Bounded distance field for assisting interactive layout design of water circuits

Masatomo INUI*, Tomohiro IMAI** and Nobuyuki UMEZU*
*Department of Mechanical Systems Engineering, Ibaraki University
4-12-1 Nakanarusawa, Hitachi, Ibaraki 316-8511, Japan
E-mail: masatomo.inui.ae@vc.ibaraki.or.jp
**Information Systems Department, Koito Manufacturing Co., Ltd
500 Kitawaki, Shimizu-ku, Shizuoka, Shizuoka 424-8764, Japan

Received: 1 February 2019; Revised: 30 March 2019; Accepted: 7 May 2019

Abstract
Manufacturing companies often define various guidelines on the part placement in a mechanical product. In this study, we present a software system to assist in the layout of parts when minimum thickness and clearance conditions between the parts are specified in the guidelines. The idea of a “bounded distance field” is proposed, which is a distance field in a limited region around the boundary surface of a solid object. Gaps generated during parts placement that are too narrow can be detected by checking the distance value at the surface polygon of the object in the field. Based on a placement of new objects or a position change of objects, the distance field in the modeling space must be updated. To reduce the update cost, we introduce the idea of a “lazy evaluation” into the distance computation. In this method, a thickness and clearance evaluation is performed by using individual fields associated with the objects, and the computation of the combined distance field of all objects is postponed for as long as possible. To evaluate the practical application of the bounded distance field and the lazy evaluation of the distance value, an interactive system to assist in the layout design of water circuits in a mold cavity is demonstrated.

Keywords: Sampled distance field, Thickness, Clearance, Layout design, CAD

1. Introduction

Thermal management is a critical issue in the design of mechanical products. In the engine compartment of an automobile, many components, some of which are heat sources, are packed in a limited space. In the layout design of this compartment, specifying appropriate clearances between the components is crucial in order to realize sufficient air flow between them. Injection molding is a popular method for fabricating plastic parts. In this process, the plastic material is injected into a heated barrel, where it is melted and forced into a mold cavity to cool, harden, and form the part. To accelerate the cooling, water circuits are realized in the mold cavity. Proper placement of holes for the water flow is critical to reduce the heat uniformly and efficiently. To assist thermal management, various simulation systems have been developed in order to analyze heat distribution and temperature change. Because most systems use costly finite element or difference methods, substantial computation time is necessary to obtain simulation results. Therefore, using such systems in the interactive design process is impractical.

A designer constructs a three-dimensional (3D) CAD model of an engine compartment by placing components in the compartment space. In the mold design, a designer lays out a water circuit by placing water pipes in a solid model of a mold cavity. To assist designers in this type of modeling task, manufacturing companies define various “guidelines” for creating the layout. Many of these guidelines address the minimum thickness and clearance between the component objects (e.g., the minimum thickness between water pipes and the mold surface in order to realize uniform cooling, and the minimum clearance between the engine components for sufficient air flows). By designing according to the guidelines, the designer can avoid obvious mistakes in parts placement. A designer first positions the component objects based on the guidelines, and then performs precise numerical analysis to confirm the performance.
In this study, we present a software system used to assist in the placement of solid objects when dimensional constraints, particularly minimum thickness and clearance conditions, are imposed on those objects. We consider the following two types of layout problems: the placement of component objects in their external space (see Fig. 1(a)) and those in the internal space of a large object (see Fig. 1(b)). The first corresponds to the layout design of an engine compartment, and the second corresponds to the placement of water pipes inside a mold cavity. By inverting all objects (water pipes and mold cavity), the latter can be converted to an equivalent layout problem of inverted objects in their external space. Therefore, only the placement of objects with minimum-clearance conditions is discussed in the remainder of this paper.

![Fig. 1 Two types of layout problems: (a) in the external space of objects with a minimum-clearance condition, and (b) in the internal space of a large object with a minimum-thickness condition.](image)

To detect efficiently the problematic placement of component objects, as in cases where gaps between objects are too narrow, an interactive system is required that can visualize the distance conditions between objects. Many methods have been developed to evaluate the distance in a 3D space. In these methods, distance evaluation using the distance field has many advantages in CAD and computer-aided manufacturing (CAM) (Frisken and Perry, 2006). An object’s distance field specifies the distance of any point in space to the boundary of that object. In the placement of a component part on an object, clearance between a point \( p \) on the component’s surface and the object can be checked by the value of \( p \) in the distance field of the object. The distance field is often constructed as a sampled volume in which each sample in the space measures the distance from the corresponding point to the object. Point sampling based on the cell structure of the space is usually used. Consider a box-like space containing solid objects. The 3D distance field of the objects is a uniform cell decomposition of the space where at each cubic cell (voxel) in that space, the distance from the center of the voxel to the closest point on the object’s surface is recorded.

