Toward Instant 3D Modeling: Highly Parallelizable Shape Reproduction Method for Soft Object Containing Numerous Tiny Position Trackers

Minoru Harimaya
Kyoto University
Kyoto, Japan
mharimaya@easter.kuee.kyoto-u.ac.jp

Ryo Shirai
Kyoto University
Kyoto, Japan
shirai@i.kyoto-u.ac.jp

Masanori Hashimoto
Kyoto University
Kyoto, Japan
hashimoto@i.kyoto-u.ac.jp

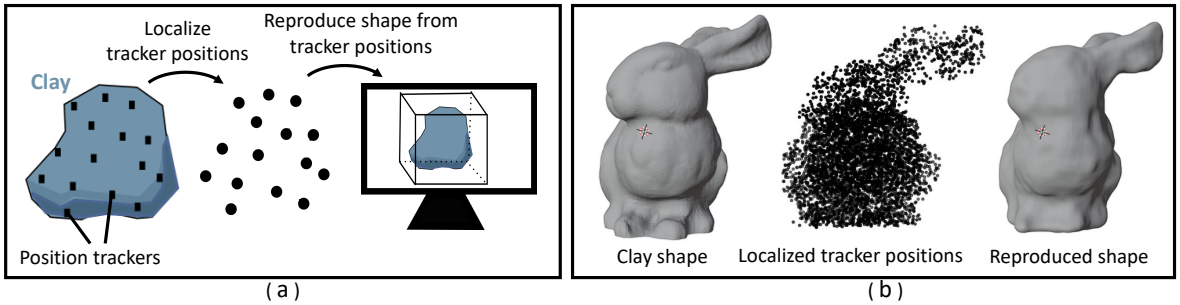

Figure 1: (a) Intuitive 3D modeling system estimates the clay shape based on the position of trackers embedded in the clay. (b) Our proposed method successfully reproduces the bunny shape only from the position information of trackers inside the original model.

ABSTRACT

This paper proposes a shape reproduction method for the clay containing numerous tiny position trackers inside to actualize a real-time intuitive 3D modeling system for non-expert users. Our method successfully reproduced 3D models reflecting the characteristics of the original models thanks to the proposed edge interpolation. Our method is highly parallelizable and even the most time consuming process requires only $O(\log N)$, where N is the number of voxels, indicating our method is potentially suitable for real-time processing.

CCS CONCEPTS

• **Human-centered computing** → **Graphics input devices**; **Information visualization**.

KEYWORDS

Instant 3D modeling, 3D shape reproduction

ACM Reference Format:
Minoru Harimaya, Ryo Shirai, and Masanori Hashimoto. 2023. Toward Instant 3D Modeling: Highly Parallelizable Shape Reproduction Method for Soft Object Containing Numerous Tiny Position Trackers. In *28th International Conference on Intelligent User Interfaces (IUI '23 Companion), March 27–31, 2023, Sydney, NSW, Australia.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3581754.3584151

IUI '23 Companion, March 27–31, 2023, Sydney, NSW, Australia

ACM ISBN 979-8-4007-0107-8/23/03.
https://doi.org/10.1145/3581754.3584151

1 INTRODUCTION

Thanks to the advancement of computer graphics technologies, applications enjoying 3D models (e.g., VR content, video games) are increasing. Therefore, research on 3D modeling methods is actively conducted to meet the increasing demand. However, currently available 3D modeling software requires expertise; namely, it cannot provide real-time intuitive modeling for non-experts. To address this issue, Ref. [5] has proposed "iClay" system. Fig. 1 shows the overview of iClay system. iClay system is supposed to provide intuitive real-time 3D modeling with the clay, in which tiny position trackers are embedded. iClay system allows non-expert users to intuitively build their 3D models only by making a 3D shape with the clay. In addition, iClay system reproduces the shape of the clay from the position information of the tracker existing not only on the surface but also existing inside the clay. Hence, iClay can even reproduce cavities inside the clay and can create a precise 3D model. iClay system requires two element technologies: tracker localization and shape reproduction from the tracker positions. Ref. [14] has developed a localization method suitable for tiny trackers but made any progress on 3D shape reproduction suitable for iClay system. As a related work, Refs. [4, 15] have proposed methods to reproduce a 2D shape from a point cloud on a 2D plane. However, extending these methods to 3D models results in huge computational cost, and hence they are not applicable to real-time 3D shape reproduction. Several methods that are based on the point clouds in 3D space have been reported [2, 7, 8]. These methods reproduce

