# Precision shape modeling by z-map model

Jung-Whan Park[1], Yun-Chan Chung[2], and Byoung-Kyn Choi[3]

[1] School of Mechanical Engineering, Yeungnam University, Gyoungsan, Korea
[2] Cubic Technology Research Center, Seoul, Korea
[3] Department of I.E., Korea Advanced Institute of Science & Technology, Daejon, Korea

## ABSTRACT

The Z-map is a special form of discrete non-parametric representation in which the height values at grid points on the xy-plane are stored as a 2D array z[i,j]. While the z-map is the simplest form of representing sculptured surfaces and is the most versatile scheme for modeling non-parametric objects, its practical application in industry (eg, tool-path generation) has aroused much controversy over its weaknesses, namely its inaccuracy, singularity (eg, vertical wall), and some excessive storage needs. Much research on the application of the z-map can be found in various articles, however, research on the systematic analysis of sculptured surface shape representation via the z-map model is rather rare. Presented in this paper are the following: shape modeling power of the simple z-map model, exact (within tolerance) z-map representation of sculptured surfaces which have some feature-shapes such as vertical-walls and real sharp-edges by adopting some complementary z-map models, and some application examples.

**Keywords:** shape modeling, z-map, CZ-map, EZ-map

## 1. Introduction

In general, the surface model for CAD/CAM is categorized into 'parametric' and 'non-parametric' surfaces. The NURBS (Non-Uniform Rational B-Spline) model is one of the common mathematical forms for the parametric surfaces, while the non-parametric surface models such as polyhedral facet[1,2], z-map[3,4], and dexel[5,6] are widely used. Each mathematical form has its own strengths and weaknesses depending on specific applications[7]. The parametric surface is a very efficient tool for use in various areas (ie, design, drawing, etc), but in some specific geometric computations (eg, Cartesian tool-path generation, etc) the non-parametric surface is a more effective one[8].

The non-parametric surface is generally defined on a two-dimensional xy domain, and can be represented by an explicit form as follows:

$$z = f(x,y) \qquad (1)$$

The z-map model is a special form of the non-parametric surface in which the height values at grid points on the xy-plane are stored as a 2D array z[i,j] (Eq. 2 & Fig. 1).

$$Z[i,j] \text{ with } x=x_o+d_x \cdot i \text{ and } y=y_o+d_y \cdot j. \qquad (2)$$

*where, $d_x$, $d_y$ : grid-interval along x and y,*
*$x_o$, $y_o$ : origin of the xy domain.*

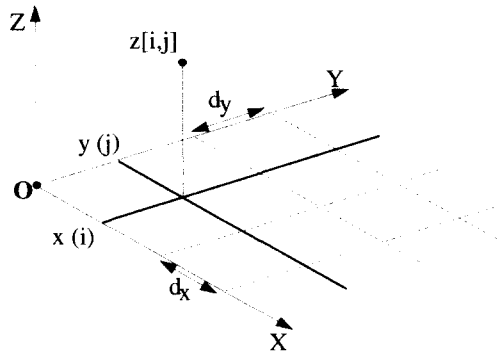



Fig. 1 Z-map model

Considering the height values (z values of a surface) of a z-map at any (x,y) position, we can directly obtain the height value from Z[i,j] if the (x,y) position falls on a pre-defined grid-point (i,j) which comes from the equation (3). The height value should be computed, however, by some appropriate interpolation method if the (x,y) position is located outside any grid-point (ie, within a rectangular-cell or on a grid-line).

$$i = (x-x_o)/d_x \quad \text{and} \quad j = (y-y_o)/d_y \qquad (3)$$

The z-map model is widely used for NC tool-path verification[4,6,9-12], CAPP (computer-aided process planning)[13,14], and NC tool-path generation[4,15-18]. It can be said that the z-map model itself is conceptually similar to the z-buffer[19] in the computer graphics area, or depth-map[21] from a range finder[20] in robot vision.

It is generally accepted that, mainly due to its simple data structure, a z-map is a good representation scheme in terms of modification of surface[2] (eg, offsetting, blending) and robustness of computation. On the other hand, it should be pointed out that de facto infeasible memory requirement and computation time (in case of a high accuracy) may be one of its drawbacks. For example, roughly 1.6GB of memory is required if one needs 0.05 unit of precision (ie, grid-interval = 0.05) for the xy-domain of 1000× 1000 units (ie, $4\times 10^8$ grid-points total, 4 bytes for each grid-point), which leads to the conclusion that a pure z-map is not a practical solution for modeling a very accurate shape with a large xy-domain.

