International Journal of
Applied Mathematics,
Electronics and Computers
ISSN: 2147-8228 www.atscience.org/IJAMEC
Advanced Technology and Science
Original Research Paper

Haptic Interaction with Deformable Objects Designed Using a Hand-drawn Sketch

Masakazu Yamaoka, Shun Ido *

Accepted 15th August 2014

Abstract: Deformable objects are used in a wide range of fields including medicine and food processing. Deformable objects simulations can be used to investigate many different kinds of objects that that have genuine relevance to the real world. The problem, however, is that for inexperienced novices in computer graphics it can be difficult to represent complex shapes. In this study, we propose a method of handling deformable objects through which a user can easily create such objects using hand-drawn sketches, and that also incorporates haptic interaction.

Keywords: *Virtual reality, Deformable object simulation, Solid generation, Hand-drawn sketch, Haptic interaction.*

1. Introduction

Since many objects in the real world can be classified as deformable objects, their simulation in a virtual world has applications in a wide variety of fields such as: medicine, where it can be used for organ research; in food processing, for the simulation of jelly. As such, much research has been conducted in this field.

Hirai simulated rheological objects, which are a sub-type of deformable objects, via three-element models in real time [1], and Tanaka applied adaptive mesh refinement to real time simulations of deformable objects [2]. The free-form deformable objects investigated in this research, however, are difficult to handle, because their modeling involves constructing polygons directly by controlling vertices and control points. Therefore, it is difficult for computer graphics novices to create complex shapes such as organs.

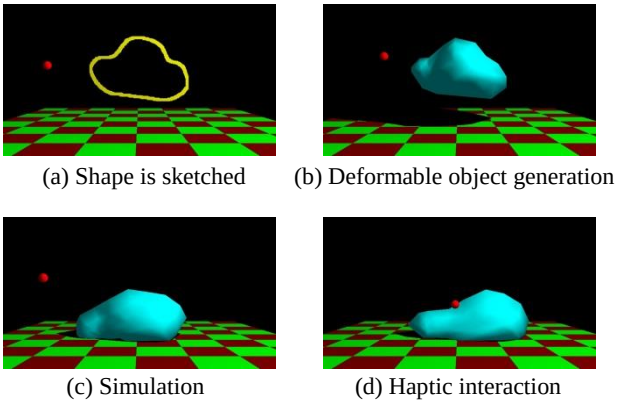
One example from the literature of an attempt to reduce the difficulty of making objects of complex shapes was presented by Igarashi, who developed Teddy [3], a modeling system employing hand-drawn sketches. In Teddy, a user can easily create solids, without any prior knowledge or experience in modeling; however, the created solids are not actually deformable objects, and accordingly, it can not be used for deformable object simulations.

In this paper, in order to easily create and simulate free-form deformable objects, we translated the solids made using Teddy into rheological objects, which can express various objects. We also integrated haptic feedback, i.e., the ability to touch the deformable objects, into the system to improve interactivity. This system allows for the interaction with variously shaped rheological objects, and also makes it easy to construct virtual worlds. Therefore, it is expected to be used in a wide variety of fields, such as surgical simulation and for amusement.

Graduate School of Science and Engineering, Ehime University, 7908577, Ehime/Japan
** Corresponding Author: Email: ido@cs.ehime-u.ac.jp*
 # This paper has been presented at the International Conference on Advanced Technology&Sciences (ICAT'14) held in Antalya (Turkey),

2. System Outline

To allow the easy simulation of deformable objects by anyone, we propose a simulation in which deformable objects are generated using a hand-drawn sketch as per the Teddy system developed by Igarashi, which works as follows. First, users draw the contour of the deformable object using a mouse. Second, a solid based on the contour is generated. Finally, the deformable object generated by translating the solid is simulated. Figure 1 shows the construction of, and interaction with, the deformable objects in this system.