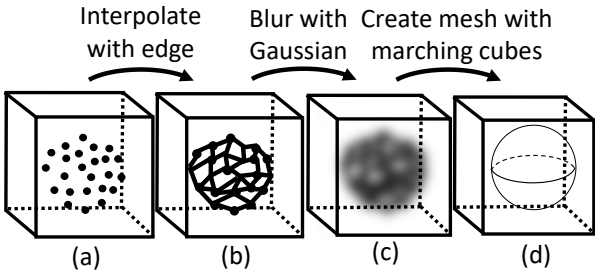

Figure 2: Process of the proposed method. (a) The proposed method initializes voxels at tracker positions with $V_{xyz} = 1$. (b) The proposed method finds pairs of trackers such that the distance between them is less than "maximum interpolation length," and sets $V_{xyz} = 1$ to voxels between them. (c) The proposed method blurs V_{xyz} with the Gaussian filter. (d) The proposed method creates meshes with marching cubes algorithm.

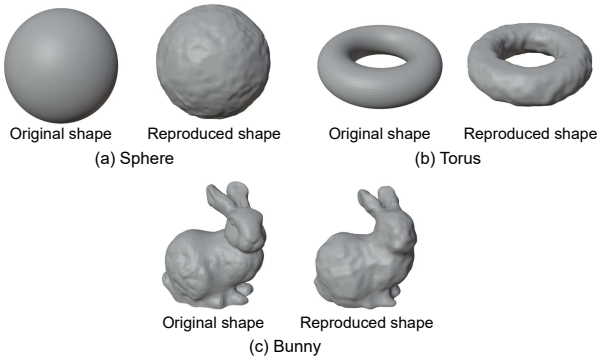

Figure 3: Shape reproduction results when maximum interpolation edge length is 1cm.

the object shape from the points cloud existing only on the object surface. On the other hand, iClay system should reproduce the 3D shape from the position information of trackers existing inside the clay. Therefore, the conventional methods cannot be applied to iClay in their original forms.

We propose a 3D shape reproduction method that uses the position information of the trackers spreading inside the clay. We demonstrate that the proposed method is highly parallelizable and is suitable for instant 3D modeling systems.

2 PROPOSED METHOD

Fig. 2 shows an overview of the proposed method. Our method first defines a 3D space, which is sufficiently large to accommodate all trackers, as shown in Fig. 2(a). Then, our method divides the 3D space into tiny small cubes called voxels. Each voxel has the same size as the tracker and holds a single scalar value V_{xyz}. Voxels that accommodate the center of any tracker are initialized as $V_{xyz} = 1$, and others are initialized as $V_{xyz} = 0$. After the initialization, the proposed method carries out the edge interpolation process searching for pairs of voxels whose distance from each other is less than a certain length, which is defined as "maximum interpolation edge length," with a breadth-first search. Then as shown in Fig. 2(b), the proposed method sets $V_{xyz} = 1$ for all voxels that lie on a straight line connecting the two voxels of the pair. To smooth the

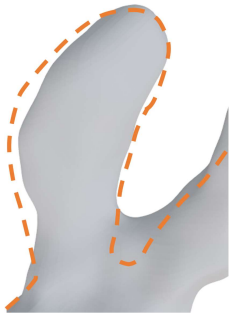

Figure 4: Difference between original shape and reproduced shape around bunny ears. Dashed line shows the outline of original shape and gray zone shows the reproduced 3D shape.

surface of the reproduced model, the proposed method further divides the existing voxels into smaller voxels. Following the voxel subdivision, as shown in Fig. 2(c), the proposed method applies a Gaussian filter to the subdivided voxels so that the values of V_{xyz} decay with the distance from the tracker. Finally, the proposed method creates meshes with "marching cubes algorithm [10]" as shown in Fig. 2(d). The marching cubes algorithm generates the surface by estimating whether each voxel, corresponding to a vertex of the marching cube, is inside or outside the clay. Whether a voxel is inside the model or not is determined by comparing V_{xyz} with a certain threshold value V_{th}; a voxel satisfying $V_{xyz} \geq V_{th}$ is considered to be inside the model and vice versa. Here, the value of V_{th} affects the volume of the reproduced model, i.e., if V_{th} is too small, the reproduced model size becomes larger than the original clay size. The proposed method utilizes the density and the number of trackers, both of them are known, to estimate the volume of the clay. Based on the clay volume, the proposed method determines the appropriate V_{th}.

Breadth-first search algorithm and Gaussian filtering are highly compatible with parallel computation. Many studies have reported that the marching cubes algorithm is also compatible with parallel computation [3, 11, 12]. Almost all processes composing our method are highly parallelizable, and hence our method potentially achieves fast 3D shape reproduction, which is indispensable for instant and real-time 3D modeling.