To speed up the computation of the sampled distance field, various technologies have been proposed (e.g., adaptive sampling, culling technology using hierarchical bounding boxes, and using the parallel processing capability of a graphics processing unit (GPU)). Although these acceleration methods are effective, constructing the distance field of 3D objects for an interactive operation is difficult. We propose a novel distance field construction method specialized to assist in the interactive layout of component objects with minimum clearance constraints. The idea of a “bounded distance field” is introduced, which is a distance field existing only in a limited region around the boundary surface of an object. Gaps generated during the object positioning process that are too narrow can be visualized by checking the distance value at the surface polygon of the objects in the field and by painting the polygon with a unique color that corresponds to the distance value. Based on a placement of new objects or a position change of the objects, the distance field of the space must be updated. Frequently updating the distance field degrades the interactivity of the system. To reduce the update cost, we introduce the idea of a “lazy evaluation” into the distance field computation. In this method, a clearance evaluation is executed by using individual distance fields associated with the limited objects. The computation of the combined distance field of all objects is postponed for as long as possible. By using these two technologies, we can execute our layout assistance system at a speed capable of interactive operation.

The remainder of this paper is organized as follows. In Section 2, previous studies on thickness and clearance visualization and distance field construction are reviewed. A bounded distance field is defined and its computational method for a polyhedral object is described in Section 3. In Section 4, we describe the detection method for gaps that are too narrow and where the bounded distance field is employed, and give the distance computation for multiple objects. In Section 5, an interactive CAD system used to assist the placement of the water pipe in a mold cavity is...
demonstrated as an application of the bounded distance field. A user of the system can change the position and orientation of the water pipes, and the system can confirm that pipe arrangements satisfy the minimum thickness condition in a second. In Section 6, we provide our conclusion.

2. Related studies

2.1 Thickness and clearance definitions and analysis

The ray and sphere methods are two major methods for evaluating the thickness and clearance in the CAD/CAM of a mechanical product (Sinha, 2007). In the ray method, the thickness at a point \( p \) on a surface is given by using a ray originating from \( p \) in a direction opposite to the local outward normal. The Euclidean distance \( d \) between \( p \) and another point \( q \) corresponds to the thickness, where \( q \) is an intersection point between the ray and the surface immediately opposite the object. In the sphere method, thickness at a point \( p \) on an object’s surface is given by the diameter of the maximum inscribed sphere that contacts the surface at point \( p \).

Subburaj et al. proposed a thickness analysis based on a voxel model and presented three new measures of thickness (Subburaj et al., 2006): “exterior thickness,” which is a modification of the sphere method, “radiographic thickness,” which is a variant of the ray method, and “interior thickness,” which is a type of distance transform explained later in this paper. Lu et al. proposed a thickness analysis based on a distance transform for detecting thicker regions of 3D objects (Lu et al., 1997). Previously, we developed a thickness and clearance visualization system for a solid model (Inui et al., 2015), whereby the thickness of a polyhedron was determined by using the sphere method with a distance field. Most clearance analysis methods developed thus far are designed for specific manufacturing requirements (e.g., potential field and configuration space for robot motion planning (Latombe, 1991) and the accessibility cone for machinability evaluation (Spitz et al., 1999)(Morimoto and Inui, 2007).

2.2 Distance field computation and visualization

Fast algorithms used to construct a sampled distance field have been actively investigated (Jones et al., 2006). Two basic approaches exist. The first is known as distance transform, which is based on the propagation of distance information (Rosenfeld and Pfaltz, 1966)(Cuisenaire, 1999). Voxel on the object boundary are detected and their exact distance values are calculated in the initialization step. The distances are then propagated to the remaining internal or external voxels based on the pre-defined template (Rhodes, 1992)(Danielsson, 1980)(Mullikin, 1992). Zhao proposed a fast sweeping method for solving the eikonal equation in a discrete manner (Zhao, 2004), and Chang et al. followed Zhao’s method for computing a complete distance field (Chang et al., 2008). The second approach is based on the exact computation of the distance between a voxel and the surface polygons. Some distance computations can be discarded based on their spatial coherence. Payne and Toga utilized this coherence by storing polygon data in a hierarchical bounding box (Payne and Toga, 1992). Guéziec extended this method to his meshesweeper algorithm (Guéziec, 2001). In this algorithm, the lower and upper bounds of distances between a voxel and any polygons in the box are used to reduce the distance computations.