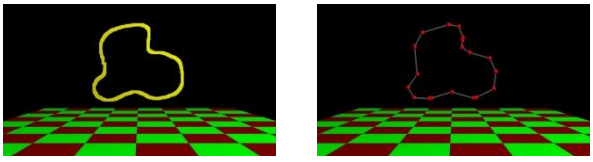
(a) Shape is sketched (b) Deformable object generation

(c) Simulation (d) Haptic interaction

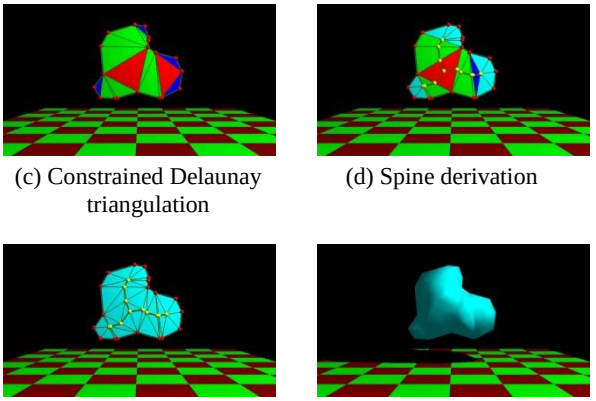
Figure 1. System stages of the construction of, and interaction with, a deformable object

3. Solid generation

The process flow used to generate the solid from the hand-drawn sketch is shown in Figure 2 [3].



(a) Line drawing (b) Sampling



(c) Constrained Delaunay triangulation (d) Spine derivation

(e) Final triangulation (f) Solid generation

Figure 2. Procedures for generating solids

3.1. Sampling

Sampling utilizes the image processing of a binarized image, and this section explains the sampling process in greater detail.

3.2. Thinning

In order to easily locate the sampling points on the drawing line, as the name suggests, thinning defines where the points are located by making the line thin. There are several thinning algorithms available; the Hilditch algorithm [4] was chosen on account of its speed advantage.

3.3. Elimination of extra line

Even after thinning, there is the possibility that an "extra line", which is the bottleneck of the sampling process, remains, and, thus, must be deleted. Figure 3 shows this process.

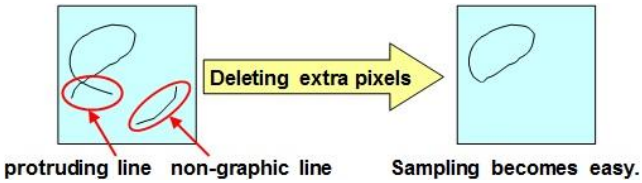


Figure 3. Elimination of the extra line

In the process of deleting this line, the value of the pixel whose connectedness is low is changed during raster scanning. This research calculates the pixel connectedness by employing the 8-neighbor concept. Figure 4 shows the patterns for deleting extra pixels.

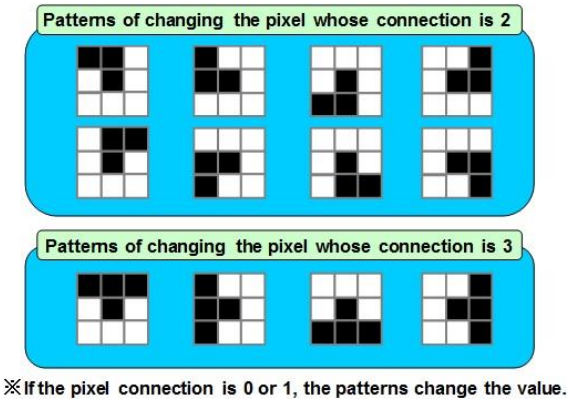


Figure 4. Patterns for deleting extra pixels

If the connectedness of a pixel is 0, it is noise and, thus, is deleted. A pixel with connectedness of 1 is also deleted, because it does not have the function of connecting other pixels. If the connectedness is 2 or 3, only specific patterns, such as those shown in Figure 4, are deleted, for the same reason. If any pixel value is changed, a raster scan of the whole image is performed again.

3.4. Vectorization of line drawing

To adequately capture certain features of the drawing's lines, such as curvature, this research samples the line drawing by vectorization. Figure 5 illustrates this process.

