A Proposal for an Algorithm for Calculating Force between Digital Objects at High Speed

Hiroshi Noborio
Department of Computer Science, Osaka Electro-Communication University, Japan
Email: nobori@osakac.ac.jp

***Abstract*—After the depth/RGB camera Kinect was firstly developed by Microsoft around 2010, an object is frequently captured and simultaneously is dealt with a set of point crowd. Therefore, we have been focusing on algorithms concerning to overlapping two or more crowds, distance calculation between two or more crowds, intersection check between two more crowds. On these circumstances, we focus on how to generate an artificial force with arbitrary direction and magnitude when two or more objects are interrupted each other in this paper. Needless to say, their objects are expressed by digital data such as point cloud, voxel cloud, sphere cloud, and their hierarchical structure in positioning. In this method, collision detection area, cuttable area, and representative point are set freely. As a result, we can select to calculate one of various normal vectors flexibly and rapidly even for rigid bodies or soft tissue composed of multiple materials. In this circumstance, we propose a novel algorithm which generates an artificial force flexibly when digital clouds are encountered each other.**

***Index Terms*—point cloud, voxel cloud, sphere cloud, interference, normal vector, force magnitude and direction**

I. Introduction

Before 1980, an object is represented by B-reps, or Constructive Solid Geometry [1]. For this reason, collision between two or more objects, or distance calculation between two or more objects are done by them [2]. However, those become time consuming as long as obstacle shape is more complicate. Therefore in 1980-1990, digital data was frequently used for checking the intersection or distance calculation between two or more objects [3]-[7]. This technique was spanned for digital game area. After that, we innovate depth camera and GPU mentioned later for the digital game.

In succession, their collision-detection algorithms were widely used for medical and dental surgical simulation and navigation [8]-[13]. The first depth camera Kinect was commercialized from Microsoft around 2010 and consequently it was sold by 35 million sets [14]. By using this, point crowds of object can be easily captured. One of many applications was to be SLAM (Simultaneous Landmark and Mapping) [15]-[17]. Also, parallel processing of GPGPU was also widely used in several graphics board for digital game [18]. The matching between point crowed and multicore of GPU is very wonderful [19]-[23].

However, motion-detection of human, collision-detection of obstacle, and fusion of multiple crowds are the main targets in digital entertainment, robotics, computer vision, augmented reality, and so on. Unfortunately, no algorithm to generate an impact force whose normal direction and magnitude is adaptively changed. Using the impact force generation algorithm, the forces acting on multiple objects are rapidly found by directly deriving the direction of such forces from the data of the digitally represented objects. Such algorithms will be useful for robot assembly, force feedback (haptics), and so on.,

Below we describe the method for calculating a digital normal with reference to the accompanying drawings. A computer Graphics Processing Unit (GPU) is used for calculating the digital normal for shape 1 (that is, for applying the calculation algorithm). Such GPUs applied to implement the algorithm when calculating the digital normal are freely programmed using a specialist programming language such as the Open Computing Language (OpenCL) and Computing Unified Device Architecture (CUDA).

Herein, a digital normal refers to a directional line segment that has a direction matching the direction of the force (or the reaction force to that force) acting on an object expressed in the "digital data (digital model)" (a point group, voxel group, and sphere groups, together with an octree or Hierarchical Sphere Model (HSM) expressing a hierarchical structure with respect to the position of the objects). In this form of implementation, as an example, the digital normal is derived as a vector quantity that closely resembles the direction of the force acting between objects when multiple objects expressed as digital data interact with one another.

Below, we describe an example method for calculating (calculation algorithm) the digital normal in which the normal vector of the force acting between two objects is sought when a certain object (a machinable object) is cut using another object (a cutting tool).

II. Impact Force Generation Algorithm

This algorithm is composed of the following five steps. In the first step, data on objects 1 and 2 are obtained in a digital format. In the second step, the position and amount by which object 1 is embedded into object 2 are determined. In the third step, the magnitude of force acting

Manuscript received September 8, 2021; revised January 22, 2022.

doi: 10.18178/joig.10.1.36-42

on object 1 is determined based on the position and amount of embedding found in step 2. In the fourth step, the direction of force acting on object 1 is determined based on the direction of the vector obtained through a summation of the base vectors obtained as vectors aimed at a representative point satisfying the fixed positional relationship with the minute elements from the center of gravity of the partial body of object 2, including the region satisfying the fixed positional relationship for each minute element in which object 1 interfaces with object 2. Finally, in the fifth step, the force acting on Object 1 is determined and output.