Each feature (vertex, edge, or facet) of a triangle mesh can be converted into its corresponding characteristic polyhedron containing the points closest to the feature. The computation cost of the distance field can be reduced by classifying the voxels according to the characteristic polyhedrons (Sigg et al., 2003). Characteristic polyhedrons for all features of objects constitute a 3D Voronoi diagram (Okabe et al., 1992). Hoff et al. proposed a Voronoi diagram computation method accelerated by polygon rendering hardware (Hoff et al., 1999). Sud et al. improved their method and used it to compute the 3D distance field (Sud and Manocha, 2003)(Sud et al., 2006). The adaptively sampled distance field (ADF) records the distance values adaptively based on local details and stores the data in a spatial hierarchy for efficient processing. An ADF that stores distances at the cell vertices of an octree was proposed in (Frisken et al., 2000)(Kim, 2014)(Bastos and Celes, 2008). The distance value in a cell are derived from the trilinear interpolation of values at the vertices.

3. Bounded distance field

Bounded distance field is a special distance field existing only in a limited region around the boundary surface of an object. By using the bounded distance field, it is possible to analyze conditions such as minimum distance and minimum clearance specified between 3D objects at high speed. The same analysis is possible using conventional
distance fields and configuration space. However, since the arrangement of parts is frequently changed in the design process, it is necessary to recalculate the distance field and configuration space each time, which hinders interactive design process. Bounded distance field is more effective for the designer because it does not require recalculation due to the position and orientation change of the objects in the design process.

3.1 Definition

A distance field defined as bounded is one that is present only in a spatial zone around the object surface, as shown in Fig. 2. The thickness \( t \) of the zone is set to more than twice the value of the minimum clearance \( w \) specified in the guidelines. In our implementation, the spatial zone is represented as a voxel model. Distance values are recorded for the center points of all voxels of the model. Positive and negative distances are given to voxels inside and outside the object, respectively. The zone separates the modeling space of the internal and external spaces with respect to the zone. A sufficiently large “positive” value is specified for any points in the internal space (yellow region in the figure). Any points in the external space are recognized as having a sufficiently large “negative” value (blue region in the figure). Bounded distance field in the external space is useful for evaluating narrow gaps. On the other hand, the bounded distance field in the internal space is for detecting thin shapes in the part. Each bounded distance field has an associated bounding box. A definition of the box and how it is used are provided later in this paper.

As in the conventional distance field, a bounded distance field can be used to evaluate the thickness and clearance of a solid object. Fig. 3 presents the thickness computation results with both conventional and bounded distance fields for the same object. In the computation of the bounded distance field, \( t \) is set to evaluate the thickness smaller than 20% of the thickest width of the object. The color map given on the object surface refers to the thickness of the object as computed by our thickness visualization software (Inui et al., 2015). This software determines the object thickness using the distance values stored in the voxels. Red and blue denote zero and maximum thickness, respectively. A coloring result using the bounded distance field (Fig. 3(b)) is similar to that used with the conventional distance field (Fig. 3(a)) for thin shapes such as blades. However, shapes thicker than \( t/2 \) are painted with the same blue color to correspond to a sufficiently large value. As shown in Fig. 3, the bounded distance field is useful to detect thin shapes (or narrow gaps) on a single object.
3.2 Computation of bounded distance field

We obtain a bounded distance field around an object’s surface in the following three-step manner:

**Input**: A tessellated CAD model of an object and thickness value \( t \) of the zone of a bounded distance field.

**Step 1**: Define a bounding box containing the spatial zone, then convert the internal space of the box to a voxel model.

