Interactive Texture Segmentation of 3D Scanned Models Leveraging Multiview Automatic Segmentation

Koki Madono
Waseda University
Tokyo, Japan
Preferred Networks
Tokyo, Japan
madonomadonorunning@gmail.com

Takeo Igarashi
Preferred Networks
Tokyo, Japan
takeo@acm.org

Hiroharu Kato
Preferred Networks Inc.
Tokyo, Japan
hkato@preferred.jp

Taisuke Hashimoto
Preferred Networks Inc.
Tokyo, Japan
hashimotot@preferred.jp

Fabrice Matulic
Preferred Networks Inc.
Tokyo, Japan
fmatulic@preferred.jp

Tsukasa Takagi
Preferred Networks Inc.
Tokyo, Japan
takagi@preferred.jp

Keita Higuchi
Preferred Networks
Tokyo, Japan
khiguchi@acm.org

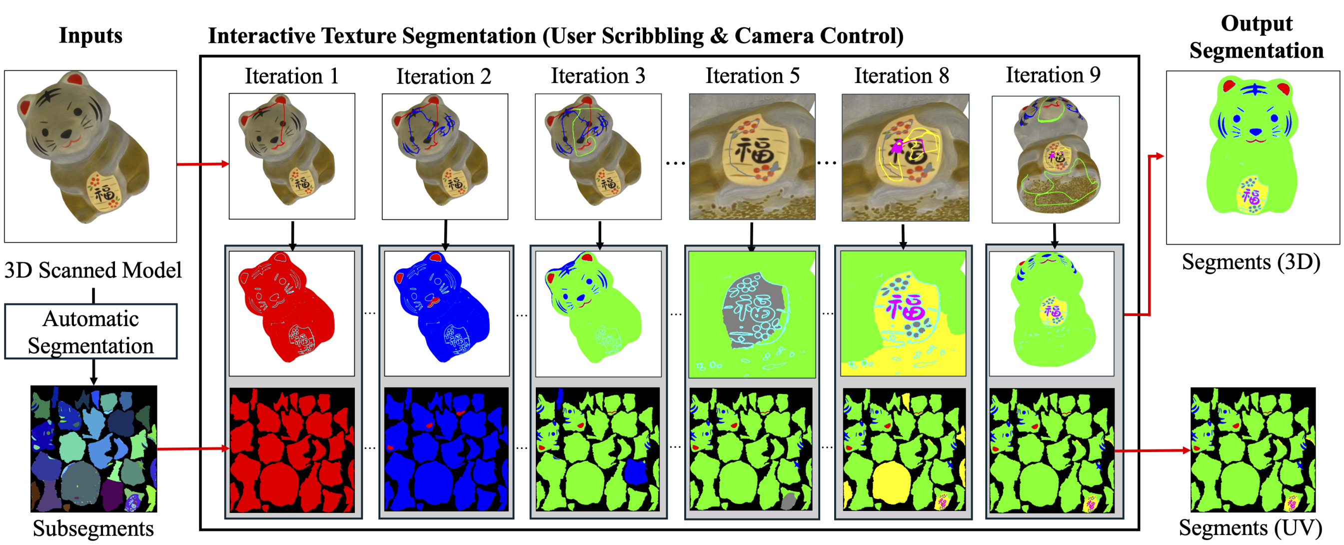

Figure 1: Workflow of our interactive texture segmentation using a 3D scanned model. Automatic segmentation divides the texture into small subsegments in a pre-processing step. The user then iteratively classifies the subsegments into segments by drawing scribbles on the 3D model. The segment ID of a subsegment is set to the ID of the nearest stroke.

Abstract

In 3D model scanning, the raw texture of a 3D model often requires segmentation into distinct regions to apply different material properties to each region. Current methods, such as manual segmentation, are labor-intensive, while automatic segmentation techniques lack user control. We propose an interactive tool that combines automatic segmentation with minimal manual intervention, striking an optimal balance between efficiency and control. Following a multiview automatic segmentation process that divides the texture into small subsegments, users cluster the subsegments into segments by drawing simple scribbles in the 3D model view. We show

CHI EA '25, Yokohama, Japan

ACM ISBN 979-8-4007-1395-8/25/04
https://doi.org/10.1145/3706599.3719697

in a user study that our approach improves segmentation accuracy and quality compared to manual segmentation with standard 3D computer graphics software. This research paves the way to more efficient texture segmentation in 3D model scanning.

CCS Concepts

• **Human-centered computing → Graphical user interfaces**; • **Computing methodologies → Computer graphics**; **Image segmentation**.