There are practical restrictions on representing some special feature shapes (eg, vertical walls) of the die part-surface with accuracy, while it might be possible for a z-map to model a smooth surface to a certain degree of precision (see section 2). The following are some feature shapes that are difficult or impossible for a simple z-map to handle with ease (see Fig. 2):

- undercut-shape,
- vertical-wall,
- sharp-edge.

The 'undercut-shape' can be modeled by another extended form of the z-map model (eg, dexel), which we do not consider in the paper. In addition, the 'vertical-wall' feature type contains a nearly vertical (very steep) wall area as well as a completely vertical one.

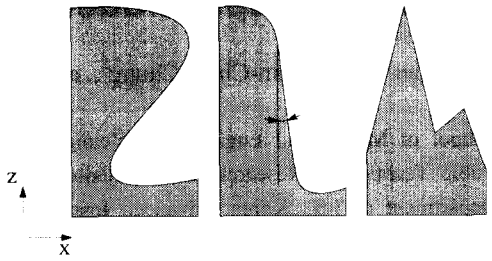



Fig. 2 Arduous feature shapes in the z-map model

In the paper, representation of surfaces by the simple z-map form and representation of feature shapes on die part-surfaces by some extended z-map forms will be presented. Section 2 deals with the conversion of compound parametric surfaces to the z-map model, and a reasonable z-value interpolation scheme on a z-map. The following section suggests some extended z-map models based on the simple z-map, which can represent 'vertical-wall' and 'sharp-edge' features with accuracy. Also, some illustrative examples are shown in section 4.

## 2. Shape representation by a z-map

The conversion of a parametric surface to a z-map Z[i,j] is performed by computing the height value at each grid-point (i,j). Once we have the z-map on hand, one of the most important operations becomes retrieval of the shape information, that is, a z-value computation of the surface $z=f(x,y)$ at any given (x,y) position. It is a very simple job to get a z-value at an (x,y) position that corresponds to a grid-point (i,j), which should be Z[i,j]. On the other hand, if an (x,y) position does not fall onto any pre-defined grid-point, an appropriate interpolation method should be used to get the z-value, where we have the surrounding grid-points' z-values. Supposing that the overall surface shape is smooth and has neither vertical-walls nor sharp-edges, we can observe that there is little difference between the interpolated z-value and its accurate one. This restricted case makes it possible for a simple z-map to represent the surface shape with accuracy to a certain degree.

### 2.1 Conversion of a parametric surface to a z-map

The conversion of a parametric surface to a z-map form - i.e., the computation of z-values at the pre-defined grid-points (see Fig. 3) - can be performed by use of differential geometry called 2D Jacobian Inversion[3] as well as by faceting of the input surface. The former method, in general, can converge to a unique solution as long as the surface is smooth, but in some cases it may fail or take too much time to arrive at a solution. On the other hand, the facet (eg, triangular polyhedra) of the input surface can be more robust (stable), which enables one to estimate the z-value at an (x,y) position directly from the corresponding triangle. However, the facet is an approximated form to the input surface (within a given tolerance), leading to local flatness or unsmooth transitions. Therefore, the 2D Jacobian inversion is tried first, and in case it fails, facets of the input surface are used to construct the z-map.

In general, the part-surface of a stamping-die is composed of as many as hundreds or thousands of faces (trimmed parametric surfaces). Given the xy-domain and grid-points of a z-map for the part-surface, each face of the compound surfaces is positioned on the domain and the z-value (at each grid-point) corresponding to the face is computed and stored at Z[i,j] by the method described earlier. In this paper the z-map converted from the input part-surface is named 'Master-model z-map'.

