'92 KACC 1992. 10. 19~21

Computation of Partial Derivatives from an Image

Woo Suk Yang* and Inhwan Han**

Dept of Electrical Engineering*

Dept of Mechanical Design and Production Engineering**

Hongik University

ChoChiWon-Up YeonKi-Kun, ChungNam, Korea 339-800

ABSTRACT

Partial derivatives are easily computed analytically assuming that all the geometric information is known. However, there are computational difficulties in getting accurate partial derivatives directly from a range image since an image is a discrete version of continuous data contaminated with some noise. In this paper, we develop a general window function to compute partial derivatives based on the least square surface fitting method. A dynamic selective surface fitting method is introduced to make the window less sensitive to noise. Any degree of partial derivative can be obtained by a simple convolution between an image and window functions.

1. INTRODUCTION

Recently, as fast and efficient range sensors have become commercially available, there has been a tremendous increase in the use of range data for computer vision applications. A range map has an identical image format with commonly used intensity data. But, range data of a 3D object presents explicit information about the surface. The 3D shape from range data directly approximates the 3D shape of the corresponding visible object surfaces. These explicit characteristics of range data make it much easier to represent and recognize general curved objects by their geometric shape. The object in an image is usually represented by surface patches with its surface boundaries so that each patch can be homogeneous in its intrinsic differential geometric properties[1,2-5,10]. Especially, Gaussian and mean curvatures[2-5] are used frequently in the processing of a range image because of its view-point independent characteristics[6,7]. These curvatures are computed from the values of partial derivatives at an image point. In general, partial derivatives are one of the most important features for the recognition of a 3D object using 3D range image.

Partial derivatives are easily computed analytically from an object model assuming that all the geometric information is known[6,7]. However, there are computational difficulties in getting accurate partial derivatives directly from a range image since an image is a discrete version of continuous data contaminated with some noise. Derivatives are very sensitive to the noise in image data. Instead of using direct numerical differentiations, directional derivatives are usually computed and evaluated at the corresponding discrete data point by using a continuous differentiable function that best fits the local surface data within a prefixed window[9]. A polynomial with a certain order is used to fit the local surface patch. Two principal methods are used to find such a differentiable function. The first method is interpolation: a smooth function is found which exactly matches the given values at the given points in the window[6]. The surface function by interpolation usually results in high order polynomials. Although it is possible to approximate any reasonable function

with high order polynomials, polynomials of high order tend to fluctuate widely. Furthermore, interpolation assumes that the data values are exact values from some unknown function. For these reasons, the second method using least square surface fitting which matches the data values as well as possible is preferred[9]. In this paper, we develop a general window function to compute partial derivatives based on the least square surface fitting method. A dynamic selective surface fitting method is introduced to make the window less sensitive to noise. Any degree of partial derivative can be obtained by a simple convolution between an image and window functions.

In section 2, we introduce a general window function to compute partial derivatives based on the least square surface fitting method. For the proof of our algorithm, we do some computer simulations and show their results in section 3. Finally, we conclude this paper in section 4 with some discussions on our approach.

2. COMPUTATION OF PARTIAL DERIVATIVES

First and second order partial derivatives are commonly observed in the processing of a range image. In this paper, we develop a general window function to compute partial derivatives based on the least square surface fitting method.

Considering the small surface patch $z = f(x,y)$ with the size of wxw, we find a best fit, $\hat{f}(x,y)$, which minimizes the mean square error such that

$$\varepsilon^2 = \sum_{(x,y)\in W} \varepsilon^2(x,y) = \sum_{(x,y)\in W} [f(x,y) - \hat{f}(x,y)]^2 \qquad (1)$$

Where

$$\hat{f}(x,y) = \sum_{\substack{i,j=0 \\ i+j\le N}}^{N} a_{i,j} x^i y^j .$$

N denotes the order of a polynomial, and W denotes the wxw window. $x_i y_j$, for $0 \le i+j \le N$, is called the basis of the polynomial.

To find a best fit, $\hat{f}(x,y)$, it is necessary to find the coefficients $a_{i,j}$ of the basis $x_i y_j$. Let $\mathbf{P}_n$ be a class of polynomials of order n. If the order, n, of a polynomial and the size, w, of a window are small, the result may be sensitive to the noise. Since $\mathbf{P}_m \supset \mathbf{P}_n$ for $n<m$, it seems better to use a polynomial with an order as high as possible. However, as the order increases, polynomials tend to fluctuate widely.