A. Step 1

The cutting object (an object wielded by a doctor or a dentist, for example, a dental bur) and an object to be cut (an object to be manipulated by a doctor or dentist, for example, a tooth) are converted into digital information (a point group, cube group, or sphere group, etc.) (see Fig. 1). Digital information (digital data) includes the point group of a dental bur, as shown in Fig. 2; the cube group (octree representation) of a row of teeth, as shown in Fig. 3; or as shown in Fig. 4, a rectangular parallelepiped group processed using the Z buffer and parallel processing function of the GPU. In addition, particle groups express deformable soft tissue.

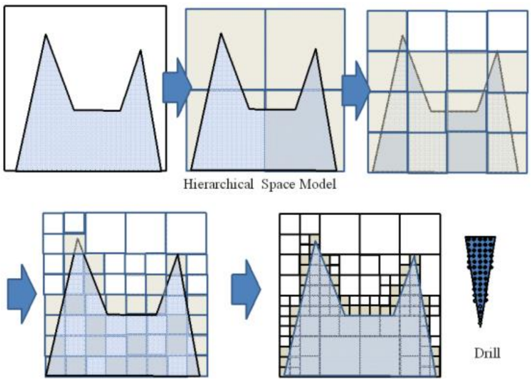

Figure 1. An example diagram of teeth and a bur represented using digital data.

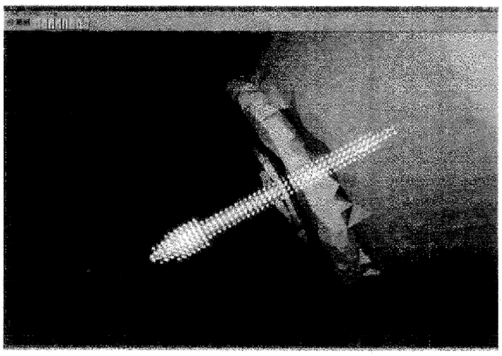

Figure 2. An example diagram of a dental bur expressed using digital data (point group).

In Fig. 1, a conceptual diagram of a method used to express a tooth (a machinable object) and a tooth bur drill (a cutting tool), represented as an octree expression of a cube group (voxel) and point group, respectively, is shown. An example diagram of a drill expressed as a point group and a dental bur to which the drill is attached is shown in Fig. 2. An example diagram of a tool expressed using an octree of a voxel group is shown in Fig. 3.

Figure 3. An example diagram of a tooth expressed using digital data (a voxel group).

A conceptual diagram used to derive a representation of tooth 12 using a rectangular parallelepiped group using the Z-buffering function of the GPU is shown in Fig. 4. Here, the object represented by the rectangular parallelepiped group is also included among those objects represented using digital data. The GPU, not shown in the figure, obtains and stores the area in which tooth 12 is present in a direction perpendicular to Z buffer 20. Note that it is assumed herein that tooth 12 is an object that can be expressed using data recorded in Standard Triangulation Language (STL). In so doing, data (digital data) representing tooth 12 as a rectangular parallelepiped group are stored in Z buffer 20. In addition, the GPU, not shown in the figure, can also divide the rectangular parallelepiped group into smaller cubes (voxels). At this time, the GPU is able to differentiate each cube, and using the relationship with the existing range of tooth 12 recorded in Z buffer 20, each voxel is divided into smaller voxels inside tooth 12 (Fig. 4(b)), voxels that do not make up tooth 12 (Fig. 4(c)), and voxels that form the interface of tooth 12 (Fig. 4(d)). Moreover, similarly for the drill at the tip of the dental bur (Fig. 2), the drill data expressed using the STL format are first converted into a voxel group, which can then be converted into data (digital data) expressing a point cloud using a voxel vertex group or barycenter group.

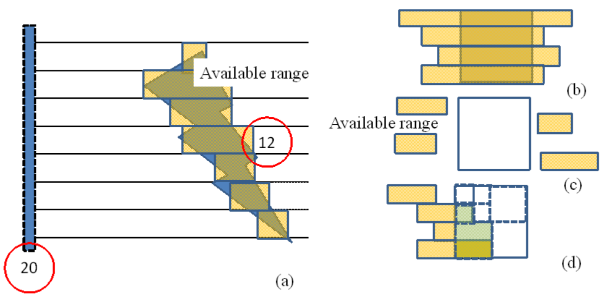

Figure 4. A conceptual diagram of a rectangular parallelepiped representation of an object.