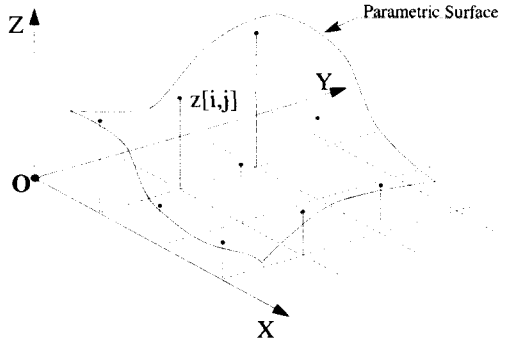



Fig. 3 Z-value computation at pre-defined grid-points

### 2.2 Interrogation of a z-value at any domain position

As mentioned before, for any domain position (x,y) which happens to fall onto a z-map grid-point (i,j) we can directly get its z-value from Z[i,j] by equations (2) and (3), while some interpolating scheme should be used for any other (x,y) position located out of a grid-point (ie, out of a small circle centered at the grid-point with a small radius). The z-value at (x,y) outside the overall z-map domain is undefined.

Now that the information on hand is four z-values at the grid-points surrounding the domain position (x,y)- like g2,g3,g6,g7 in Fig. 4, rectangular bilinear interpolation may be one simple way to evaluate z-values. It may, however, result in an erroneous evaluation in the case of a non-planar surface area. Considering that any 'local' shape of CAD surfaces can be well approximated by a cubic equation, we suggest an interpolating method which utilizes the non-parametric cubic Bezier curve form[3,22].

Suppose that a domain position P=(x,y) is located in a rectangular cell ($x_i \le x \le x_{i+1}$, $y_j \le y \le y_{j+1}$) of a z-map Z[i,j] ($0 \le i < m$, $0 \le j < n$) as shown in Fig. 4, whose relative position $(u,v)$ within the cell is obtained from the equation (4).

$$u = (x-x_i)/(x_{i+1}-x_i)\ ;\ v = (y-y_j)/(y_{j+1}-y_j)\ ;\ 0 \le u,v \le 1 \quad (4)$$

*where,* $x_i \le x \le x_{i+1}$ *and* $y_j \le y \le y_{j+1}$.

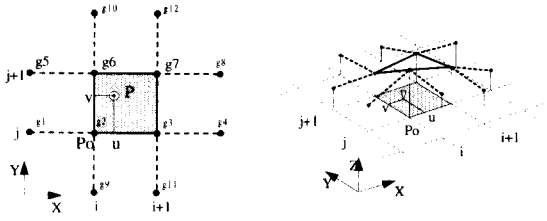



Fig. 4 Z-value interpolation for non-grid point

Let us define $\mathbf{W}_1$ and $\mathbf{W}_2$ from the x-directional grid-points ($g_1,g_2,g_3,g_4$) and ($g_5,g_6,g_7,g_8$) as follows (see Fig. 4), where each component $z_i$ comes from the z-value of a grid-point $g_i$:

$$\mathbf{W}_1 = (z_1,z_2,z_3,z_4) \text{ and } \mathbf{W}_2 = (z_5,z_6,z_7,z_8).$$

With $\mathbf{W}_1$ and $\mathbf{W}_2$ at hand, we can define the control points of two non-parametric cubic Bezier curves: $\{\mathbf{V}_k = (v_0,v_1,v_2,v_3);\ k=1,2\}$. Eq. (5) evaluates the control points for k = 1 from $\mathbf{W}_1$, those for k = 2 can similarly be obtained.

$v_0 = z_2, \quad v_3 = z_3,$

$v_1 = v_0 + s \cdot d_x/3, \quad v_2 = v_3 - t \cdot d_x/3,$ (5)

*where, $d_x$ = x-directional grid-interval,*

*$s$ = slope of the circular tangent at $g_2$ (from $z_1$, $z_2$, $z_3$),*

*$t$ = slope of the circular tangent at $g_3$ (from $z_2$, $z_3$, $z_4$).*

Each non-parametric cubic Bezier curve is defined by Eq. (6) from the control points ($v_0, v_1, v_2, v_3$), and the height value corresponding to the relative position ($u$) in Eq. (4) can be computed as $\omega$. Let $\omega_1$ and $\omega_2$ be two height values at $u$ from $\mathbf{V}_1$ and $\mathbf{V}_2$, respectively.

$$\omega(u) = \sum_{i=0}^{3} v_i B_i^3(u), \quad 0 \le u \le 1 \tag{6}$$

*where, $v_i$ = Bezier curve's control point,*

*$B_i^3(u)$ = Cubic Bernstein polynomial.*

Now, for the y-directional grid-points ($g_9, g_2, g_6, g_{10}$) and ($g_{11}, g_3, g_7, g_{12}$), the height values at $g_i$'s are defined as $\mathbf{W}_3$ and $\mathbf{W}_4$ as follows:

$\mathbf{W}_3 = (z_9, z_2, z_6, z_{10})$ and $\mathbf{W}_4 = (z_{11}, z_3, z_7, z_{12})$.