Regardless of what the optimal values are for n and w, one question may arise: what is the maximum order of polynomials which guarantees the solution for $a_{i,j}$ with given wxw window W? Or, given N, the order of polynomial, what is the minimum size of the window which guarantees the existence of a solution? There are $(n+1)(n+2)/2$ bases for the polynomials in class $\mathbf{P}_n$. The wxw window has w^2 elements. The size $w^2 > (n+1)(n+2)/2$ does not necessarily guarantee the solution. Assuming a polynomial of order N, let us find a minimum window size wxw which guarantees the existence of the solution, namely the coefficient $a_{i,j}$.

Equation (1) has a minimum only when $\partial\varepsilon / \partial a_{i,j} = 0$ for $0 \le i+j \le N$. Hence, we have

$$\sum_{(x,y)\in W} f(x,y) x^n y^m = \sum_{\substack{i,j=0 \\ i+j\le N}}^{N} a_{i,j} \sum_{(x,y)\in W} x^{i+n} y^{j+m}$$

$$\text{for} \quad 0 \le n+m \le N \qquad (2)$$

Considering the element x_i and y_j of the window, equation (2) can be conveniently expressed in matrix-vector form such that

$$\boldsymbol{V}^T \boldsymbol{V} \boldsymbol{a} = \boldsymbol{V}^T \boldsymbol{f} \qquad (3)$$

where $\boldsymbol{V}$ is a w^2 x$(n+1)(n+2)/2$ matrix. And, $\boldsymbol{a}$ and $\boldsymbol{f}$ are $(n+1)(n+2)/2$x1 column vectors such that

$$\boldsymbol{V} = \begin{bmatrix}
1 & x_0 & y_0 & x_0^2 & x_0 y_0 & y_0^2 & \cdots & y_0^N \\
1 & x_0 & y_1 & x_0^2 & x_0 y_1 & y_1^2 & \cdots & y_1^N \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdots & \cdot \\
1 & x_0 & y_{w-1} & x_0^2 & x_0 y_{w-1} & y_{w-1}^2 & \cdots & y_{w-1}^N \\
1 & x_1 & y_0 & x_1^2 & x_1 y_0 & y_0^2 & \cdots & y_0^N \\
1 & x_1 & y_1 & x_1^2 & x_1 y_1 & y_1^2 & \cdots & y_1^N \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdots & \cdot \\
1 & x_1 & y_{w-1} & x_1^2 & x_1 y_{w-1} & y_{w-1}^2 & \cdots & y_{w-1}^N \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdots & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdots & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdots & \cdot \\
1 & x_{w-1} & y_{w-1} & x_{w-1}^2 & x_{w-1} y_{w-1} & y_{w-1}^2 & \cdots & y_{w-1}^N
\end{bmatrix},$$

and

$$\boldsymbol{a} = \begin{bmatrix} a_{00} \\ a_{10} \\ a_{01} \\ a_{20} \\ a_{11} \\ a_{02} \\ \vdots \\ a_{0N} \end{bmatrix}, \qquad f = \begin{bmatrix} f_{00} \\ f_{01} \\ f_{02} \\ \vdots \\ f_{0,w-1} \\ f_{10} \\ \vdots \\ f_{w-1,w-1} \end{bmatrix}$$

where f_{ij} denotes $f(x_i,y_j)$.

Let us call $\boldsymbol{f}$ a vectorized form of data within a window. If $\boldsymbol{V}^T\boldsymbol{V}$ is not singular, equation (3) has the unique solution $\boldsymbol{a} = (\boldsymbol{V}^T\boldsymbol{V})^{-1}\boldsymbol{V}^T\boldsymbol{f}$. It can be proven that $w \geq N+1$ guarantees $\boldsymbol{V}^T\boldsymbol{V}$ to be nonsingular.