3 EVALUATION

This section evaluates the proposed method with benchmark 3D models. We implemented the proposed method with C++ language. Position trackers are scattered in the 3D models randomly with the density of 10 trackers per 1 cm^3. The 3D space which accommodates all trackers is a cube with 16 cm on each side. The trackers and initial voxels are both 0.1 cm on a side. Voxels are further divided into cubes of 0.02 cm on each side in a later process to smooth the surface of the reproduced model. As for blurring, we use $35 \times 35 \times 35$ Gaussian filter whose standard deviation is 17.

Fig. 3 shows the appearances of the original and reproduced models with our method. We evaluated two simple models, sphere and torus, and one complex model, Stanford Bunny [1]. Fig.3 indicates that our method reproduces 3D models successfully even for the complex model, reflecting the characteristics of the original

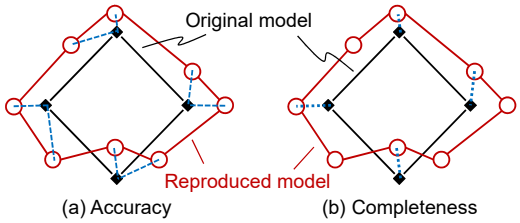

Figure 5: Calculation method of accuracy and completeness. Accuracy and completeness are symmetrical metrics. (a) To compute accuracy, for each point on the reproduced model, we measure the distance to the nearest point on the original model and take its average. (b) To compute completeness, for each point on the original model, we measure the distance to the nearest point on the reproduced model and take its average.

Table 1: The effect of the edge interpolation. The maximum interpolation edge length is set to 1cm.

	w/ edge interpolation		w/o edge interpolation	
	Accuracy [cm]	Completeness [cm]	Accuracy [cm]	Completeness [cm]
Sphere	0.0210	0.0624	3.07	0.113
Torus	0.0134	0.0602	0.345	0.108
Bunny	0.134	0.107	0.966	0.114

shape. However, there are minor differences between the reproduced and original models (e.g., the face and ears of the bunny). Fig. 4 shows a magnified view of the rabbit's ears illustrating the difference between the reproduced and original models. As Fig. 4 indicates that the region between the rabbit's ears is filled, our method tends to fill small depressions in the original model. This is an inherent problem since the density of the trackers is limited, and then our method cannot distinguish between the region where the tracker is sparse and the outside of the model. However, this problem could be mitigated by, for example, estimating the gap in the original model with machine learning, which is our primary future work.

To quantitatively evaluate the proposed method, we adopt two common metrics for evaluation: accuracy and completeness [6, 9, 13]. Accuracy and completeness are symmetrical metrics, and Fig. 5 illustrates their calculation. Both metrics indicate error distances, and hence lower these values mean better reproduction results.

First, we evaluate the contribution of the edge interpolation. Table 1 shows the metrics with and without the edge interporation. As Table 1 indicates, all metrics are greatly improved with the edge interpolation. Table 1 also shows that the benefit of the edge interpolation is greater for simple models than for the complex model. This is because the simple 3D models used in the evaluation has no bumpy region, and hence the issue of the proposed method stated above, accidentally filling the bumpy region of the original model, does not make effect.

Next, we evaluate the relationship between the the maximum interpolation edge length and two metrics. Figs. 6 and 7 show the evaluation results. The results show that accuracy is greatly improved by increasing the maximum edge length, while completeness is not. Completeness is not affected by the maximum interpolation edge length since the number of vertices in the reproduced model is far larger than that of the original model. Therefore, at least one vertex of the reproduced model would be generated at a position

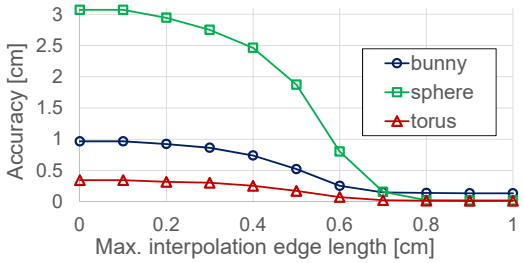

Figure 6: Relationship between accuracy and maximum interpolation edge length.

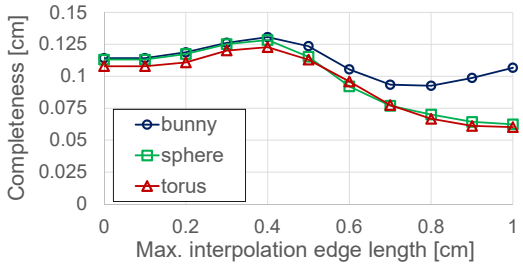

Figure 7: Relationship between accuracy constants and maximum interpolation edge length.

very close to each vertex of the original model and consequently completeness is kept low. Fig. 6 points out that accuracy does not improve any more when the maximum edge length exceeds 0.8 cm. Increasing the maximum interpolation edge length results in huge computation. Aiming at the real-time processing for instant 3D modeling, we should build an algorithm to automatically determine the appropriate interpolation edge length based on the tracker density, which is one of our future work.