From $\omega_1, \omega_2$ and $\mathbf{W}_3$, we define another set of control points $\mathbf{V}_3 = (v_0, v_1, v_2, v_3)$ of a non-parametric cubic Bezier curve by Eq. (7). Also $\mathbf{V}_4$ can be obtained similarly.

$v_0 = \omega_1, \quad v_3 = \omega_2,$

$v_1 = v_0 + s \cdot d_y/3, \quad v_2 = v_3 - t \cdot d_y/3,$ (7)

*where, $d_y$ = y-directional grid-interval,*

*$s$ = slope of the circular tangent at $g_2$ (from $z_9$, $z_2$, $z_6$),*

*$t$ = slope of the circular tangent at $g_6$ (from $z_2$, $z_6$, $z_{10}$).*

$\mathbf{V}_3$ and $\mathbf{V}_4$ define two non-parametric Bezier curves (Eq. 6), and height values $\omega_3$ and $\omega_4$ at the y-directional relative position ($v$ in Eq. 4) are calculated by Eq. (6). The height value $Z_P$ at the domain position P (Fig. 4) is finally computed by the following linearly blended function:

$Z_P = (1 - u) \cdot \omega_3 + u \cdot \omega_4.$

When P happens to be in a rectangular cell which is located on the boundary of the z-map domain, those 12 grid-points (as shown in Fig. 4) cannot be defined, but is resolved by adding 'dummy' grid-points to construct a complete set. As an example, let z-map indices of Po (see Fig. 4) be ($i = 0$, $0 < j < n$). Then we define two additional grid-points at (i-1, j) and (i-1, j+1), where the height values of those pseudo grid-points are obtained by linear extrapolation of Z[i,j], Z[i+1,j] and Z[i,j+1], Z[i+1,j+1], respectively.

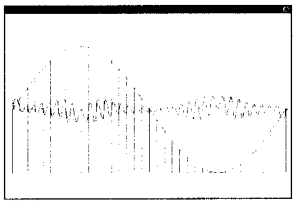
Table 1 and Fig. 5 show some illustrative examples, in which the suggested interpolating scheme is compared with the ordinary bilinear interpolation - the difference between interpolated z-values from z-map and 'accurate' z-values of the analytic surface.

Table 1. Illustrative results of z-value interpolation for a domain-point

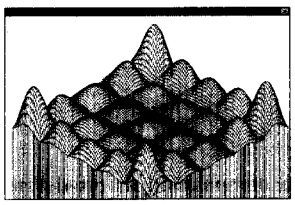
| Surface equation | grid-interval ($d_x$=$d_y$) | Max. deviation (z-value) | |
|---|---|---|---|
| | | A* | B** |
| $x^2+y^2+z^2=100^2,\ z>70$ | 1 | $\pm 3.0\times10^{-6}$ | $-3.7\times10^{-3}$ |
| $z = 100 \cdot \sin(\pi x/180)$ | 1 | $\pm 1.2\times10^{-5}$ | $\pm 3.8\times10^{-3}$ |
| $z = 1000 \cdot \sin(\pi x/180)$ | 1 | $\pm 2.1\times10^{-4}$ | $\pm 3.8\times10^{-2}$ |

* Non-parametric Bezier interpolation method

** Bi-linear interpolation method



(a) Surface: $z = 100*\sin(\pi x/180)$



(b) Surface: $x^2+y^2+z^2 = 100^2$, $z > 70$

Fig. 5 Magnified error pattern for z-value interpolatio

According to the test, and considering that ordinar machining tolerance (accuracy) falls in the range of 10

to $10^{-2}$mm, it is possible to represent a smooth shape ($G^1$-continuous, without vertical-wall nor sharp-edge) by a simple z-map whose grid-interval is less than 1mm (eg, 0.7mm).

