolution value at every hexagon.

From now on, we use the general implementation shown above for the Gaussian convolution, and J is assigned the value of 8 and t is equal to 3.

3.2 Approach to derivatives

Denote a_r the point in the middle of a_5 and a_6 and a_l the point in the middle of a_2 and a_3 as shown in Figure 6. Assume that the light intensity at a_r is the average of the light intensities at a_5 and a_6 denoted by L_r, and the light intensity at a_r is the average of the light intensities at a_2 and a_3, denoted by L_l. Note that

- the distances between a_l and a_0, and between a_r and a_0 are 2, and
- the distances between a_1 and a_0, and between a_4 and a_0 are also 2.

Then we have the following implementation of the derivatives of L with respect to x and y.

$$\begin{aligned} L_x(x,y;t) &= \frac{\frac{L_r-L_0}{2}+\frac{L_0-L_l}{2}}{2} \\ &= \frac{1}{4}(L_r-L_l) \\ &= \frac{1}{8}[L(x+2,y+1;t) \\ &+ L(x+2,y-1;t)] \\ &- \frac{1}{8}[L(x-2,y+1;t) \\ &+ L(x-2,y-1;t)] \qquad (7) \end{aligned}$$

and similarly

$$\begin{aligned} L_y(x,y;t) &= \frac{(L_4-L_0)+(L_0-L_1)}{4} \\ &= \frac{1}{4}[L(x,y+1;t) \\ &- L(x,y-1;t)]. \qquad (8) \end{aligned}$$

It is easy to see that the above representation converges to the real derivatives of L with respect to x and y respectively, as more and more hexagonal pixels are collected. This is because that a_1 and a_4 are the only two pixels closest to a_0 in the $y-$direction, and a_2, a_3, a_5 and a_6 are the only four pixels closest to a_0 in the $x-$direction. The derivative in the gradient direction at (x,y) can then be obtained by applying Equation 4.

Our derivative approximation shown above implies that the derivatives of L at a point or pixel are only influenced by the light intensities of its six neighbouring pixels. So, the derivatives are defined locally. This is an important property which, together with some other properties, leads to a parallel implementation for edge detection. This research will be presented in the paper in preparation by the authors.

3.3 Edge points based on gradient

Recall that $\bar{v}$ is the gradient vector of $L(x,y;t)$ at (x,y). Denote the values of $L_{\bar{v}}$ at a_0, a_1, a_2, a_3, a_4, a_5 and a_6 by L_0, L_1, L_2, L_3, L_4, L_5 and L_6 respectively.

As we are now considering the derivatives in discrete space, the zero-crossing points of the 2nd derivative of L in the gradient direction, denoted by $L_{\bar{v}\bar{v}}$ may not exist and the computation of the higher order derivatives is very time consuming. Hence, we need a method without using the 2nd order derivatives to determine the points or pixels at which the $L_{\bar{v}}$ has a maximum. Our approach is based on the results of the first order derivatives and the value of gradient in the mimic Spiral Architecture.

We propose a procedure to determine whether a_0 is an edge point or pixel as follows refering to Figure 7. In Figure 7, $T1$ cuts the left-hand side edge of pixel 5 (or pixel a_5) at one quarter potision from the top. Similarly, we obtain $T2$, $T4$ and $T5$.

- If the gradient direction (or $\bar{v}$) at pixel 0 (or a_0) is between $T0$ and $T1$, or between $T3$ and $T4$, then it is obvious that pixels 2 and 5 (or a_2 and a_5) among the neighbouring pixels of a_0 contribute the most to the change of brightness (or grey value) at a_0 in the gradient direction. Hence,

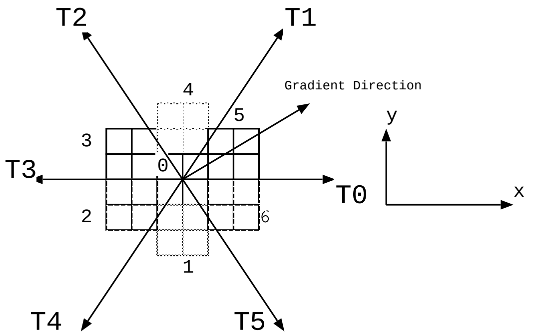

Figure 7: Gradient direction.