The window is a local concept. Hence, without the loss of generality, we can specify $x_i = y_i = -(w-1)/2 + i$ for $i = 0, 1,$ line. Assume that a_{ij} is the n^{th} element of $\boldsymbol{a}$. Let $\boldsymbol{W}$ be $(\boldsymbol{V}^T\boldsymbol{V})^{-1}\boldsymbol{V}^T$ whose nm^{th} element is w_{nm}, and $(\boldsymbol{W})_n$ denote the row vector of $\boldsymbol{W}$ corresponding to the element a_{ij} of $\boldsymbol{a}$. Then, the coefficient a_{ij} becomes

$$a_{ij} = (\boldsymbol{W})_n \cdot \boldsymbol{f} = \sum_{m=0}^{(N+1)(N+2)/2} w_{nm} f_m \tag{4}$$

where f_m denotes the m^{th} element of $\boldsymbol{f}$. Consequently, the coefficients of the polynomial for an entire surface data is computed by using a pre-computed separable convolution operator.

Now, we have the polynomial function $\hat{f}(x,y)$ which best fits the wxw local surface patch as follows.

$$\begin{aligned} \hat{f}(x,y) &= \sum_{i,j=0}^{N} (\boldsymbol{W})_n x_i y_j \cdot \boldsymbol{f} \\ &= (\boldsymbol{b}\boldsymbol{W}) \cdot \boldsymbol{f} \end{aligned} \tag{5}$$

where $\boldsymbol{b} = \boldsymbol{b}(x,y) = [\ 1, x, y, x^2, xy, y^2, x^3, \ldots, y^N]$. Let us call the $1\text{x}(N+1)(N+2)/2$ row vector $\boldsymbol{b}$ a basis vector. The directional derivatives can be easily obtained from the equation (5) such that

$$\frac{\partial^{n+m}\hat{f}(x,y)}{\partial^n x \partial^m y} = \left\{ \frac{\partial^{n+m}\boldsymbol{b}(x,y)}{\partial^n x \partial^m y} \boldsymbol{W} \right\} \cdot \boldsymbol{f} = (\boldsymbol{W}^{nm})^T \cdot \boldsymbol{f} \tag{6}$$

$\boldsymbol{W}^{nm}$ is a vectorized form of a window. Therefore, partial derivatives of the depth can be computed by using pre-obtained windows.

However, objects do not usually have entirely smooth surfaces. The mean square error obtained from equation (1) increases in the neighborhood of a discontinuity. Therefore, the mean square error should not be taken at the points of discontinuity. For this reason, we do not require the point of interest to be at the center of the window. We select the best window $\boldsymbol{W}^{nm}(x-u,y-v)$, where $0 \leq u,v \leq w$, which minimizes the following index.

$$I(x,y) = \underset{0 \leq u,v \leq w}{\text{Min}} \left\{ \sum_{(s,t) \in W(x-u,y-v)} \varepsilon^2(s,t) + \lambda |\varepsilon(x,y)| \right\} \tag{7}$$

where $\varepsilon(x,y) = f(x,y) - \hat{f}(x,y)$ and λ is a weighting factor for the point of interest. It is easy to compute that

$$\varepsilon(x,y) = (\boldsymbol{I}_{xy} - \boldsymbol{b}\boldsymbol{W}) \cdot \boldsymbol{f} \tag{8}$$

where $\boldsymbol{I}_{xy}$ denotes a $1\text{x}(N+1)(N+2)/2$ row vector whose element corresponding to the position (x,y) has a value 1 and all the other elements are 0. Moreover,

$$\sum_{x,y} \varepsilon^2(x,y) = \boldsymbol{f}^T(\boldsymbol{A} + \boldsymbol{I} - \boldsymbol{B})\boldsymbol{f} \tag{9}$$

where

$$\boldsymbol{A} = \boldsymbol{W}^T \left(\sum_{x,y} \boldsymbol{b}^T \boldsymbol{b} \right) \boldsymbol{W}$$

$$\boldsymbol{B} = \left(\sum_{x,y} \boldsymbol{I}_{xy}^T \boldsymbol{b} \right) \boldsymbol{W} + \boldsymbol{W}^T \left(\sum_{x,y} \boldsymbol{b}^T \boldsymbol{I}_{xy} \right)$$

All the above window functions can be prepared initially by off-line computation.

3. SIMULATION

In this section we describe some of the simulation results using various real range image data. The program is written in C and run on a MicroVAX computer under the ULTRIX operating system.