Keywords

Interactive Framework, 3D Mesh Segmentation

ACM Reference Format:
Koki Madono, Takeo Igarashi, Hiroharu Kato, Taisuke Hashimoto, Fabrice Matulic, Tsukasa Takagi, and Keita Higuchi. 2025. Interactive Texture Segmentation of 3D Scanned Models Leveraging Multiview Automatic Segmentation. In *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems (CHI EA '25), April 26–May 01, 2025, Yokohama, Japan.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3706599.3719697

1 Introduction

Advancements in computer technology have made high-quality 3D scanning accessible to a wide range of users. Modern 3D reconstruction techniques allow for the creation of highly accurate models with textures from photographs. As the use of 3D models becomes more prevalent, post-processing becomes essential to clean up and customize these models so that they meet specific user needs, such as altering colors, modifying materials, and reshaping parts. Unlike 3D objects created with modeling software such as Blender [1] and Zbrush[2], the user has very little control over the initial structure of 3D models obtained from scans, which often lack clean segments corresponding to meaningful subcomponents. Consequently, users must segment these rough 3D models in post-processing to enable subsequent editing of models' parts and their properties, such as material and texture attributes, for later use in 3D applications.

A straightforward method for segmenting 3D models involves users manually partitioning the texture using 3D software. For example, Blender offers basic 3D painting functionality, such as paint brushes and fill tools, which can facilitate 3D model segmentation. However, this manual segmentation process is labor-intensive and requires using these 3D painting tools with meticulous precision.

On the other hand, automatic segmentation solutions aim to eliminate the need for user involvement. However, heuristic techniques such as k-means clustering [11] and superpixel segmentation [1] struggle to accurately identify useful segments based on both 3D shape and texture information. More advanced techniques using deep learning demonstrate superior capabilities in automatically extracting meaningful segments [4, 7, 16], but these methods do not guarantee results that align with users' needs or application-specific requirements. Despite these technological advancements, the wide range of user preferences and the subjective nature of ideal segmentation present challenges for solutions solely relying on automated processes. This highlights the need for adaptable, user-driven techniques to adequately address the requirements of applications involving 3D scanned models.

We propose a semi-automatic interactive approach to texture segmentation, addressing the limitations of both fully manual and fully automatic methods. Our technique aims not only to reduce the labor-intensive efforts associated with manual segmentation but also to give users greater control over automatic processes. Additionally, efficient manual input is crucial for the creation of annotated datasets to improve automatic segmentation. Given the diverse needs and preferences of users, even for the same 3D model, supporting flexible, tailored and effective segmentation is essential to produce high-quality 3D models.

Our technique first involves pre-computing texture image segmentations and multiview-rendered image segmentations of the 3D scanned model using geometric features that are not contained in the texture image. From these results, boundaries are extracted and textures are split into small subsegments, which serve as segment candidates. In a subsequent interaction phase, users add color scribbles on the 3D view of the scanned model to guide the subsegment-selection process for segmentation. Specifically, when a color scribble is added, a segmentation ID is assigned to subsegments intersected by the scribble, and the ID is propagated to nearby subsegments to generate a segmentation. This process is repeated with the user adding or removing scribbles to iteratively refine the segmentation according to their needs.

Our study reveals that our method offers superior segmentation accuracy compared to traditional manual painting interfaces in terms of quality and user experience.

[1]https://www.blender.org
[2]https://www.maxon.net/ja/zbrush

2 Related Work

Various interactive segmentation techniques have been proposed.

Heuristic-based approaches have often been used for interactive segmentation. For example, Forlines et al. [3] propose a segmentation technique to guide human's visual search, Igarashi et al. [5] propose an interactive 3D volume segmentation approach using value thresholding, Nigolian et al. [12] adapt graph structure tools for interactive 3D medical segmentation, and Kawabe et al. [6] present a user interface that uses the k-means algorithm to segment material surfaces with sliders to allow the user to adjust hyperparameters. Heuristic methods can be easily adapted to several types of input data, however finding good hyperparameters is difficult and in some cases, accuracy is quite low.

To address the limitations of heuristic approaches for interactive segmentation, machine learning techniques have been considered. For example, Wang et al. [15] propose a scene segmentation approach for surgery learning experience, Style2fab [2] enables interactive personalized fabric 3D models creation via generative AI, Zhou et al. [17] use interactive machine teaching segmentation via user gestures in videos, and Kim et al [8] use AI assistance for human-in-the-loop 3D semantic segmentation. Furthermore, several prompt-based approaches have been proposed in the interactive segmentation literature [9] [13] [14] [10]. These approaches improve the performance of segmentation accuracy, however recent machine learning-based automatic segmentation models can struggle to segment small parts of textures, which can lead to undesirable results for the user.