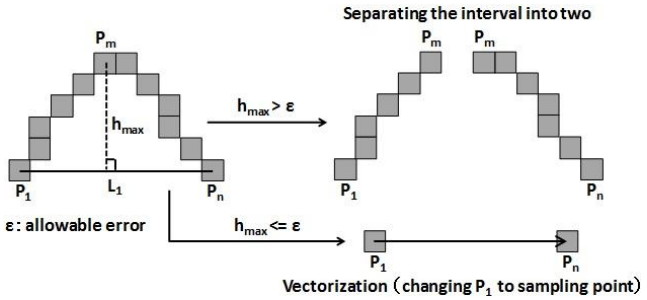
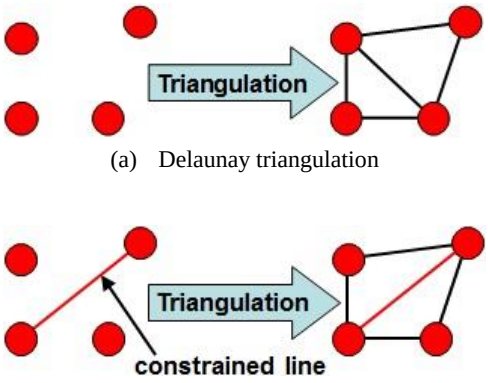


Figure 5. Vectorization of the line drawing

First, the segment L_1, which connects the start point P_1 and the end point P_n, is created. Second, the value h_{max}, which is the maximum distance between all line drawing pixels $P_i(1 < i < n)$ and the segment L_1, is calculated. The error calculated for any line drawing pixels and the segment L_1, consisting of the pixels from P_1 to P_n, is lower than ε, therefore if h_{max} is lower than the allowable error ε, a sampling point is put at P_1. On the other hand, in the case where $h_{max} > \varepsilon$, the line drawing from P_1 to P_n is segmented, consisting of segment P_1 to P_m remains mostly distant from P_1, and segment P_m to P_n. Applying the same process to these segmented lines eventually results in polygonal lines, and their distance from any line drawing pixel is lower than ε.

3.5. Constrained Delaunay triangulation

Constrained Delaunay triangulation is conducted to connect sampling points with edges. Delaunay triangulation is triangulation in which the circumcircles of each triangle generated from a point group do not include the other points of the triangle. Constrained Delaunay triangulation is Delaunay triangulation using constrained lines that must become the edges of the triangle. For example, Figure 6(a) shows the result of Delaunay triangulation, and Figure 6(b) shows the result of constrained Delaunay triangulation, in which a constrained line has been added to the point group of figure 6(a). A comparison of the two figures illustrates the difference.



(a) Delaunay triangulation

(b) Constrained Delaunay triangulation

Figure 6. Comparison of Delaunay triangulation methods

The triangles in constrained Delaunay triangulation are classified

within three categories: triangles without external edges (junction triangle), triangles with one external edge (sleeve triangle), and triangles with two external edges (terminal triangle). Figure 7 shows this triangle classification system.

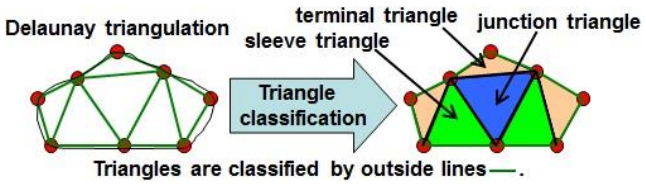


Figure 7. Triangle classification for constrained Delaunay triangulation

To perform constrained Delaunay triangulation, simple Delaunay triangulation is first generated from a point group. Then, after selecting one of the constrained lines, Delaunay triangulation is carried out on the triangles that cross this line. Figure 8 illustrates this process.