Thus, in Step 1, a digital data representation of the object to be simulated in terms of a mechanical interaction is obtained. Note that the step for conversion into digital data, shown in Fig. 4, is not essential, and digital data representing the simulation subject may be obtained using another method.

B. Step 2

Here, we check the interaction between the digital information (digital data) acquired in Step 1 (Fig. 5). For example, the points of the dental bur in Fig. 2 are applied to the cube group (octree) of the row of teeth shown in Fig. 3, and the interface between the points and cubes is checked either sequentially or in parallel. However, it is advantageous for the coordinate systems of both objects subject to an interface check to be matched in advance. For example, where there are two rectangular parallelepiped groups being operated on by the GPU, after matching their coordinate systems in advance, their interface is checked according to a shared coordinate system (see Fig. 4).

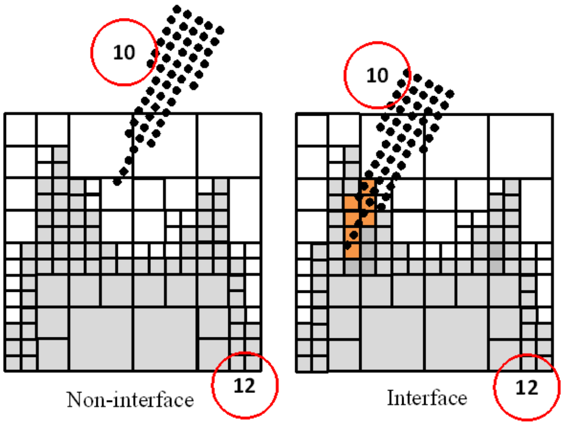

Figure 5. A conceptual diagram for checking the interface between a bur and a tooth.

Note that the checking of an interaction is not limited to the above method. Thus, when an interaction is detected in Step 3, the degree to which the cutting tool sinks into the workpiece (the relative relational position of the two objects) is determined based on the cutting difficulty of the workpiece (hardness, etc.) and the cutting ability of the cutting tool.

C. Step 3

Here, the embedding position is determined using the positioning information of the object to be cut (workpiece) along with its hardness (for example, the tooth voxels hold different hardness data for each material quality, such as the enamel, dentin, pulp, and caries) and the location information of the cutting object (which should include the speed, acceleration, and position) and the cutting ability (the cutting area is defined separately from the shape of the object, provisioned for materials with easy-to-cut sections and parts that are harder to cut, as well as various objects that range from easy to difficult to cut) (see Fig. 5). At this time, taking the state of interaction between the two objects as an index, their movements are searched using a binary search, Newtonian method, or other similar approach, and thus the embedded position (that is, the amount of embedding) is determined at a high speed through trial and error. Here, this index should be an absolute value (magnitude) of a force acting on the two objects, calculated based on the Kelvin-Voigt model. In addition, the index may be an absolute value (magnitude) of a force calculated based on a viscoelastic model other than the Kelvin-Voigt model. Furthermore, the index may be a quantity derived from a mechanical model other than a viscoelastic model. Alternatively, the index may be an embedded state represented by a function, or if an embedded state cannot be expressed using a function, it may be presented using a lookup table.

In this way, the relative positional relationship between the workplace and the cutting tool is determined at this point. In Step 4, the force acting on the cutting tool (the reaction force that the cutting tool receives from the workpiece) is then calculated for the positional relationship determined in Step 3.

D. Step 4

First, the absolute value (magnitude) of the force (reaction force from the workpiece received by the cutting tool) is determined from the embedding depth determined in Step 3. In this case, a method employing the Kelvin-Voigt model is commonly used. That is, a spring constant is multiplied by the amount of embedding, and the relative speed immediately before an interaction is multiplied by a damper constant; these products are then added together. However, this is not limited to the use of the Kelvin-Voigt model as long as the state of embedding between the two colliding objects, along with the speed and acceleration immediately before the collision, is used.