- if ($L_0 \geq L_2$ and $L_0 > L_5$) or ($L_0 > L_2$ and $L_0 \geq L_5$),
  we record a_0 as an edge pixel.
  This is because that L_0 in these cases is either largest or smallest among the $L_{\bar{v}}$ values along the gradient direction at a_0.

- Similarly if the gradient direction is between $T1$ and $T2$, or between $T4$ and $T5$
  - if ($L_0 \geq L_1$ and $L_0 > L_4$) or ($L_0 > L_1$ and $L_0 \geq L_4$)
    record a_0 as an edge pixel.

- And if the gradient direction is between $T2$ and $T3$, or between $T5$ and $T0$
  - if ($L_0 \geq L_3$ and $L_0 > L_6$) or ($L_0 > L_3$ and $L_0 \geq L_6$)
    record a_0 as an edge pixel.

The above procedure implies that if a_0 is an edge pixel, then $L_{\bar{v}}$ is a maximum in the gradient direction.

4 EDGE DETECTION FOR 'THE DUCK'

The edge algorithm essentially consists of the following steps:

1. Blur the initial sample image using the Gaussian convolution approach introduced in the previous section. One may use the Edge Focusing Technique described in [HeH98] to obtain the Gaussian scale (a value of parameter t) used for the Gaussian convolution.

2. Threshold[2] the Edge Map obtained in the previous step at a pre-determined grey level.

We use 'the duck' image as displayed in Figure 5 to demonstrate the above algorithm.

When we use the edge definition in the mimic Spiral Architecture as shown in Section 3, a figure containing the edge map of this image is Figure 8.

Figure 8: The edge map of 'the duck'.

Blurring this image using the Gaussian convolution defined in Section 3 with the number of iterations $J = 8$ and the resolution level $t = 3$, the image of 'the duck' (Figure 5 at this coarser resolution level is shown in Figure 9.

Figure 10 is the corresponding edge map of the Gaussian blurred image (Figure 9).

Figure 9 is thresholded at grey level of 32. Its edge map after the thresholding is shown in Figure 11.

It is obvious that the edge map at the coarser resolution level ($t = 3$) is clearer than that of

[2] Thresholding an image at a grey level l is to set the grey values to be $l \times n$ if they are between nl and $(n+1)l$ ($n = 0, 1, 2, \cdots, \frac{256}{l}$).

Figure 9: The Gaussian blurred image of 'the duck'.

Figure 11: Edge map by thresholding the Gaussian blurred image at level 32.

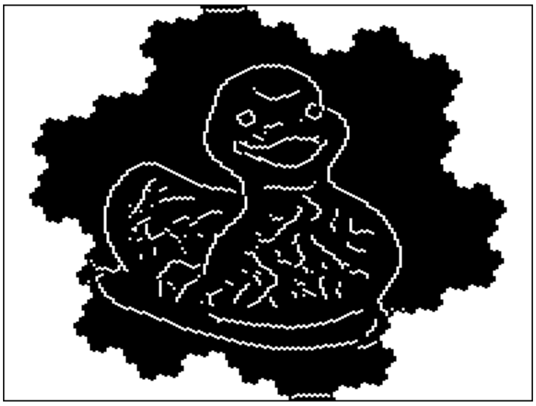

Figure 10: The edge map of the Gaussian blurred 'duck'.

Figure 12: Edge map of Figure 5.

the original image (with $t = 0$). This is because some less critical edge pixels have been removed by the Gaussian filter.

If we are not interested in the details of 'the duck', a rough sketch of it, which is Figure 11, may be more applicable.

5 A COMPARISON

In this section, we compare our experimental results displayed in the previous section with the results shown in [He99].