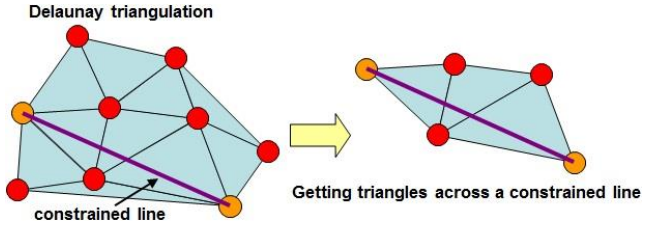


Figure 8. Acquisition of triangles across a constrained line

To evaluate the triangles across a constrained line, the system inspects the edges that cross the constrained line, and then selects triangles made up of such edges. Next, it divides the points of the triangles according to their location with respect to the constrained line. After this, having added two points from the constrained line to the divided point groups, Delaunay triangulation is carried out on each of the latter. The reason for dividing points is that some of triangles in the original Delaunay triangulation may contain the constrained line, and thus any constrained line in the original Delaunay triangulations used to generate the divided point groups must become an external edge. A point's location relative to the constrained line is indicated by the sign of z in eq. (1),

$$z = y - a * x - b \tag{1}$$

Where a is the gradient of the constrained line, b is the intercept of the constrained line, and x and y are the XY coordinates of the point. If the area consisting of the triangles across a constrained line are divided into two point groups, all triangles across the constrained line are deleted. Figure 9 shows the area consisting of points divided by a constrained line.

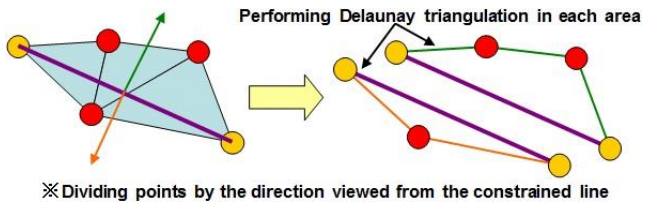


Figure 9. Classification of points using a constrained line

Constrained Delaunay triangulation is conducted first by performing Delaunay triangulations in each area; however, there is a problem with this process: there is the possibility of failing to perform a Delaunay triangulation if the divided area is a concave polygon. This is because the triangles that consist of both the points outside and inside the divided area overlap with the triangles that consist of points only within the divided area. To remedy this, one of the overlapping triangles is deleted by determining which is inside and outside (from the view of the polygon of the divided area) using the external edge of the divided area. By serially applying this process to all constrained lines, constrained Delaunay triangulation can be successfully performed.

4. Conversion of solids into deformable objects

In order to maintain the deformable object shape during deformable object simulation, calculations are required wherein the whole solid is divided into tetrahedrons; however, the solids generated in Teddy do not perform full tetrahedral division. As such, this research performs tetrahedral division of the solid from final triangulation according to the triangle type. Figure 10 shows the triangle types of the final triangulation and the solid generated from them.

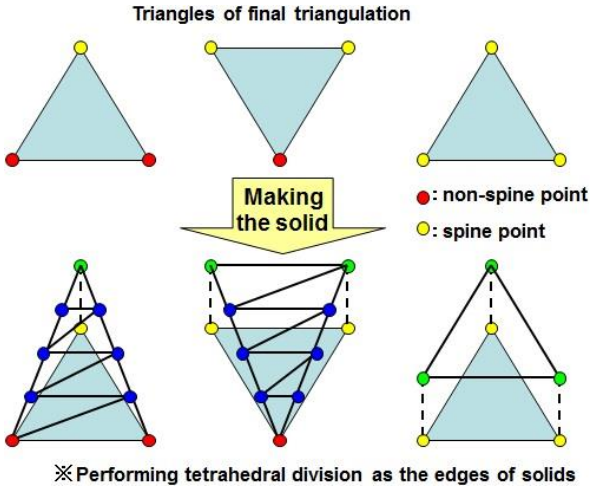


Figure 10. Triangle types in the final triangulation and the solid generated from them

In Figure 10, the triangle types are determined by the number of triangle points of the spines, which are the edges made up of the points elevated for the purpose of making the solid. If the final triangulation is converted to a solid, the triangles are split into a pair of top and bottom triangles. One is the inside triangle of the solid, the other is the surface triangle of the solid, and two edges of the surface triangle contain several points that expand the solid into an ellipse.

The construction method of the tetrahedrons for each triangle type is as follows. For tetrahedral division of triangles with one spine point, the edges connecting the point are used to expand the solid into an ellipse such that the spine point is inside the solid. Afterwards, tetrahedral division is carried out by pairing the triangles made by dividing the surface triangle with the spine point inside the solid. Figure 11 shows the tetrahedral division of a triangle with one spine point.