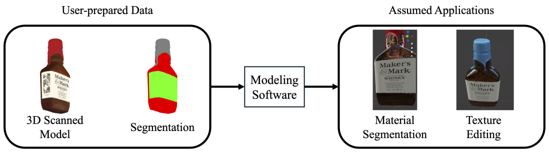

Figure 2: Applications for editing 3D scanned models using segmentation information.

In practical workflow, 3D paint software like Blender[1] and ZBrush[2] enhance interactivity by integrating numerous convenient tools within their interfaces, e.g. bucket and brush tools to directly paint on the UV map. These interfaces further provide a dual view of both the 3D model and the UV map during segmentation to facilitate painting operations. However, these solutions require a certain level of practice and proficiency, potentially making them less accessible to novice users.

To achieve easy-to-use efficient interactive segmentation, we propose an approach that first segments the texture into a multitude of small subsegments and then gradually merges these regions based on user-provided scribbles. This approach strikes a balance between automation and user control, allowing for precise and intuitive segmentation.

3 User Interface

3.1 Target Problem Setting

Our research question is how to leverage the efficiency of modern automatic segmentation algorithms, while retaining some of the flexibility and control afforded by manual techniques, to segment 3D scanned models for which 3D mesh and UV texture data are available. The resulting segmentation data can then be used to edit material and texture properties of these 3D models as illustrated in Figure 2.

While manual segmentation affords full control to precisely assign class labels to each UV pixel, this process is tedious and time-consuming. In practice, certain areas lend themselves to straightforward classification into the same category due to uniform colors or shapes. Conversely, other regions with non-uniform textures present challenges.

To mitigate the need for manual labor, we propose a hybrid segmentation approach: automating the segmentation of easily identifiable regions and enabling user-guided segmentation for more complex areas. To achieve this, we decompose the UV texture into subsegments of varying sizes and merge these subsegments into segments based on scribbles (strokes) subsequently input by the user. This approach aims to streamline the texture segmentation process, enhancing flexibility and efficiency when handling 3D scanned models.

3.2 User-Guided Segmentation Workflow

Our segmentation technique, illustrated in Figure 3, consists mainly in two steps:

(1) Preprocessing: Automatic subsegment generation and computation of subsegment distance table
(2) User-Guided Segmentation: The user iteratively adds scribbles on the UI, upon which segmented regions are updated, until the results are satisfactory.

We assume that in the preprocessing stage, subsegments and subsegment distance table have been computed which is used on further user-guided segmentation. During the preprocessing stage, the user draws a scribble and confirms the 3D segmentation results. The user can repeatedly draw scribbles until they are satisfied with the outcome. In this process, subsegments and a subsegment distance table are utilized to enable faster and more accurate segmentation through heuristic computation. The details of the two steps are illustrated in supplementary materials.

3.3 Design Rationale and Tools

Figure 4 shows a screenshot of our interactive texture segmentation interface. Our system aims to be straightforward and intuitive for novice users and therefore we create simple and intuitive tools for our interface.

The UI consists mainly of two panels showing the 3D model with its original texture (Textured View) and the model with the current color-coded segmentation results (Segmentation View) (Figure 4). The two views can be simultaneously panned, rotated and scaled using shift bottom, the camera control button on the right, and mouse respectively. After choosing a color representing the ID of a particular region to segment, the user draws short strokes (scribbles) with the mouse on the corresponding parts of either view (Textured or Segmentation View). In the background, the system assigns the selected segment ID to the subsegments intersected by the scribble and then propagates this segment ID to neighboring subsegments to generate a segmentation. The Segmentation View is subsequently updated to reflect the new segmentation. This procedure is repeated, where the user chooses a different color to mark other regions of the model. At each stage, the user can change the size of the brush as well as undo/redo operations to facilitate the scribbling process. When the user is satisfied with the results, they can save the UV map (to be used in a 3D application to further edit material and texture properties).

3.4 Segmentation Example

Figure 1 shows an example of the interactive segmentation process with a figurine. The process begins with a model obtained from scanning the physical figurine and computing subsegments using the algorithm. After subsegmentation is complete, the user adds scribbles with the interface to perform the actual segmentation. In our example, the user starts by adding red scribbles to cluster the first set of features of the figurine: its mouth and ear. Since these are the initial scribbles, all subsegments are colored in red. Next, the user draws blue scribbles for the black elements of the figurine, which triggers a recalculation of segment boundaries using the algorithm. After this second iteration, the mouth and ear remain grouped under the same category (and therefore in red color), while the face surface becomes a distinct blue segment. This segmentation means the mouth and ear are still close to the previously drawn red scribble. The user then proceeds to add further scribbles in different colors for the other regions of the object until they are satisfied with the results.