When applying the Kelvin-Voigt model, the absolute value (magnitude) of the force acting on the cutting tool is determined as follows: First, the coefficient of drag (spring constant $\boldsymbol{k}$) applied to deform the workpiece (tooth), and the coefficient of drag (viscosity constant $\boldsymbol{c}$) corresponding to the speed of the deformation, are obtained. Here, spring constant $\boldsymbol{k}$ is determined using the following:

$$k=(Ne \times ke + Ni \times ki + Np \times kp)/(Ne + Ni + Np) \quad (1)$$

where $\boldsymbol{Ne}$ is the number of enamel voxels that interface with the cutting tool, $\boldsymbol{ke}$ is the spring constant of the enamel voxels, $\boldsymbol{Ni}$ is the number of dentin voxels that interface with the cutting tool, $\boldsymbol{ki}$ is the spring constant of the dentin voxels, $\boldsymbol{Np}$ is the number of pulp voxels that interface with the cutting tool, and $\boldsymbol{kp}$ is the spring constant of the pulp voxels. The viscosity constant $\boldsymbol{c}$ is obtained using the following:

$$c=(Ne \times ce + Ni \times ci + Np \times cp)/(Ne + Ni + Np) \quad (2)$$

here, $\boldsymbol{ce}$ is the viscosity constant of the enamel voxels, $\boldsymbol{ci}$ is the viscosity constant of the dentin voxels, and $\boldsymbol{cp}$ is the viscosity constant of the dental pulp voxels. Next, based on the amount of embedding, $\boldsymbol{d}$, of the cutting tool (drill) found during Step 3, and the relative speed of the cutting tool, $\boldsymbol{v}$, with respect to the workpiece, which is determined separately, the magnitude of the reaction force $\boldsymbol{FR}$ is determined using the following:

$$FR = k \times d + c \times v \quad (3)$$

In this way, the magnitude of the reaction force received by the cutting tool can be determined in Step 4. The direction of this reaction force is then calculated in Step 5.

E. Step 5

Here, the direction of the reaction force is determined. Specifically, the digital normal is calculated based on the relative positional relationship between the cutting tool and the workpiece.

First, collision detection area ***D*** and cuttable area ***C***, described later, are set. Next, a region (a "collision area" or an "interface area") in which collision detection area ***D*** is embedded into another object (workpiece) is calculated. Then, where the workpiece is represented by a rectangular parallelepiped group, as shown in Fig. 4, the embedding area (collision area) is discretized (divided) into minute elements (if there are digital data that from the outset separate the collision area of the workpiece into a set of microelements, this may also be used), and the group of minute elements that constitute the collision area is expressed as a set of minute regions (regions expressed as digital data, cubes (voxels), spheres, or points).

Note that collision detection area ***D*** is a three-dimensional area that can be set to any arbitrary shape. As an example, collision detection area ***D*** is a three-dimensional area that has the same shape as the object being manipulated (cutting tool). Collision detection area ***D*** may also differ from the shape of the tool being manipulated (cutting tool). For example, in Fig. 6(a), the entire object (cutting tool) is shown as collision detection area ***D***, whereas in Fig. 6(b)-(d), the areas of collision detection area ***D*** are shown with shapes differing from those of the object (cutting tool) (rectangular parallelepiped, cutting surface only, and an ellipsoid). The presence or absence of an interaction with the workpiece is detected in collision detection area ***D***. Cuttable area ***C*** can also be set to any shape, and may be set to the same shape as that of collision detection area ***D***. Cuttable area ***C*** may alternatively take a shape different to that of collision detection area ***D***. For example, in Fig. 7(a) and (b), the collision area (collision voxel set ***B***) is divided into three collision voxels. In Fig. 7(c)-(e) a rectangular parallelepiped cutting area ***C***, spherical cutting area ***C***, and ellipsoidal cutting area ***C*** are set, respectively. Moreover, representative point ***Gr*** is set in cuttable area ***C***. Representative point ***Gr*** of cuttable area ***C*** should be matched with a characteristic point based on the geometric shape of area ***C***, such as the center of gravity or the center of the area (Fig. 8(a)-(c)). However, there is no particular restriction on the relationship between representative point ***Gr*** for area ***C*** and area ***C*** itself; characteristic point ***Gr*** may be set based on any arbitrary relationship to area ***C*** (Fig. 8 (a)-(f)).

Next, a base vector (a vector that is fundamental to the calculation of the digital normal vector) is calculated for each minute area (for each minute element obtained by dividing the collision area). With reference to Fig. 9, we will now describe the method used for calculating the base vectors.