**Step 2**: For the center point of each voxel, compute the signed distance value.

**Step 3**: Eliminate the voxels whose absolute value of the distance is greater than \( t/2 \).

**Output**: Bounded distance field around the object’s surface in the voxel representation with a bounding box.

### 3.2.1 Step 1: Constructing a voxel model

An axis-aligned box enclosing the input polyhedral model is defined. The box is then expanded by \( t/2 \), where \( t \) denotes the thickness of the bounded distance field. The expansion is achieved by shifting the six rectangular faces of the box outwards. Construction of the voxel model is performed using the vertical ray method. Consider a uniform axis-aligned grid with equal intervals in the box. The spatial cell structure defined by the grid is projected onto the bottom face of the bounding box, which is a face parallel to the xy plane and with the smallest z coordinate. From each projected square cell, an upward ray is extended along the direction of the z axis from the center of the cell, and the intersection points of the ray with the surface polygons of the given model are computed. The intersection points on a ray are sorted based on their z coordinates, and a set of segments corresponding to the internal part of the object is derived. Spatial cells whose center is on the segments are selected as the internal voxels. Through a similar method, a set of segments corresponding to the external part of the object are computed and the external voxels locating on the segments are obtained.

### 3.2.2 Step 2: Computing a signed distance value

To prepare data suitable for the distance computation, surface polygons of the input model are classified into small groups based on the proximity between polygons. This classification proceeds by using an oriented bounding box (OBB) tree (i.e., a hierarchical structure of boxes (Gottschalk et al., 1996)(Moller and Haines, 1999)). The basic process in the distance field construction is to compute the distance between the center of each voxel and a polygon on the object surface. The computation cost can be reduced by properly culling the surface polygons that do not contribute to the distance computation. Traverse the hierarchical OBB tree from the root to the leaf nodes in a breadth-first manner. The culling operation is executed in each layer. Assume that \( n \) bounding boxes \((BBox_0, BBox_1, \ldots, BBox_{n-1})\) are found in a layer. Here, \( d_i \) represents the shortest distance between a voxel center \( v \) and a point on the surface of \( BBox_i \), whereas \( D_i \) represents the longest distance between \( v \) and a point on \( BBox_i \). Because polygons within \( BBox_i \) are bounded by the box, if \( d_i \) is greater than \( t/2 \), the polygons within the box do not contribute to the distance computation of \( v \). Therefore, this box is excluded from the following computation.

Select a box \( BBox_{\text{min}} \) whose value \( D_{\text{min}} \) is the smallest among the \( D_i \) values of the remaining boxes. If \( d_i \) is greater than \( D_{\text{min}} \), then the distance between \( v \) and any polygon within \( BBox_i \) must be greater than that between \( v \) and any polygon within \( BBox_{\text{min}} \). Therefore, polygons within \( BBox_i \) can be ignored when constructing the distance field. In the next layer in the breadth-first tree traversal, child bounding boxes of the remaining boxes are only evaluated in the culling operation. After the culling operation during the OBB-tree traversal, some OBBs enclosing polygons sufficiently close to \( v \) are obtained at the leaf nodes of the tree. If no OBBs are remained, point \( v \) has no polygons closer than \( t/2 \) and a sufficiently large value is returned as the distance value for \( v \). Otherwise, the point-polygon distance computation is applied to the polygons within the obtained leaf OBBs and \( v \). This computation is executed using a GPU in a parallel manner. See our previous study (Inui et al., 2015) for implementation details.

### 3.2.3 Step 3: Eliminating far voxels

After the distance computation, voxels whose absolute distance value is greater than \( t/2 \) are selected and eliminated from the voxel model. Because the vertical ray method is used in converting a polyhedral model to its equivalent voxel model, voxels located on the same vertical ray are recorded in association with a cell in the xy plane where the ray starts. For each cell in the xy plane, voxels on the ray are scanned, and voxels whose distance value is greater than \( t/2 \) or smaller than \( -t/2 \) are selected and erased. In the placement operation of component objects in a large object (e.g., in the layout design of water pipes in the mold cavity), bounded distance fields associated with objects must be inverted. This operation is realized by simply reversing the sign of the distance values of the voxels.
4. Fast detection of narrow gaps between objects

This section discusses the method of detecting narrow gaps between objects in the layout design using the bounded distance field.