## 3. Shape representation by extended z-map models

Due to some special features such as sharp-edges (SE) or vertical-walls (VW) on part-surfaces of mold & dies, the simple z-map may not be an appropriate form (see Fig. 6). We suggest two extra z-map models which can represent SE and VW feature shapes within a given error value: CZ-map (Core Z-map) and EZ-map (Extended Z-map).

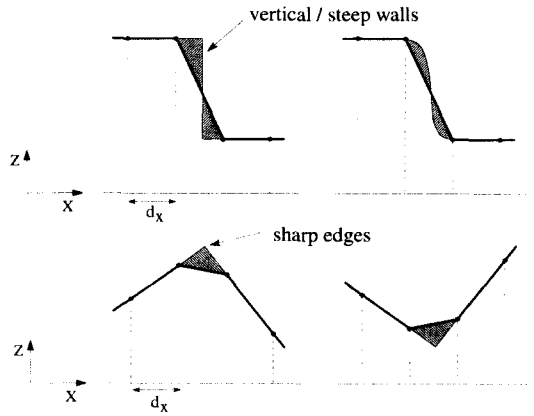



Fig. 6 Wrongly-defined feature shapes in the z-map model

### 3.1 CZ-map model

This model is conceptually similar to a real 'core' part in molding die manufacturing. The overall shape is modeled by a simple z-map with practical (ie, 0.5~1mm) grid-intervals (base z-map), and additional 'core' z-maps (CZ-maps) with smaller grid-intervals ($\delta_x$ and $\delta_y$) are inserted at the local areas of SE or VW feature shapes (see Fig. 7). Each CZ-map basically has the same data structure as a z-map, which enables the CZ-map to inherit the merits of the z-map model.

The two-dimensional domain of a CZ-map is defined from each line segment of the feature-curves on a base z-map (see Fig. 7-a). Also, we try to minimize the number of CZ-maps on a base z-map by defining the local coordinate frame of each CZ-map - the rotation-angle ($\theta$) and the origin's position (see Fig. 7-b). There are two types of feature-curves: VW (Vertical-Wall) curves which are traced out on the base z-map, and SE (Sharp-Edge) curves found from boundary edges of the input parametric compound surfaces. Construction of a CZ-map can be performed by the method described earlier (§ 2.1).

The grid-intervals of a CZ-map are selected as follows;

$\delta_y$ = given tolerance value,

$\delta_x = k \cdot \delta_y$ ($k \geq 1$),

*where* $\delta_x < d_x$ *and* $\delta_x < d_y$,

where the direction orthogonal to the feature-curve segment has a smaller grid-interval ($\delta_y$) since the feature shapes should be represented mainly by the CZ-maps' y-directional grid-points. We selected the value of $k$ such that the y-directional deviation of a feature-curve traveling through a CZ-map domain should be equal to or less than $\delta_y$ when the x-directional one is $\delta_x$. Let $\theta_x$ be $\tan\theta_x = \delta_y/\delta_x = 1/k$, where $\theta_x$ is the maximum angle between the feature-curve and the local x-axis of a CZ-map domain. Setting $\theta_x$ to be a certain value, we can get the value of $k$ from the above equation. In the case of $\theta_x = \pm 20°$, we get $k = 1/\tan 20° \approx 3$. Due to the different grid-intervals along the local x- and y-axis ($\delta_x$ and $\delta_y$), the additional memory requirement may be minimized.

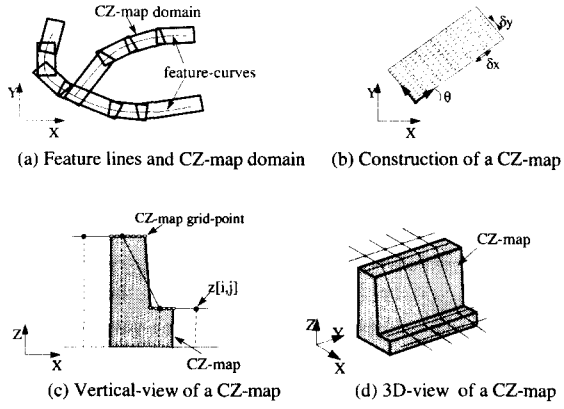
The following procedure describes the construction of a set of CZ-maps based on a Master-model (base) z-map Z[i,j].