The tetrahedral division of triangles with two spine points is performed as follows. The edge of the tetrahedron is constructed, and the points on the base to the right and left of the non-spine points inside the solid are separated. The edges connecting the points that expand the solid into an ellipse using the spine points inside the solid are first generated for the right and then the left

side, and then the edges are generated connecting pairs of left and right side spine points inside the solid. Figure 12 illustrates this process.

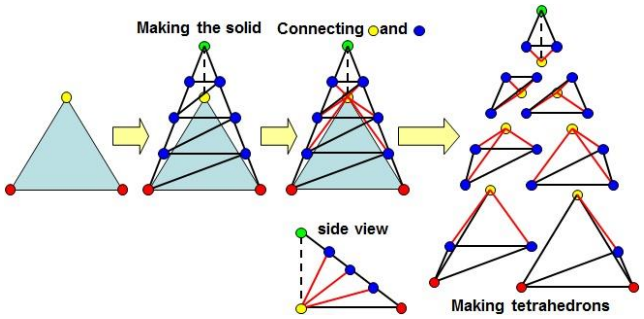


Figure 11. Tetrahedral division of a triangle with one spine point

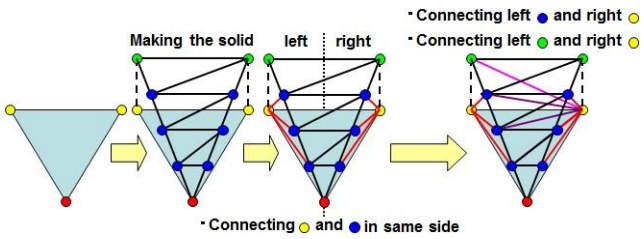


Figure 12. Construction of the edges in a triangle with two spine points

For tetrahedral division of the solid of the triangle with two spine points, the solid is sliced into rings along the edges and tetrahedral division is performed for each sliced layer. Figure 13 illustrates this process.

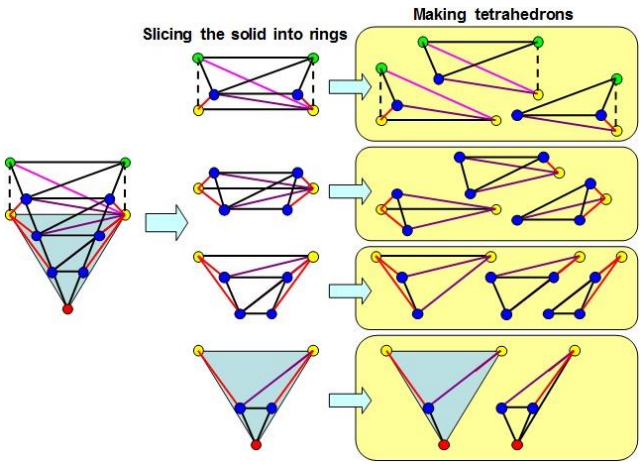


Figure 13. Tetrahedral division of a triangle with two spine points

Tetrahedral division of triangles with three spine points follows a different process to the other triangle types. First, a point is created in the solid in order to split it. The reason for the addition of this point is that, when compared with the edges generated from triangles with two spine points, the edges of the tetrahedrons generated from triangles with three spine points cross over other edges. The new edges for these triangles are made by connecting the points of the top and bottom triangles, including the triangle generated during the making of the solid, with the point located equidistant from the triangle points. Figure 14 illustrates this process.