4.1 Placement of objects with bounded distance fields

A bounded distance field of an object is defined based on an axis-aligned bounding box that encloses the object’s surface. An axis-aligned local coordinate frame is assigned to the bounding box such that its origin is at the corner of the box with the smallest x, y, and z coordinates. Any voxel in the bounded distance field can be uniquely indexed as \((n_i, n_j, n_k)\), where \(n_i\) denotes the position of the voxel in the spatial grid in the x-axis direction of the local coordinate frame, and \(n_j\) and \(n_k\) denote the position of the voxel in the directions of the y and z axes, respectively. Coordinates of the corner point \(c\) with the smallest coordinates in a voxel at \((n_i, n_j, n_k)\) are obtained by the following simple calculation:

\[
c = o + d \times (n_i i + n_j j + n_k k)
\]

where \(o\) represents the coordinates of the origin of the local coordinate frame in the modeling space \(W\), \(d\) is the size of each voxel, and \(i, j,\) and \(k\) denote unit vectors corresponding to x-, y-, and z-axis directions of the local coordinate frame in \(W\).

In the layout design, a designer often moves a component object by using rigid transformations such as translations and rotations. The same transformations are applied to the local coordinate frame associated with the bounding box of the object, and the object and bounded distance field surrounding the object are moved simultaneously, as shown in Fig. 4. These transformations are repeatedly applied to objects in the modeling space and multiple bounded distance fields with different positions and orientations are placed in the space. Because only coordinate transformations on the local coordinate frame are necessary, these transformations can proceed at an interactive speed.

![Fig. 4 Transformation occurring in an object and bounded distance field around the object.](image)

4.2 Evaluation of distance from a single object

Consider a single object with a bounded distance field placed in the modeling space. The distance of a point \(p\) from the object’s surface is determined according to one of the following conditions:

- If \(p\) is outside the bounding box of the bounded distance field, a sufficiently large negative value is returned as the distance of \(p\).
- If \(p\) is in the bounding box but not in the voxel model of the bounded distance field, then a sufficiently large negative value is returned as the distance of \(p\).
- If \(p\) is in a voxel model, then find a voxel \(v\) where \(p\) exists. Return the distance value \(d\) recorded at \(v\) as the distance of \(p\) from the object’s surface.

Consider a vector \(u\) from the origin point \(o\) of the local coordinate frame of the bounded distance field to the point \(p\). Project \(u\) to three axes of the local coordinate frame. If the length of one projected vector is not in the range of the box size (see Fig. 5(a)), then \(p\) does not exist in the bounding box of the bounded distance field. Otherwise, subdivide the length of the projected vectors by the voxel size to determine the location of the point of the voxel model of the bounded distance field. If no voxel exists at the obtained location (see...
Fig. 5(b)), $p$ exists outside the bounded distance field and is recognized as sufficiently far from the object’s surface. If a voxel is obtained at the location, the distance value recorded at the voxel is returned as the distance of $p$ (see Fig. 5(c)). The distance of $p$ obtained in this method is measured from a surface of the object in its normal direction. Therefore, it corresponds to the clearance value between $p$ and the object derived using the ray method for the thickness/clearance evaluation.

Fig. 5 Computation of the distance of $p$ from a single object. (a) $p$ is outside the bounding box, (b) $p$ is inside the box but outside the bounded distance field, (c) $p$ is inside the bounded distance field.

4.3 Evaluation of distance from multiple objects

Consider multiple objects placed in the modeling space. In this case, the distance of a point $p$ is determined using Boolean union of the bounded distance fields of the objects. This kind of distance field can be computed by using the same logic as when the conventional distance fields are combined (Frisken and Perry, 2006). In the layout design, a designer frequently changes the position and orientation of component objects to determine their optimal configuration. Because the translation and/or rotation of objects change the distribution of distances in the modeling space, Boolean union of the distance fields of the objects must be updated. Computing the combined distance fields consumes time. Therefore, frequently updating the distance field in the modeling space severely impairs the interactivity of the layout design.

Fig. 6 Evaluation of the distance of a point $p$ within two bounded distance fields $A$ and $B$.

To solve this problem, we introduce the idea of lazy evaluation into our distance computation with multiple objects. In this method, the distance evaluation of a point is executed by using individual distance fields associated with the objects, and the computation of the combined distance fields of all objects are postponed for as long as possible. Consider the distance computation of a point $p$ on an object $S$ in a space having multiple objects. By checking the intersection between the bounding box associated with $S$ and bounding boxes of other objects, bounded distance fields which can contain $p$ within are selected. In Fig. 6, $p$ is contained within two bounded distance fields, $A$ and $B$, of two objects. For each object, compute the distance of $p$ from the object’s surface by using its associated distance field as previously explained. After the distances for all objects are obtained, the maximum one is selected as the distance of $p$ from its nearest object (the maximum value is used because negative distance values are stored in the external voxels of the
bounded distance field).