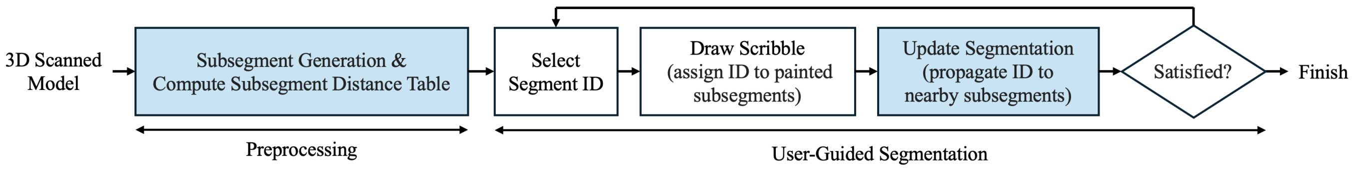

Figure 3: Overview of segmentation workflow on our interface. The user iteratively provides scribbles to the 3D model and evaluates the automatic segmentation results until achieving satisfaction. The user workflow part is indicated by white box and machine workflow is indicated by a blue box.

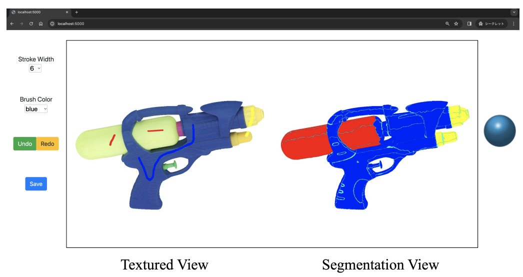

Figure 4: Screenshot of our system. It consists of brush adjustment (brush size, color), undo/redo operation, save operation of UV map, and camera control. On the main screen, the user can simultaneously check both textured mesh and segmentation mesh which always visualize the boundaries by light blue color. Once the scribble is added to a textured view, the segmentation results are updated. In the Figure's example, the user added three scribbles (red, blue, yellow), from which the segmentation is automatically estimated.

With this iterative process, users can progressively refine the segmentation, while leveraging the system's automatic segment clustering. At any time, the user can rotate the camera view of the 3D model to confirm and adjust the current segmentation for all parts of the model.

4 Implementation

4.1 Preprocessing

4.1.1 Automatic Segmentation of UV texture map. We assume a 3D scanned model and corresponding UV texture as input for the preprocessing stage. The UV texture map consists of disconnected patches. Leveraging the availability of UV texture maps, we apply a 2D segmentation model to the UV map to obtain a more comprehensive segmentation map. The UV texture is segmented through SAM-HQ [7] as follows:

$$S_{UV} = \mathrm{SAM}\left(T_{UV}\right) , \tag{1}$$

where S_{UV} denotes a segmentation extracted from the UV texture, *SAM* denotes a 2D segmentation model, and T_{UV} denotes a UV texture.

4.1.2 Automatic Segmentation of Rendered Images. In 3D surface segmentation, both geometry and texture are crucial components. As mentioned in Section 2, 2D segmentation models typically demonstrate superior accuracy and flexibility compared to their 3D counterparts. However, simply applying 2D segmentation to texture images is insufficient because it cannot capture geometric features. Therefore, we apply 2D segmentation to multiview-rendered images to improve the quality of subsegment generation.

We first prepare the cameras $\boldsymbol{C}^*$ which cover the entire 3D surface of a mesh. Then, we conduct camera view rendering that obtains rendered images I_{C^*} of each camera view. Then, we compute the segmentation results as follows:

$$S_{I_{c_i}} = \mathrm{SAM}\left(I_{\boldsymbol{c}_i}\right) , \tag{2}$$

where $\boldsymbol{c}_i$ denotes i-th camera in the optimal set of cameras $\boldsymbol{C}^*$, $S_{I_{c_i}}$ denotes the segmentation result of i-th camera view's image $I_{\boldsymbol{c}_i}$, and SAM denotes the segmentation model (specifically SAM-HQ [7] in our case)..