For the realistic simulation, we compute and evaluate the Gaussian curvature K using the partial derivatives obtained from our approach. Since Gaussian curvature K at a surface point is view-point independent, it is one of the most frequently used features in 3D image processing areas. K has some advantages in image processing. First of all, Gaussian

curvature is invariant to arbitrary transformations of (u,v) parameters as long as the Jacobian of (u,v) transformation is nonzero. Moreover, Gaussian curvature is invariant to arbitrary translations and rotations of a surface. Gaussian curvature is isometric invariant of a surface.

The Gaussian curvature K is the product of principal curvatures. Principal curvatures represent the normal curvatures k_n of a surface which takes maximum and minimum values with respect to $\dot{\boldsymbol{u}}$ where $\boldsymbol{u} = [u,v]^T$, and $\dot{\boldsymbol{u}} = d\boldsymbol{u} / ds$. u and v are the parameters of a surface, and s represents a path length. The dimension of curvature is an inverse of the *radius of curvature* ρ. The radius of curvature becomes infinite at points of inflection. A higher curvaceous surface has a smaller radius of curvature, which means larger values of K.

K are obtained such that

$$K = k_1 k_2 = \frac{LN - M^2}{EG - F^2} \tag{10}$$

We set $E = \boldsymbol{x}_u \cdot \boldsymbol{x}_u$, $F = \boldsymbol{x}_u \cdot \boldsymbol{x}_v$, $G = \boldsymbol{x}_v \cdot \boldsymbol{x}_v$, $L = \boldsymbol{x}_{uu} \cdot \boldsymbol{n}$, $M = \boldsymbol{x}_{uv} \cdot \boldsymbol{n}$, and $N = \boldsymbol{x}_{vv} \cdot \boldsymbol{n}$. Also, $\boldsymbol{x}_{uv} = \partial^2 \boldsymbol{x} / \partial u \partial v$, and $\boldsymbol{x}_{uv} = \boldsymbol{x}_{vu}$.

Any curved object can be described by the combination of spheres. A sphere is simple to express, and the simulation dealing with a sphere is easy to analyze its results. For this reason, we use spheres with different sizes to examine the performance of our approach. Five spheres are considered with different radiuses such as 30, 50, 70, 100, and 150 pixels. We also assume that the images are perturbed by noise. Gaussian noise is considered with a variance of 2%, 4%, 6%, 8%, and 10% where % denotes the percentage with respect to the resolution of a pixel.

With the presence of noise, the best results are obtained with the 7x7 window. A 7x7 window was also suggested by Besl and Jain[2] for a contaminated image. Figures 1-4 illustrate the histogram of curvatures. These histograms are obtained using 7x7 windows, where the horizontal axis represents the percentage of ratios between the correct curvature values and the curvatures obtained from image points. The left vertical axis denotes the number of pixels obtained from an image without noise. The right vertical axis denotes the same number as the left vertical axis but for an image with noise. As we can see from the figures, correct curvature values are observed at a considerable amount of pixels. Since a small amount of noise introduced to a smaller sphere causes less effect than the noise introduced to a larger sphere, we can see from the figures that the curvature corresponding to a smaller radius with smaller noise is more clustered near the actual value than the curvature for a larger radius with larger noise.

4. CONCLUSIONS

There are computational difficulties in getting accurate partial derivatives directly from a range image since an image is a discrete version of continuous data contaminated with some noise. In general, partial derivatives for 3D objects are computed using the techniques developed for 2D image processing. The literature survey shows that there is no known exact technique for the computation of partial derivatives from a range image.

In this paper, we develop a general window function to compute partial derivatives based on the least square surface fitting method. A dynamic selective surface fitting method is introduced to make the window less sensitive to noise. According to our approach, any degree of partial derivative can be obtained by a simple convolution between an image and window functions.

REFERENCES

[1] G. J. Agin and T. O. Binford, "Computer Description of Curved Objects," *IEEE Trans. on Computer*, pp. 439-449, Apr., 1976.