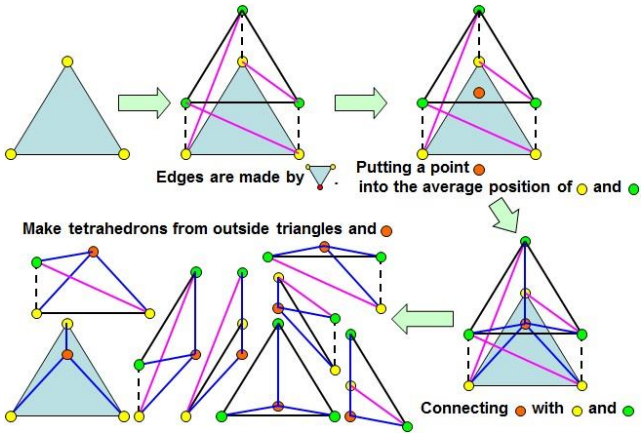


Figure 14. Tetrahedral division of triangles with three spine points

5. Haptic device

To enhance interactivity with rheological objects, this research employed a haptic device, SPIDAR-G, to incorporate the haptic sense, which is the force that acts on the pointer controlled by the user. SPIDAR-G is a haptic device developed by Satou Lab at the Tokyo Institute of Technology. In SPIDAR-G, haptic feedback and control over movement in six degrees of freedom and spin are obtained by detecting the position and pose of the grip by the eight strings and the eight motors [5]. Figure 15 shows the device.

Figure 15. SPIDAR-G

6. Rheological object simulation

To simulate rheological objects, the edge of the generated solid was translated into a three-element model, which is the smallest mechanical model able to represent rheological objects. The properties of the rheological objects were primarily expressed using reference [1].

6.1. Three-element model

The three-element model [1] is a mechanical model able to capture the deformation of rheological objects,
and requires the fewest number of springs and dampers,
as shown in Figure 16.

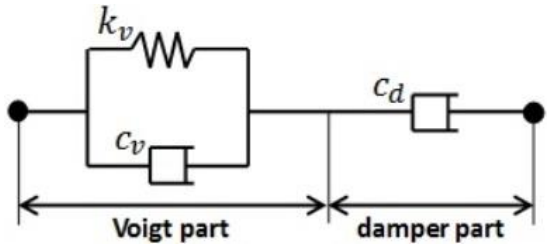


Figure 16. Three-element model

The force of the three-element model is calculated according to

eq. (2).

$$\dot{l}_k = \frac{(x_j - x_i)\cdot(v_j - v_i)}{l_k}$$

$$\dot{a} = \frac{-k_v(al_k - L_k) - (c_v a - c_d(1-a))\dot{l}_k}{(c_v + c_d)l_k} \quad (2)$$

$$f = c_d\left((1-a)\dot{l}_k - \dot{a}l_k\right)e$$

Where x_i and v_i are respectively the position and the velocity of the mass point i, a is the length rate of the Voigt part in the length of the three-element model, l_k is the length of the three-element model, L_k is the natural length of the Voigt part, k_v is the elasticity of the Voigt part, c_v is the viscosity of the Voigt part, c_d is the viscosity of the damper part, e is the unit vector of the position from the mass point i to the mass point j, and f is the force generated by the three-element model.

6.2. Force-dependent nonlinear damper

The deformation of the three-element model continues to increase or decrease as long as force is applied, and, therefore, if rheological objects are simulated under gravity, the three-element model will continuously deform. The problem of maintaining the "normal" shape of rheological objects thus arises. In order to solve this problem, a force-dependent nonlinear damper [1] is applied to the damper part. The force-dependent nonlinear damper changes the viscosity of the damper part according to the force applied to the three-element model, which is calculated according to eq. (3).

$$c_d = \begin{cases} c_{max} & (f < f_{min}) \\ Ae^{-Bf} & (f_{min} \le f \le f_{max}) \\ c_{min} & (f > f_{max}) \end{cases}$$

$$c_{max} = Ae^{-Bf_{min}} \quad (3)$$

$$c_{min} = Ae^{-Bf_{max}}$$

Where c_d is the viscosity of the damper part, f is the force generated by the three-element model, f_{min} is the damper force required for the elements to begin working normally, f_{max} is the total force after the additional force has been added to the three-element model, A and B are constants, c_{max} is the maximum viscosity, and c_{min} is the minimum viscosity.