**Procedure Construct_CZmap (S , Z[i,j] , $\varepsilon_r$ ⇒ CZ-map) ;**

1. Input: Trimmed parametric surfaces S, Z-map Z[i,j] ;
2. Trace 2-D VW feature-curves from Z-map → {$L_i$; i=1,...,m} ;
3. Trace 2-D SE feature-curves from S→{$L_i$; i=1,...,m} ;
4. For i = 1 to m, do following:
   1) Define a series of CZ-map domains for $L_i$ → {$D_{ij}$; j=1,...,p} ;
   2) For j = 1 to p, compute z-values for the grid-points in each domain $D_{ij}$ → $CZ_k$ ;
5. Return a set of CZ-map {$CZ_k$} ;

Since a CZ-map on a base z-map should be more

precisely defined, evaluation of the z-value at a domain position (x,y) gives a better result on a CZ-map ($CZ_k$) than on its base z-map as long as (x,y) is located within the domain of $CZ_k$. The evaluation is performed by the method described earlier (§ 2).




(a) Feature lines and CZ-map domain (b) Construction of a CZ-map

(c) Vertical-view of a CZ-map (d) 3D-view of a CZ-map

Fig. 7 CZ-map (Core Z-map) model

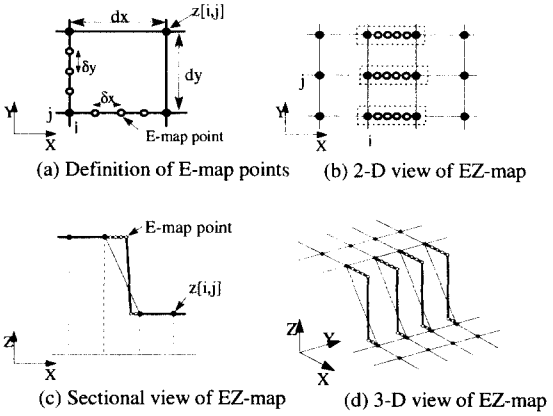
### 3.2 EZ-map model

The EZ-map (Extended Z-map) model, where each designated grid-edge (ie, a segment between two consecutive grid-points) of a base z-map has additional grid-points (e-map points), also can locally enhance the modeling accuracy (see Fig. 8). Those grid-edges which should have e-map points can be selected by tracing out the feature-curves as in the previous section and defining the feature areas along the feature-curves. The e-map points on a grid-edge are uniformly located (see Fig. 8-a & b), each of which contains the corresponding z-value of the input part-surface (see Fig. 8-c & d).

The two-dimensional distance between e-map points (ie, $\delta_x$ and $\delta_y$) can be selected by considering a given machining tolerance which, in general, should be in the range of 0.01~0.05mm for mold & die manufacturing. Construction of an EZ-map can be described as follows:

**Procedure Construct_EZmap (S, Z[i,j], $\varepsilon_\tau$ ⇒ EZ-map) ;**

1. Input: Trimmed parametric surfaces S, Z-map Z[i,j];
2. Trace 2-D SE feature-curves from S → $\{L_p\}$;
3. For all ij-th horizontal & vertical grid-edges ($E_x$ & $E_y$), do following:
   1) If (slope of $E_x \geq \varepsilon_\tau$) or ($E_x$ crosses any $L_p$), define e-map points on $E_x$ → $\{EZ_x[i,j][k]; k=0,..,n\}$ ;
   2) If (slope of $E_y \geq \varepsilon_\tau$) or ($E_y$ crosses any $L_p$), define e-map points on $E_y$ → $\{EZ_y[i,j][k]; k=0,..,n\}$ ;
4. Compute all z-values of e-map points from S ;
5. Return EZ-map ;




(a) Definition of E-map points (b) 2-D view of EZ-map

(c) Sectional view of EZ-map (d) 3-D view of EZ-map

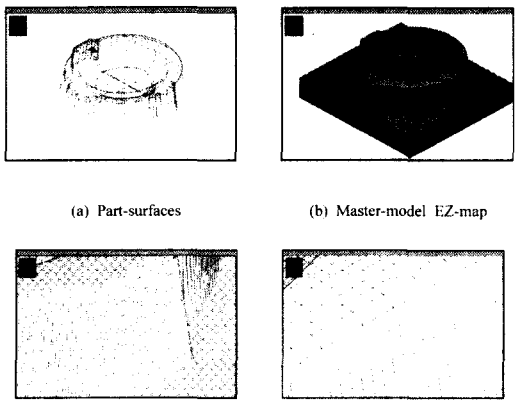
Fig. 8 EZ-map (Extended Z-map) model

Since not all the grid-edges have e-map points, it can be said that the additional memory requirement of a EZ-map is less than that of a set of CZ-maps for a same part-surface (from a few tests).