[2] P. J. Besl and R. C. Jain, "Three Dimensional Object Recognition," *ACM Computing Survey* 17, No. 1,

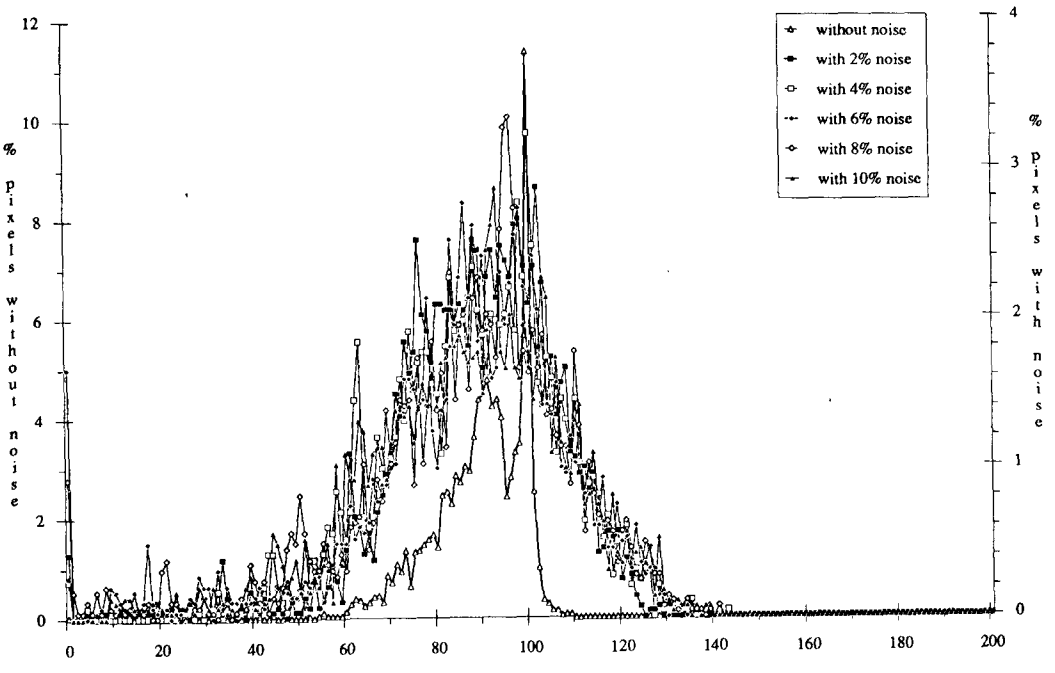

Fig. 1. *K* Histogram of an Image with a Sphere of Radius 30

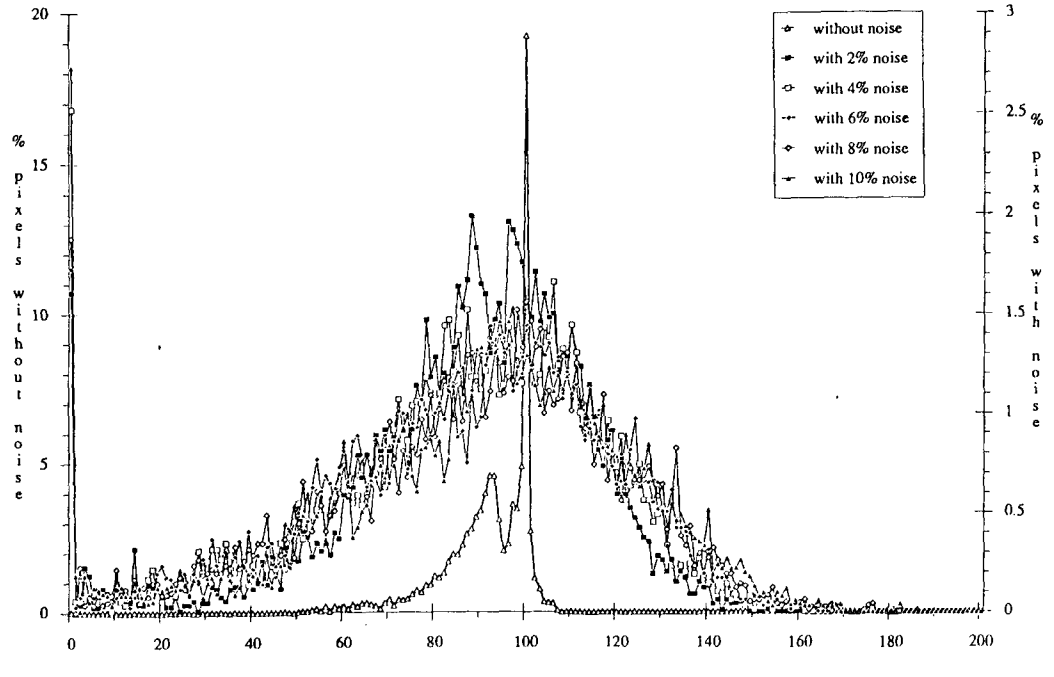

Fig. 2. *K* Histogram of an Image with a Sphere of Radius 50

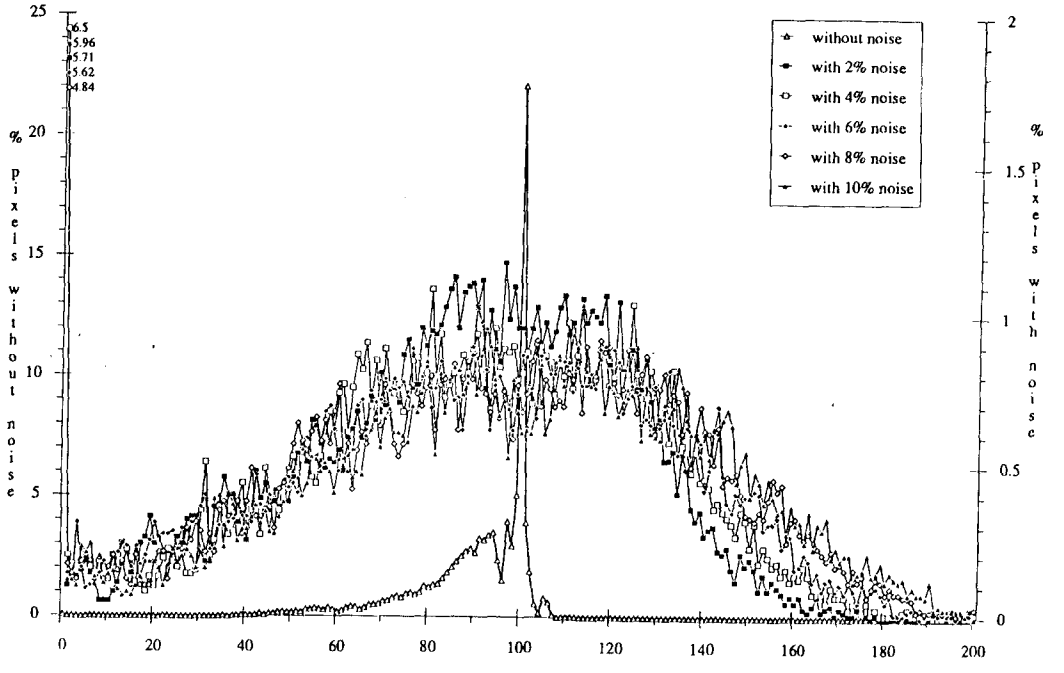

Fig. 3. K Histogram of an Image with a Sphere of Radius 70

pp.75-145, 1985.

[3] P. J. Besl and R. C. Jain, " Segmentation through Variable-order Surface Fitting," *IEEE Trans. on Pattern Anal. Machine Intell.*, Vol. 10, pp. 167-192, March 1988.

[4] P. J. Besl and R. C. Jain, "Segmentation through Symbolic Surface Description," Proc. *IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 77-85, Miami Beach, FL, Jun., 1986.

[5] M. Brady, J. Ponce, A. Yuille, and H. Asada, "Describing Surfaces," Proc. *2nd Int. Conf. on Robotics Research*, pp. 5-16, 1985.

[6] A. I. Borisenko and I. E. Tarapov, *Vector and Tensor Analysis with Application*, New York: Dover, 1968.

[7] I. Faux and M. Pratt, *Computational Geometry for Design and Manufacture*, Ellis Horwood Ltd, 1979.

[8] O. D. Fauguras and M. Herbert, "The Representation, Recognition, and Locating of 3-D Object," *Int. J. Robotic Research*, Vol.5, No.3, pp. 27-52, Fall, 1986.

[9] R. M. Haralick, "Digital Step Edges from Zero Crossing of Second Directional Derivatives," *IEEE Trans. on Pattern Anal. Machine Intell.*, Vol. 6, 1, pp. 58-68.

[10] T. Fan, G. Medioni, and R. Nevatia, "Segmented Description of 3-D Surfaces," *IEEE Journal of Robotics and Automation*, Vol. RA-3, No. 6, pp. 527-538, Dec., 1987.

[11] R. Nevita and T. O. Binford, "Description and Recognition of Curved Objects," *Artificial Intelligence*, Vol. 8, pp. 77-98, 1977.