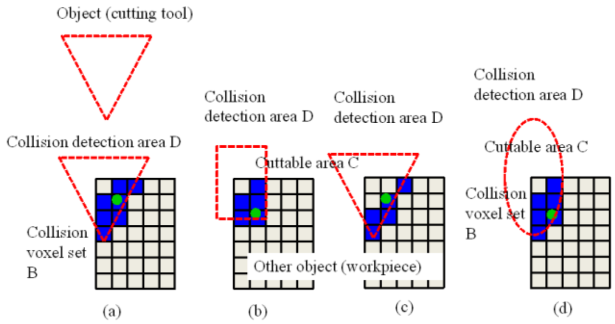

Figure 6. A diagram describing the free setting of collision detection area D.

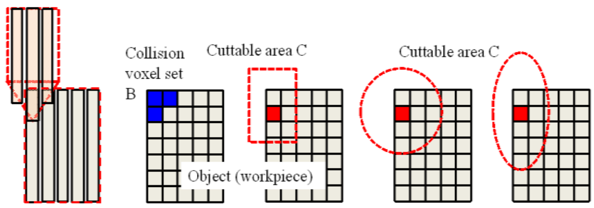

Figure 7. A diagram describing the free setting of cuttable area C.

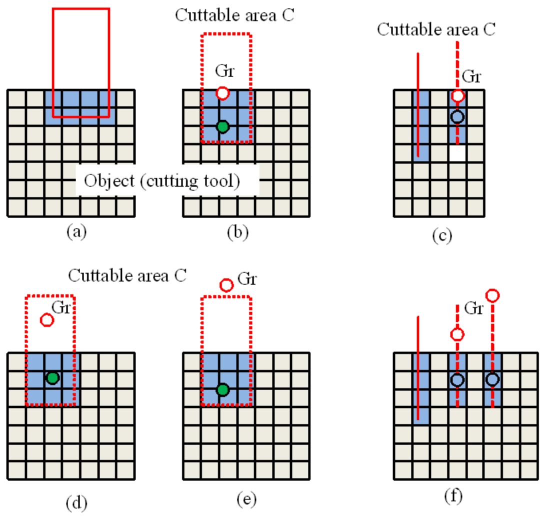

Figure 8. A diagram describing the free setting of representative point Gr.

Fig. 9 is a conceptual diagram that shows the process for deriving the base vector for the embedding amount calculated in Step 3. The derivation of the base vectors in tooth 12 with a uniform hardness are shown in Fig. 9(a)-(d), and the derivation of base vectors in which the hardness (cutting difficulty) is non-uniform owing to factors originating in the structure of the tooth are described in Fig. 9(e)-(h). As described above, when a hard section and a soft section are combined in a part of a workpiece, it can create a variation in the base vector, for example, by changing the position of the center of gravity ***Gc***. Needless to say, the changes in the force direction owing to the combination of hard and soft parts in the workpiece are not limited to this.

Regarding the range of embedding amount determined in Step 3, tooth 12 (workpiece) is represented by a rectangular parallelepiped group, as shown in Fig. 4, and divided into minute elements (Fig. 9(a)), which include voxels, as an example. The minute elements may be elements such as points or spheres.

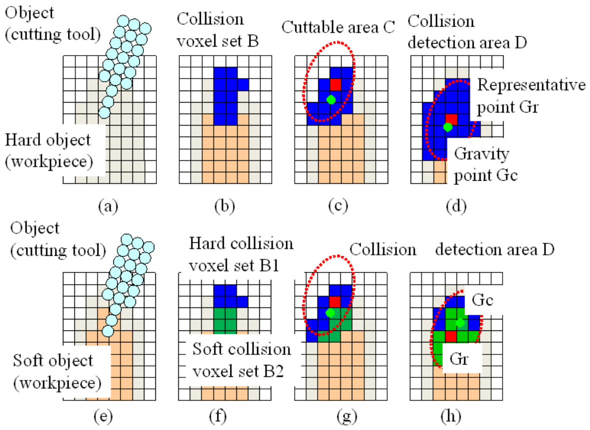

Figure 9. A diagram showing an example calculation involving the cutting of multiple materials with a different cutting difficulty (hardness, etc.).

Next, the collision area (where drill 10 and tooth 12 interface) of the tooth voxel (the collision voxel) is specified. In Fig. 9(b), the set of tooth voxels in the collision area is referred to as collision voxel set ***B***.