To evaluate the z-value at a domain position P=(x,y) the simplest way may be to combine the four grid-edge curves (point sequences) surrounding P into a cubic surface patch (eg, bi-cubic Coons patch) with a rectangular topology. In some cases, however, a one-patch model may give an incorrect result when any of the grid-edge curves is not $G^1$-continuous (ie, sharply bent) or has an abrupt change in curvature values locally. In this case it is necessary for the patch to be subdivided for more accurate interrogation.

## 4. Illustrative examples

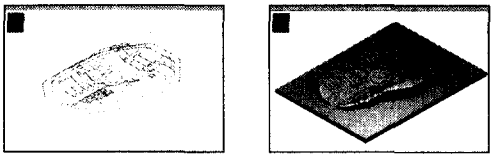
This section describes the EZ-map construction for two die-surface models. Fig. 9 shows the EZ-map construction for a partial connecting-rod, and a passenger car fuel tank model case is shown in Fig. 10. Those cases have no SE (sharp-edge) feature shapes, and the summarized results are found in Table 2.



(a) Part-surfaces (b) Master-model EZ-map

(c) Master-model EZ-map (magnified) (d) E-map points on z-map grid-edges

Fig. 9 Application example: Connecting-rod(partial)



(a) Part-surfaces (b) Master-model EZ-map

Fig. 10 Application example: Passenger car fuel tank

Table 2. Summary of EZ-map construction for two example models

| Model | | Connecting-rod | Fuel-tank |
|---|---|---|---|
| XY-dimension (mm) | | 110 × 120 | 1190 × 890 |
| Grid-interval (mm) | Z-map | 0.4 | 0.8 |
| | E-map | 0.02 | 0.04 |
| Memory (KB) | Z-map | 330 | 6,620 |
| | E-map | 1,120 | 8,000 |
| | EZ-map | 1,450 | 14,620 |
| E-map ratio (%) | | 17.8 | 6.3 |

The grid-intervals ($d_x$=$d_y$) of the two Master-model (base) z-maps are 0.4 and 0.8, respectively. E-map points are distributed on each grid-edge at regular intervals; 0.02 and 0.04, respectively for the two cases (ie, the number of e-map points on each grid-edge is 19).

Considering the passenger car fuel tank model, we need approximately 2.2 times the memory for constructing the EZ-map (14.62MB) as compared with the base z-map (6.62MB) (see Table 2). It may be meaningful to note that roughly 2.6GB of memory is needed to construct a simple z-map whose grid-interval is 0.04 (e-map points' interval).

The 'e-map ratio' in Table 2 means the number of grid-edges to that of all grid-edges of the base z-map. It can be noted that the connecting-rod model has a higher value than the fuel tank model, since the former has more VW (vertical-wall) feature area than the latter does.

## 5. Conclusions and discussions

We analyzed the modeling power of the simple z-map model, and suggested two extended z-map models which can more precisely represent feature-shapes such as VW (vertical-wall) or SE (sharp-edge). It may be, in some cases, practically infeasible for a simple z-map to represent those feature shapes with accuracy (ie, very small grid-intervals leading to huge memory needs).

Z-MASTER[22], a commercial CAM system which has been developed by the authors, adopted the EZ-map model to generate finish-cut NC tool-paths for stamping-die machining. In terms of memory and accuracy requirements, it is practically feasible to use those extended z-map models for mold & die manufacturing. Actually, due to some strengths, the z-map model is widely used in many areas[8] such as the following:

- NC cutting simulation & verification,
- Rendering,
- Virtual prototyping and styling,
- NC tool-path generation,
- CAPP, computer vision.