Figure 12 is the edge map of the sample image (Figure 5) obtained in [He99]. It is found that the edge map we obtain in this paper as displayed in Figure 8 contains less edge points and is a bit clearer than Figure 12.

Figure 13 is the edge image of the Gaussian blurred image of 'the duck' with $J = 30$ and $t = 3$ as shown in [He99]. Comparing this with our edge map as displayed in Figure 10, one will find that our edge map is much clearer than the map in Figure 13. Note that we put $J = 8$ and $t = 3$ to obtain our edge map shown in Figure 10 comparing with $J = 30$ and $t = 3$ used in [He99]. This implies that our Gaussian convolution speed is much faster than the one used in [He99].

6 CONCLUSION

In this paper, we have done the following:

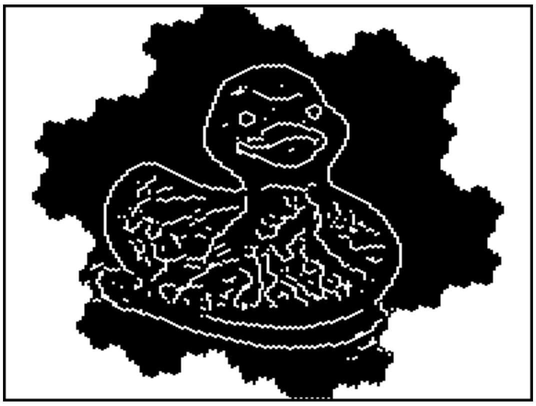

Figure 13: Edge image of the blurred image with $J = 30$ and $t = 3$.

1. We defined the Gaussian convolution operator with discrete data in a mimic Spiral Architecture. This adaption and significant extension to the convolution was first defined in this paper. Its implementation is simple. Its computational speed is fast as it is defined locally.

2. Derivatives of functions defined on the mimic Spiral Architecture were constructed. These derivatives converge.

3. Edge points were defined using only the 1st order derivatives based on the mimic Spiral Arcitecture. The traditional edge detection derived from the zero-crossing points of 2nd order derivatives requires much more time to complete. Furthermore, there is not an easy way to find the zero-crossing points with discrete data.

4. A sample image called 'the duck' was used to demonstrate the efficacy of the edge detection algorithms proposed.

5. A comparison between our algorithm shown in this paper and the one proposed in [He99] is made. This indicates a better resolution using our algorithm.

REFERENCES

[Bergholm87] F. Bergholm. Edge focusing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(6):726–741, 1987.

[He99] X. He. *2D-Object Recognition with Spiral Architecture.* PhD thesis, University of Technology, Sydney, 1999.

[HeH98] X. He and T. Hintz. Multi-scale edge detection based on Spiral Architecture. In *1998 International Computer Symposium*, pages 159–165, Taiwan, 1998.

[HeHS98] X. He, T. Hintz, and U. Szewcow. Replicated shared object model for edge detection with Spiral Architecture. *Lecture Notes in Computer Science*, 1388:252–260, 1998.

[Koenderink84] J. J. Koenderink. The structure of images. *Biological Cybernetics*, 50:363-370, 1984.

[Lindeberg94] T. Lindeberg. *Scale-Space Theory in Computer Vision.* Kluwer Academic Publishers, London, 1994.

[Sheridan96] P. Sheridan. *Sprial Architecture for Machine Vision.* PhD thesis, University of Technology, Sydney, 1996.

[Sporring97] J. Sporring, M. Nielsen, L. Florack, and P. Johansen. *Gaussian Scale-Space Theory.* Kluwer Academic Publishers, 1997.

[Tian00] H. Tian, T. Srikanthan, and K. Asari. A new approach for object boundary extraction based on Heuristic Search. In *Proc. International Conference on Image Science, Systems, and Technology*, pages 693–699, Las Vegas, Nevada, 2000.

[Zhang94] X. Zhang and H. Deng. Distributed image edge detection methods and performance. In *Proc. 6th IEEE Symposium on Parallel and Distributed Processing*, pages 136–143, Dallas, Texas, 1994.