Then, in collision voxel set ***B***, the respective base vectors are obtained for each collision voxel. For example, for the second collision voxel from the left in the second row from the top of set ***B*** in Fig. 9(b), cuttable area ***C*** is set around the corresponding collision voxel, and the center of gravity ***Gc*** of the tooth voxel, including cuttable area ***C***, is found. Note that voxels only partially within area ***C*** may be included. Then, from the center of gravity ***Gc*** of the tooth voxel, which includes cuttable area ***C*** set as the center of the collision voxel of interest, the vector heading toward the representative point of the corresponding area ***C***, Gr (vector ***GcGr***), is set as the base vector of the collision voxel of interest (Fig. 9(c) and (g)). Base vectors are obtained for the other collision voxels in the same manner. Example base vectors found for the third collision voxel from the left in the fifth row from the top of set ***B*** are shown in Fig. 9(d) and (h).

Assuming that the collision voxels that constitute collision voxel set ***B*** are ***bj*** (j = 1, 2, ..., n), the base vector ***fj*** (j = 1, 2, ..., n) is written as follows:

$$\vec{f_j} = \overrightarrow{Gc_jGr_j} \tag{4}$$

here, ***Grj*** is the position of the representative point of cuttable area ***C*** set around collision voxel ***bj***, whereas ***Gcj*** is the center of gravity of the tooth voxel, including cuttable area ***C*** set centering around collision voxel ***bj***. Here, it should be noted that representative point ***Grj*** for cuttable area ***C*** satisfies a fixed positional relationship between every ***j*** and collision voxel ***bj***. This is because representative point ***Gr*** for area ***C*** is set at a fixed (arbitrary) position with respect to area ***C***, and area ***C*** is set at a fixed position (an arbitrary position, matching the center of area ***C***, for example) for each collision voxel ***bj***. Moreover, ***Grj*** is the position of representative point ***Gr*** in collision detection area ***C*** set centering on collision voxel ***bj***. The position of representative point ***Gr*** in area ***C***, ***Grj***, set to satisfy a fixed (arbitrary) positional relationship with corresponding voxel ***bj*** within the range of collision voxel ***bj***, may be used in the calculation of base vector ***fj***.

By taking the average of the base vector ***fj*** (j = 1, 2,...n) obtained in this manner, a normal vector (a vector matching the direction of the reaction force) is calculated. Specifically, normal vector ***F*** is given by the following:

$$\vec{F} = \frac{1}{n}\sum_{j=1}^{n}\vec{f_j} \tag{5}$$

The cutting difficulty (hardness) of each collision voxel caused by internal tooth tissues can be considered in the derivation of the base vector. In this case, the tooth voxel may hold a coefficient relating to such difficulty (hardness). For example, each voxel contains a real coefficient (***h*** > 0) that increases along with the hardness. Moreover, in calculating the center of gravity ***Gc***, the difference in the density of the tooth voxels can be considered (Fig. 9). In this case, the tooth voxel may contain density data. For example, if we refer to Fig. 9(e)-(h), the collision area extends from the tooth voxel corresponding to the tooth enamel to the tooth voxel corresponding to the dentin. The collision voxel set is therefore composed of enamel and dentin tooth voxels. At this time, base vector f becomes the following:

$$\vec{f_j} = h_j \cdot \overrightarrow{Gc_jGr_j} \tag{5'}$$

here, as described above, the center of gravity ***Gcj*** can be derived under consideration of the density of a tooth voxel as well as its volume.

Moreover, in deriving the base vector, it is also possible to provide a distribution to the cutting ability of drill 10 (cutting tool). For example, a specified section of drill 10 (e.g., the outer edge of the tip) may be provided a comparatively high cutting ability, whereas another section of drill 10 (e.g., the core of the tip) may be given a comparatively low cutting ability. The effect of this can be reflected in the magnitude of the base vector in the same form as correction coefficient h above, which concerns the hardness of the tooth. In this case, whether the collision voxels collide with any part of drill 10 is determined, and according to the result, the right side of Equation (5') is further multiplied by a correction coefficient regarding the cutting ability of the drill.

Vector FR, which expresses the reaction force, is then derived based on the magnitude of reaction force ***FR*** obtained in Step 4, as well as the normal vector ***F*** found in this step. For example, ***FR*** is described as follows:

$$\overrightarrow{FR} = \frac{|FR|}{|\vec{F}|} \times \vec{F} \tag{6}$$

Note that the absolute value of vector ***F*** on the right side of Equation (6) may be omitted. Even in this case, the magnitude of reaction force vector ***FR*** can change in proportion to the magnitude of the reaction force. Moreover, the magnitude of ***FR*** can be adjusted by changing the setting regarding the relation between representative point ***Gr*** in cuttable area ***C*** and area ***C*** itself.