The paper focuses on a practical method of overcoming the accuracy enhancement while maintaining the basic structure of the simple z-map model. Though it is still versatile, the z-map may be the most appropriate mathematical model for NC cutting simulation and verification of a die part-surface. It should be noted, however, that a very accurate modeling scheme is necessary for a part-surface with VW or SE features to be verified (eg, overcut, collision, etc) after the cutting-simulation. As shown in the paper, a simple z-map is not a good choice for such featured shapes, mainly due to extreme memory and computation time. As a

concluding remark, the extended z-map models, while still maintaining the basic data structure of a z-map, can be feasible alternatives to the simple z-map model.

## References


1. Hwang, J. S., "Interference-free tool-path generation in the NC machining of parametric compound surfaces," Computer-Aided Design, Vol. 24, No. 12, pp. 667-676, 1992.
2. Lee, S. X., and Jerard, R. B., "5-axis machining of sculptured surfaces with a flat-end cutter," Computer-Aided Design, Vol. 26, No. 3, pp. 165-178, 1994.
3. Choi, B. K., Surface Modeling for CAD/CAM, Elsevier, New York, 1991.
4. Choi, B. K., Chung, Y. C., Park, J. W., and Kim, D. H., "Unified CAM-system architecture for die and mould machining," Computer-Aided Design, Vol. 26 No. 3, pp. 235-243, 1994.
5. Hook, T. V., "Real time shaded NC milling display," Computer Graphics, Vol. 20 No. 4, pp. 15-20, 1986.
6. Kim, C. B. and Yang, M. Y., "A study on the verification of 5-axis CNC machining," Journal of KSME, Vol. 18 No. 1, pp. 93-100, 1994.
7. Park, J. W., "A study on the development of a CAM system for 5-axis NC machining of die & mold,", Ph.D. Thesis, KAIST, 1995 (In Korean).
8. Choi, B. K., Chung, Y. C., and Park, J. W., "Application and extension of z-map model," Proc. of the Pacific Conference on Computer Graphics and Applications, Seoul, Aug, pp. 221-234, 1995.
9. Andersen, R. O., "Detecting and eliminating collisions in NC machining," Computer-Aided Design, Vol. 10 No. 4, pp. 231-237, 1978.
10. Hsu, P. L., and Yang, W. T., "Realtime 3D simulation of 3-axis milling using isometric projection," Computer-Aided Design, Vol. 25 No. 4, pp. 215-224, 1993.
11. Takata, S. Tsai, M. D., and Inui, M., "A cutting simulation system for machinability evaluation using a workpiece model," Annals of the CIRP, Vol. 38, pp. 417-420, 1989.
12. Chung, Y. C., "Cutting simulation and verification of NC data for unified CAM system for die and mold manufacturing," Ph.D. Thesis, KAIST, 1996 (In Korean).
13. Yoo, W. S., and Choi, B. K., "CAPP for Die Cavity Machining," Proceedings of the IFIP TC5/WG5.3 Eighth Int'l PROLAMAT Conference, Tokyo, Japan, pp. 437-447, 1992.
14. Kim, D. H., "Development of a Loading Schedule System and CAPP for CIM of Molding Dies," Ph.D. Thesis, KAIST, 1997 (In Korean).
15. Takeuchi, Y. Sakamoto, M. Abe, Y., and Orita, R., "Development of a personal CAD/CAM system for mold manufacture based on solid modeling techniques," Annals of the CIRP, Vol. 40, pp. 455-458, 1989.
16. Saito, T., and Takahashi, T., "NC machining with G-buffer method," Computer Graphics, Vol. 25 No. 4, pp. 207-216, 1991.
17. Choi, B. K., Park, J. W. and Jun, C. S., "Cutter-location data optimization in 5-axis surface machining," Computer-Aided Design, Vol. 25 No. 6, pp. 377-386, 1993.
18. Park, J. W., Chung, Y. C., and Choi, B. K., "Pencil curve tracing via virtual digitizing," Machining Impossible Shapes (Olling, G. ed), pp. 279-292, 1999.
19. Foley, J. et al, Computer Graphics: Principles and Practice, 2nd ed., Addison-Wesley, 1990.
20. Faugeras, O., Three-dimensional Computer Vision: a Geometric Viewpoint, The MIT Press, 1993
21. Horn, B. K. P., Robot Vision, The MIT Press, 1986.
22. Farin, G., CAGD: a Practical Gguide. Academic Press, New York, 1997.
23. Z-Master Reference Manual, Cubic Tech, Korea, 1992.