Finally, we discuss the time complexity of the proposed method. When a sufficient number of processors are given ($N_p > N$) and the proposed method is run in parallel, the time complexity becomes very small, where N_p and N are the numbers of processors and voxels, respectively. The time complexity of the edge interpolation process is $O(l/r)$, where l is the maximum interpolation edge length and r is the length of a voxel edge. The time complexity of the Gaussian filtering process is $O(W)$, where W is the filter size. Determining appropriate V_{th} is the most time consuming process. However, the time complexity of determining appropriate V_{th} is only $O(\log N)$. Finally, the time complexity of marching cubes algorithm is $O(1)$ since it can be executed with voxel-level parallelism. None of the process require a complex computation, and hence our method can potentially achieves real-time processing on, for example, GPU.

4 CONCLUSION

We proposed a shape reproduction method for the clay containing numerous tiny position trackers inside. Our method reproduced 3D model successfully reflecting the characteristics of the original model thanks to the proposed edge interpolation. Theoretical discussion showed that when the sufficient number of processors are given, the time complexity of our method is very small and even the most time consuming process requires $O(\log N)$, where N is the number of voxels, which indicates that our method potentially achieves real-time processing.

ACKNOWLEDGMENTS

This work was supported by JSPS KAKENHI Grant Numbers JP21K21284 and JP22K17867.

REFERENCES

[1] Cited 01/2023/09. The Stanford 3D Scanning Repository. http://graphics.stanford.edu/data/3Dscanrep/.

[2] Matthew Berger, Andrea Tagliasacchi, Lee Seversky, Pierre Alliez, Joshua Levine, Andrei Sharf, and Claudio Silva. 2014. State of the art in surface reconstruction from point clouds. *Eurographics 2014-State of the Art Reports* 1, 1 (2014), 161–185.

[3] Christopher Dyken, Gernot Ziegler, Christian Theobalt, and Hans-Peter Seidel. 2008. High-speed marching cubes using histopyramids. In *Computer Graphics Forum*, Vol. 27. Wiley Online Library, 2028–2039.

[4] Herbert Edelsbrunner, David Kirkpatrick, and Raimund Seidel. 1983. On the shape of a set of points in the plane. *IEEE Transactions on information theory* 29, 4 (1983), 551–559.

[5] Masanori Hashimoto, Ryo Shirai, Yuichi Itoh, and Tetsuya Hirose. 2017. Toward real-time 3D modeling system with cubic-millimeters wireless sensor nodes. In *2017 IEEE 12th International Conference on ASIC (ASICON)*. 1065–1068. https://doi.org/10.1109/ASICON.2017.8252663

[6] Rasmus Jensen, Anders Dahl, George Vogiatzis, Engin Tola, and Henrik Aanæs. 2014. Large scale multi-view stereopsis evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 406–413.

[7] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. 2006. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, Vol. 7.

[8] Alireza Khatamian and Hamid R Arabnia. 2016. Survey on 3D surface reconstruction. *Journal of Information Processing Systems* 12, 3 (2016), 338–357.

[9] Yiyi Liao, Simon Donné, and Andreas Geiger. 2018. Deep Marching Cubes: Learning Explicit Surface Representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[10] William E Lorensen and Harvey E Cline. 1987. Marching cubes: A high resolution 3D surface construction algorithm. *ACM siggraph computer graphics* 21, 4 (1987), 163–169.

[11] Paul Mackerras et al. 1992. A fast parallel marching-cubes implementation on the Fujitsu AP1000. (1992).

[12] Timothy S Newman and Hong Yi. 2006. A survey of the marching cubes algorithm. *Computers & Graphics* 30, 5 (2006), 854–879.

[13] Steven M Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. 2006. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *2006 IEEE computer society conference on computer vision and pattern recognition (CVPR'06)*, Vol. 1. IEEE, 519–528.

[14] Ryo Shirai, Yuichi Itoh, and Masanori Hashimoto. 2021. Make it Trackable: An Instant Magnetic Tracking System With Coil-Free Tiny Trackers. *IEEE Access* 9 (2021), 26616–26632.

[15] Safeer Babu Thayyil, Jiju Peethambaran, and Ramanathan Muthuganapathy. 2021. A sampling type discernment approach towards reconstruction of a point set in r2. *Computer Aided Geometric Design* 84 (2021), 101953.