III. Discussion

With the present method, the direction normal to the object in the collision area is derived from the point of interest (collision voxel ***bj***) in the collision area and the center of gravity (***Gcj***) of the object mass with a non-uniform density within the vicinity (area ***C***), and by taking the average of the base vector, the direction normal to the object expressed using digital data with no information regarding the surface is determined. Moreover, by changing the shape of collision detection area ***D***, the shape of cuttable area ***C***, and the position of representative point ***Grj*** of area ***C***, a constant bias can be applied to the force vector obtained. By adjusting the amount of this bias, it is possible to apply a constant bias to a haptic input/output device, as described later, and it thus becomes possible to appropriately adjust the tactile sensation perceived by the user from this haptic input/output device.

IV. Conclusion

As described previous, this algorithm can rapidly calculate the force (vector quantity) acting on an object through the dynamic interaction between objects represented by digital data. Note that when the force generated around the embedding area of the two colliding objects is controlled by the Newtonian equation of motion, it is also possible to determine the physical quantities relating to the parallel movement of the two objects. In addition, if the force generated is processed using the "moment of force" concept, or if a relation similar to the Newtonian equation of motion is used between the inertial moment of object and the angular acceleration α, it is also possible to determine the physical quantity relating to the rotational movement of the two objects.

In our method, collision detection area D, cuttable area C, and representative point Gr may be set freely when applying the present method. As a result, with the present method, normal vectors can be calculated rapidly even for rigid bodies or soft tissue composed of multiple materials.

Moreover, because this method uses the digital data of an object, the number of calculations can be reduced. By outputting the reaction force obtained from such calculations through an interface device capable of transmitting a tactile sensation (force sensation), the user can be informed of the hardness and deformation of the object. In the above, we present an example of a drill (cutting tool) and a tooth (object to be cut) as example objects represented using digital data. Objects are not limited to these, however, and cutting tools include scalpels, drills, scalers, and injection needles, whereas cuttable objects include bone, cartilage, internal organs, muscle, fat, skin, blood vessels, and nerves.

Furthermore, the use of the embedding position (amount of embedding) obtained in Step 3 of this algorithm makes it easy to express the interaction between the drill and tooth, visually. In addition, using information regarding the embedding position (amount of embedding), even where the tooth deforms from the drilling applied, because GPUs are originally processors specialized in displaying objects expressed using data (triangular polyhedron) in an STL format, when the deformation of the rectangular parallelepiped group and the deformation of the STL-format data are linked, it also becomes simple to display the deformed tooth visually. In other words, the present method is not only useful for providing tactile information at high speed and with high quality in simulations such as a cutting action in a virtual reality space, it can also be helpful in generating visual information in the corresponding simulation.

Conflict of Interest

The author declares no conflict of interest.

References

[1] G. Elber and V. Shapiro, "Solid modeling theory and applications," *Computer Aided Design*, vol. 36, no. 11, pp. 1011-1116, 2004.

[2] C. J. Ong and E. G. Gilbert, "Fast versions of the Gilbert-Johnson-Keerthi distance algorithm: Additional results and comparisons," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 4, pp. 531-539, 2001.

[3] H. Noborio, S. Fukuda, and S. Arimoto, "A new interference check algorithm using octree," in *Proc. IEEE International Conference on Robotics and Automation*, Raleigh, North Carolina, 1987, pp. 1480-1485.

[4] Y. H. Liu, H. Noborio, and S. Arimoto, "Hierarchical Sphere Model (HSM) and its application for checking an interference between moving robots," in *Proc. IEEE Int. Work. Intel. Robot and Systems*, Tokyo, Japan, 1988, pp. 801-808.

[5] H. Noborio, S. Fukuda, and S. Arimoto, "Fast interference check method using octree representation," *Journal of Advanced Robotics*, vol. 3, no. 3, pp. 193-212, 1989.

[6] S. Gottschalk, M. Lin, and D. Manocha, "OBBTree: A hierarchical structure for rapid interference detection," in *Proc. the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, New Orleans, USA, 1996, pp. 171-180.

[7] G. V. D. Bergen, "Efficient collision detection of complex deformable models using AABB trees," *Journal of Graphics Tools*, vol. 2, pp. 1-13, 1997.

[8] H. Noborio, D. Sasaki., Y. Kawamoto, T. Tatsumi, and T. Sohmura, "Mixed reality software for dental simulation system," in *Proc. the IEEE International Workshop on Haptic Audio Visual Environments and Games*, 2008, pp. 19-24.