![Bounded distance field of B](image)

Fig. 7 Problematic placement of two objects when an object wall is too thin.

### 4.4 Detection of narrow gaps

Consider the evaluation of clearances between multiple objects. Each surface polygon of the object is subdivided to a set of sufficiently small polygons of nearly the same size. A gap narrower than a predetermined value \( w \) can be visualized by painting surface polygons on an object with a unique color (red in our implementation) if the distance of one of its vertices from the surfaces of the other objects is smaller than \( w \). In this method, the cost for visualizing narrow gaps between objects is proportional to \( nm \) where \( n \) is the number of objects in the modeling space and \( m \) is the number of polygons of all objects. In our experiments, the visualization can be executed at a speed sufficiently fast for the interactive operation when \( n \) is less than 5 and number of polygons of each object is around 1 million.

![Detection result of spaces between water pipes and mold surface that are too narrow](image)

Fig. 8 Detection result of spaces between water pipes and mold surface that are too narrow.

Our method generally returns reasonable detection results. However, some problematic cases exist. Fig. 7 illustrates one of these cases. Object \( A \) with a thin wall locates at a position very close to another object \( B \). Because the wall of \( A \) is very thin, we detect not only a polygon on the wall facing \( B \) (polygon \( f_0 \) in the figure), but we also detect an additional polygon on the opposite side of the wall (polygon \( f_1 \) in the figure) that is too close to \( B \). Note that this type of special placement rarely occurs (i.e., does not occur in normal use).
Fig. 9 Application result of our system to assist water circuit placement in a complex mold cavity.

5. Application for assisting water circuit layout

In the injection molding of a plastic part, the temperature of the part becomes 200 °C immediately after the molding process is completed. To cool down the molded part, a water circuit is embedded in the mold cavity. In the guidelines of the mold cavity design, the distance between the mold surface and water pipes must be greater than 2.5
times the diameter of the pipe. If the distance is smaller than this, the temperature drop of the part will be too fast and distortions will occur in the molded product. The water pipe is fabricated by the drilling operation. Therefore, only straight pipes are allowed in the water circuit. Because the mold surface generally has a complex curved shape, understanding the distance between the water pipe and mold surface is difficult for the designer.

To demonstrate the practical application of a bounded distance field, we implemented a system that assists in a water circuit layout using VisualStudio and CUDA 7.0. Fig. 8 illustrates the simple placement process of a water pipe in a mold cavity. A grey cylinder represents the pipe shape. To visualize the placement of the pipe, a geometric model of the mold cavity is given in the wireframe illustration. When the pipe approaches the mold surface, the surface region (painted in red) warns that the pipe is too close to the mold surface. Fig. 9 (a) and (b) present two application results of the system for water circuits and mold cavities with considerably complex shapes. In order to maintain confidentiality, the shape of the models is modified much from the actual parts. In these computations, bounded distance fields with 20 million voxels are used. Some parts of the mold surface (see the arrows in the figure) are properly painted in red to show where the water circuit is too close to the mold surface. 1.05 second is necessary for computing the distance values for all polygons and obtaining a colored picture through an interactive placement process of water pipes using a PC with Core i7 (4.0GHz), 16GB memory and GTX-1060 GPU (6GB memory).

6. Conclusions

In this study, we presented a software system to assist in the placement of component objects when dimensional constraints are imposed on the minimum thickness and clearance between objects. A distance field defined in a limited zone around the boundary surface of a shape was introduced. Narrow gaps generated during part positioning were detected by checking the distance value at the surface polygon of the objects in the field. Based on the new placement of an object, the distance field in the modeling space must be updated. To reduce the update cost, a thickness and clearance evaluation was executed by using individual distance fields, and the computation of combined distance fields of all objects was postponed for as long as possible. To evaluate the practical applicability of our method, an interactive system for assisting layout of a water circuit in a mold cavity was demonstrated.