6.3. Maintenance of shape

Rheological objects behave such that volume change is small when an external force is applied. Furthermore, in order to continue normal simulation, it is important that the rheological objects are not broken by a strong external force. Rheological objects built using a truss structure break easily under the large volume change induced by the addition of a strong external force. Thus, to solve this problem, spring models are employed to maintain the area and volume of the rheological objects. Using the method used to maintain the deformable object's shape, we explain the global volume constant condition [6], which calculates the force required to maintain the shape for the whole volume of the deformable object, and also the method used to maintain each surface area and each tetrahedron volume [7].

6.4. Global volume constant condition

Rheological objects behave such that the difference of changing the volume is small. Thus, to express this property, a global volume constant condition is applied [6], which maintains the virtual rheological object's volume by generating force at certain points via springs and dampers depending on whole volume of the rheological object. Figure 17 illustrates this condition.

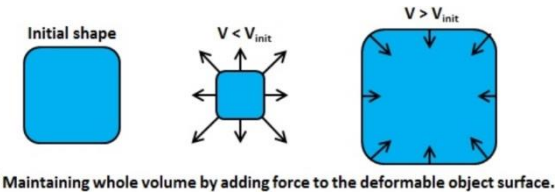


Figure 17. Generation condition of the global volume constant condition

The pressure p generated by the global volume constant condition is calculated according to eq. (4).

$$p = -K(V - V_{init}) + C\dot{V}$$

$$F_k^{face} = pS_k n_k^{out} \quad (4)$$

$$F_i^{vertex} = \sum_{k \in A_i} \frac{F_k^{face}}{3}$$

Where V is the volume of the deformable object, V_{init} is the initial volume of the deformable object, K is the elasticity, C is the viscosity, F_k^{face} is the force applied to the k th triangle face, S_k is the area of the k th face, n_k^{out} is the normal of the k th face, A_i is the set of the triangle face which the i th point belongs to, and F_i^{vertex} is the force applied to the i th point.

6.4.1. Maintenance of each surface area and each tetrahedron volume

If we maintain the rheological object shape using only the global volume constant condition, it becomes easy to maintain the total volume of the rheological object but maintaining the initial shape becomes difficult, making it troublesome to represent objects that readily maintain their shape, such as organs and jelly. Thus, to maintain the initial shape, the volume of the tetrahedrons and the area of the triangle surfaces of the rheological objects were maintained using springs [7]. The force required to maintain the area of the surface (Figure 18) is calculated according to eq. (5).

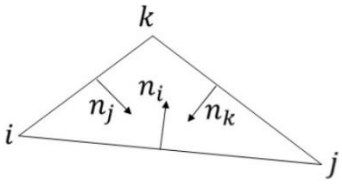


Figure 18. Surface area maintained by applied forces

$$F_i = \frac{k(S - S_{init})}{|S_{init}|}\left(n_i - \frac{n_j + n_k}{2}\right) \quad (5)$$

Where F_i is the force applied to the mass point i on the surface ijk, k is the elasticity, S is the current surface area, S_{init} is the surface area at the start of the simulation, and n_i is the unit vector which points toward the inside of the surface, perpendicular to the edge facing the mass point i.

The force required to maintain the volume of the tetrahedron (figure 19) is calculated according to eq. (6),

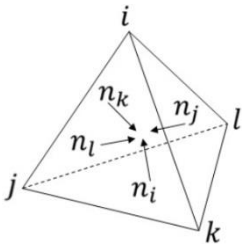


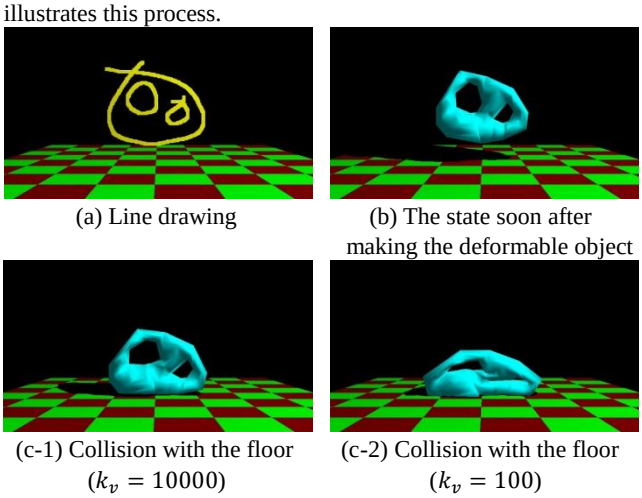
Figure 19. Tetrahedron volume maintained by applied forces

$$F_i = \frac{k(V - V_{init})}{|V_{init}|}\left(n_i - \frac{n_j + n_k + n_l}{3}\right) \quad (6)$$

Where F_i is the force adding the mass point i, k is the elasticity, V is the current volume of the tetrahedron, V_{init} is the tetrahedral volume at the start of the simulation, and n_i is the unit vector normal to the surface pointing toward i and the inside of the tetrahedron.