[9] H. Noborio and Y. Kawamoto, "Digital collision checking and scraping tooth by dental bar," in *Proc. 3rd IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics*, Tokyo, Japan, September 2010, pp. 227-234.

[10] Y. Yoshida, S. Yamaguchi, Y. Kawamoto, H. Noborio, S. Murakami, and T. Sohmura, "Development of a multi-layered virtual tooth model for the haptic dental training system," *Dent. Mater. J.*, vol. 30, no. 1, pp. 1-6, 2011.

[11] I. R. D. Boer, M. Lagerweij, P. Wesselink, and M. M. Vervoorn, "Evaluation of the appreciation of virtual teeth with and without pathology," *European Journal of Dental Education*, vol. 19, no. 2, pp. 87-94, July 2014.

[12] A. Kato, A. Ziegler, M. Utsumi, and T. Takeichi, "Three-dimensional imaging of internal tooth structures: Applications in dental education," *Journal of Oral Biosciences*, vol. 58, pp. 100-111, June 2016.

[13] S. Perry, M. F. Burrow, W. Leung, and S. M. Bridges, "Simulation and curriculum design: A global survey in dental education," *Australian Dental Journal*, vol. 62, no. 4, pp. 453-463, April 2017.

[14] Wikipedia. Kinect. [Online]. Available: https://en.wikipedia.org/wiki/Kinect

[15] J. Pan and D. Manocha, "GPU-based parallel collision detection for real-time motion planning," in *Algorithmic Foundations of Robotics IX*, D. Hsu, *et al.*, Eds., Springer Berlin Heidelberg, 2010, pp. 211-228.

[16] J. Pan, S. Chitta, and D. Manocha, "Probabilistic collision detection between noisy point clouds using robust classification," in *Robotics Research*, H. Christensen and O. Khatib, Eds., Springer, Cham., 2017, pp. 77-94.

[17] M. Muradi and R. Wanka, "Sample-based motion planning for multi-robot systems," in *Proc. 6th International Conference on Control, Automation and Robotics*, Singapore, 2020, pp. 130-138.
[18] Wikipedia. Graphics processing unit. [Online]. Available: https://en.wikipedia.org/wiki/Graphics_processing_unit
[19] H. Noborio, T. Kunii, and K. Mizushino, "GPU-based shortest distance algorithm for liver surgery navigation," in *Proc. the 10th Anniversary Asian Conference on Computer Aided Surgery*, Fukuoka, 2014, pp. 42-43.
[20] H. Noborio, T. Kunii, and K. Mizushino, "Omni-directional shortest distance algorithm by complete parallel-processing based on GPU cores," *International Journal of Bioscience, Biochemistry and Bioinformatics*, vol. 8, no. 2, pp. 79-88, 2018.
[21] T. Li, T. Bolkart, M. J. Black, and J. Romero, "Learning a model of facial shape and expression from 4D scans," *ACM Transactions on Graphics*, vol. 36, no. 6, article 194, 2017.
[22] J. Romero, D. Tzionas, and M. J. Black, "Embodied hands: Modeling and capturing hands and bodies together," *ACM Transactions on Graphics*, vol. 36, no. 6, article 245, 2017.
[23] O. Hirose, "A Bayesian formulation of coherent point drift," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 7, pp. 2269-2286, 2020.

Hiroshi Noborio received the B.Eng. and M.Eng. in Department of Computer Science from Shizuoka University in 1982 and 1984 respectively. In succession, he received the Ph.D. in Department of Mechanical Engineering, Faculty of Engineering Science, Osaka University in 1987. After getting Ph.D., he continued to work the same department as assistant professor. In 1988, he moved to the Department of Precision Engineering, Osaka Electro-Communication University. From April 1996 to March 1997, he received Humboldt-Fellowship and worked at Lehrstuhl fur Steuerungs- und Regelungstechnik, Technische Universitat Munchen. Right now, he is a professor in the Department of Computer Science.

His research interests have spanned various areas of robotics, CV (computer vision), CG (computer graphics), XR (VR+MR+AR). Prof. Noborio has been developing novel methodologies in the sensor-based and model-based navigation such as path-planning of mobile robot and motion-planning of robotic manipulator. Lately Prof. Noborio has started novel projects in the developments of surgical guiding of dental implant and surgical simulation/navigation of Liver, Kidney and Brain.