Acknowledgements

This research was supported by JSPE KAKENHI (Grant Number 17K06075)

References

Bastos T., Celes W., GPU-accelerated adaptively sampled distance fields, Proc. of IEEE International Conference on Shape Modeling and Applications (2008) pp.171–178.
Chang B., Cha D., Ihm I., Computing local signed distance fields for large polygonal models, Computer Graphics Forum, Vol. 27, No. 3 (2008) pp.799–806.
Cuisenaire O., Distance transformations: Fast algorithms and applications to medical image processing, Ph.D Thesis (1999) Universite Catholique de Louvain.
Danielsson P.E., Euclidean distance mapping, Computer Graphics and Image Processing, Vol. 14 (1980) pp.227–248.
Frisken S.F., Perry R.N., Pockwood A.P., Jones T.R., Adaptively sampled distance fields: A general representation of shape for computer graphics, Computer Graphics Proceedings, Siggraph ’00 (2000) pp.249–254.
Frisken S.F., Perry R.N., Designing with distance fields, Proceeding SIGGRAPH ’06 (2006) pp.60-66.
Gottschalk S., Lin M.C., and Manocha D., OBBTree: a hierarchical structure for rapid interference detection, SIGGRAPH ’96 Proceedings of the 23rd annual conference on computer graphics and interactive techniques (1996) pp.171-180.
Gu’eziec A., Meshsweeper. Dynamic point–to–polygonal mesh distance and applications, IEEE Transactions on Visualization and Computer Graphics, Vol. 7, No. 1 (2001) pp.47–61.
Hoff K.E., Culver T., Keyser J., Lin M., Manocha D., Fast computation of generalized Voronoi diagrams using graphics hardware, Computer Graphics, Proc. Siggraph ’99 (1999) pp.277–286.
Inui M., Umezu N., Wakasaki K., Sato S., Thickness and clearance visualization based on distance field of 3D objects, Journal of Computational Design and Engineering, Vol. 2, No. 3 (2015) pp.183–194.
Jones M.W., Bærentzen J.A., Sr´amek M., 3D Distance Fields: A survey of techniques and applications, IEEE Transaction on Visualization and Computer Graphics Vol. 12, No. 4 (2006) pp.881–599.

Kim Y.J., Exact and adaptive signed distance fields computation for rigid and deformable models on GPUs, IEEE Transaction on Visualization and Computer Graphics, Vol. 20, No. 5 (2014) pp.714–725.

Latombe J.-C., Robot Motion Planning (1991) Kluwer Academic Publishers.

Lu S.C., Rebello A.B., Miller R.A., Kinzel G.L., Yagel R., A simple visualization tool to support concurrent engineering design, Computer-Aided Design, Vol. 29, No. 10 (1997) pp.727–735.

Moller T., Haines E., Real-Time Rendering (2008) CRC Press.

Morimoto K., Inui M., A GPU based algorithm for determining the optimal cutting direction in deep mold machining, Proc. IEEE International Symposium on Assembly and Manufacturing (ISAM 2007) (2007).

Mullikin J.C., The vector distance transform in two and three dimensions, CVGIP: Graphical models and image processing, Vol. 54, No. 6 (1992) pp.526–535.

Okabe A., Boots B., Sugihara K., Spatial tessellations, concepts and applications of Voronoi diagrams (1992) John Wiley & Sons.

Payne B.A., Toga A.W., Distance field manipulation of surface models, Computer Graphics and Applications, Vol. 12, No. 1 (1992) pp.65–71.

Rhodes F., Discrete Euclidean metrics, Pattern Recognition Letters Vol. 13, No. 9 (1992) pp.623–628.

Rosenfeld A., Pfaltz J.L., Sequential operations in digital picture processing, Journal of the ACM, Vol. 13, No. 4 (1966) pp.471–494.

Sigg C., Peikert R., Gross M., Signed distance transform using graphics hardware, Proc. of IEEE Visualization ’03 (2003) pp.83–90.

Sinha B., Efficient wall thickness analysis methods for optimal design of casting parts, Presented at Engineering Design (2007).

Subburaj K., Patil S., Ravi B., Voxel-based thickness analysis of intricate objects, International Journal of CAD/CAM, Vol. 6, No. 1 (2006).

Spitz S.N., Spyridi A.J., Requicha A.A.G., Accessibility analysis for planning of dimensional inspection with coordinate measuring machines, IEEE Trans. Robotics and Automation Vol. 15, No. 4 (1999) pp.714–727.

Sud A., Manocha D., Fast distance field computation using graphics hardware, UNC Computer Science Technical Report TR03-206 (2003).

Sud A., Govindaraju N., Gayle R., Manocha D., Interactive 3D distance field computation using linear factorization, Proc. of the 2006 symposium on interactive 3D graphics and games (2006) pp.117–124.

Zhao H., A fast sweeping method for Eikonal equations, Mathematics of computation, Vol. 74, No. 250 (2004) pp.603–627.