7. Creation example

To verify that the rheological objects designed by hand-drawn sketches can express any object shape, rheological objects which had Voigt parts of differing elasticity were made using the same hand-drawn sketches, and we confirmed the rheological object's shape when it collided with the floor after free fall. Figure 20 illustrates this process.



(a) Line drawing

(b) The state soon after making the deformable object

(c-1) Collision with the floor ($k_v = 10000$)

(c-2) Collision with the floor ($k_v = 100$)

Figure 20. The deformation of rheological objects generated from a hand-drawn sketch

Rheological objects that have Voigt parts with small elasticities (Figure 20(c-2)) undergo a greater change in their shape when they collide with the floor than those that have Voigt parts with large elasticities (Figure 20(c-1)). This demonstrates that the latter are softer, and, thus, this method can express various objects by changing the input parameters.

8. Evaluation experiment

To evaluate the usability of this system, three users made deformable objects and interacted with them through the haptic sense feature. Their feedback is reported here:

- The ease of moving the polygons changes according to point of interaction.
- Mouse operation is intuitive and easy.
- I want to see objects of greater or lesser thickness.
- It is difficult to recognize the position of the haptic pointer, and so it is difficult to interact with the deformable object.
- It would be good if we could change the parameters through the GUI.
- To easily draw objects in photos, I want to be able to insert a photo as the background when I draw.
- Error handling is required for when the processing becomes heavy during the generation of complex objects.

From the above feedback it can be seen that this system still has some room for improvement, such as developing a system for adjusting parameters, e.g., hardness and thickness, and improving the ease of haptic interaction.

9. Conclusions

This system enables a user to easily make and simulate rheological objects with different shapes. In future work, to enhance the usability of this system, we will improve the GUI and the haptic interaction method.

References

[1] Masafumi Kimura, Yuuta Sugiyama, Seiji Tomokuni, and Shinichi Hirai, "Constructing Rheologically Deformable Virtual Objects", Proc. IEEE Int. Conf. on Robotics and Automation, pp.3737-3743, Taipei, September, 2003.

[2] Huynh Quang Huy Viet, Yoshinori Tsujino, Hironori Yamashita, Yasufumi Takama and Hiromi T.Tanaka, "A Real-time Dynamic Deformable Model for Soft Tissue Simulation using Adaptive Mesh Refinement", In Proc. of Game-on Asia, 2007.

[3] Takeo Igarashi, Satoshi Matsuoka, Hidehiko Tanaka, "Teddy: a sketching interface for 3D freeform design", Proceedings of the 26th annual conference on Computer graphics and interactive techniques, pp.409-416, July, 1999.

[4] C.J.Hilditch, "Linear skeletons from square cupboards", in B. Meltzer and D.Michie eds., Machine Intelligence Vol.4, pp.403-420, 1969.

[5] Makoto Sato, "Development of String-based Force Display: SPIDAR", VSMM2002 (The Eighth International Conference on Virtual Systems and Multi Media), pp.1034-1039, 2002 9.

[6] Ryo Nogami, Hiroshi Noborio, Seiji Tomokuni, and Shinichi Hirai, "A Comparative Study of Rheology MSD Models whose Structures are Lattice and Truss", Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp.3809-3616, Sendai, September, 2004.

[7] M. Teschner, B. Heidelberger, M. Mueller, M. Gross, "A Versatile and Robust Model for Geometrically Complex Deformable Solids", Proceedings of CGI'04, Crete, Greece, pp.312-319, June, 16-19, 2004.