4.1.3 Boundary-Based Subsegment Generation. Merging the different segmentation results can be difficult since merging is highly correlated with the color information. Instead of merging segments, We combine the boundaries from the UV map and the 3D space with a simple *add* operation as follows:

$$\boldsymbol{Bd}^* = \boldsymbol{Bd}_{UV} + \boldsymbol{Bd}_{3d} , \tag{3}$$

where $\boldsymbol{Bd}^*$ denotes the final boundary representation, integrating both UV and 3D boundary information to form a comprehensive boundary map. Using an inverse of the boundary (1 - $\boldsymbol{Bd}^*$), we use connected-component labeling (CCL) to convert mask islands to rough subsegments $\boldsymbol{P}_{\boldsymbol{Bd}^*}$. Then, we interpolate the label of boundary $\boldsymbol{Bd}^*$ to calculate the subsegments $\boldsymbol{P}_*$.

4.1.4 Precomputation of Subsegment Distance Table. Utilizing the subsegments P^* acquired in Section 4.1.1 and 4.1.2, we compute the subsegment distance table. First, we create a graph connecting neighboring subsegments with edges. When we create the graph, we define the cost of the edge by the color difference × Euclidean distance of the two subsegment's center points). Next, we define the distance between all pairs of subsegments using the Dijkstra algorithm. This step is important when determining the cost of the shortest path on the graph at user-guided segmentation (Sec. 4.2) Specifically, the distance calculation assumes that subsegments of similar colors and neighboring locations are likely to be classified within the same segment later during user-guide segmentation.

4.2 User-Guided Subsegment Clustering

Our interactive segmentation system clusters subsegments into segments under the guidance of user-provided scribbles as we shown

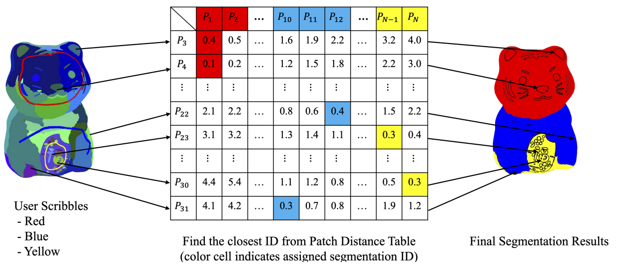

Figure 5: Propagation of Subsegment IDs. Based on the user scribbles, we assign the segmentation ID to the non-scribbled subsegments by checking the closest user scribble in the subsegment distance table. This yields the final segmentation results.

in Figure 5. Specifically, the user assigns segment IDs by drawing colored scribbles on the model, and the system propagates these segment IDs to all other subsegments. Specifically, the closest subsegments of already assigned subsegments are determined and the corresponding ID is assigned to the target subsegment.

For each new scribble operation, the i-th subsegment's ID is updated according to the newly added (or removed) scribbles. Subsegments with no scribbles are automatically assigned to the closest scribbled ID using the equation:

$$k^* = \underset{P_j \cap \boldsymbol{B}_{UV,t} \neq \emptyset}{\arg\min} \; \boldsymbol{G}_{P_i, P_j} \tag{4}$$

where k^* denotes the optimal index of the segmentation ID corresponding to the nearest scribbled subsegment, and $P_j \cap \boldsymbol{B}_{UV,t} \neq \emptyset$ denotes whether subsegment P_j is marked in t-th iteration's scribbling map $\boldsymbol{B}_{UV,t}$. Finally, we update the segmentation ID of subsegment P_i to denote it as k^*-th segmentation ID col_{k^*}.

5 User Study

We conducted a user study to assess the segmentation accuracy of our technique and interface compared to a baseline interface utilizing brush painting and a bucket tool. Since there is no established automatic process to segment scanned 3D models, we adopt fully manual brush-based painting as our comparison baseline, which is the baseline technique also used by Kawabe et al. [6] in their evaluation.

5.1 Experimental Settings

We recruited sixteen participants from our institution, all of whom had prior experience with 3D computer graphics, including gaming and CAD. Each participant took part in the study for approximately one hour. At the beginning of the session, participants were briefed on the experiment's procedures and asked to provide informed consent. An introductory tutorial on 3D segmentation was then presented, alongside detailed instructions on how to operate both the proposed and baseline interfaces. Participants were then trained with models (different from those used in the main task) using both interfaces to familiarize themselves with the tools.

For the main study, participants were asked to segment the broccoli and watergun models using each interface, i.e. participants performed four segmentation tasks in total. To enable direct comparisons of participants' performance and their experience with each interface, we adopted a within-subjects design. The objective was to achieve a segmentation that closely resembled a predefined target segmentation. Participants were allowed to refer to the target segmentation for the model at any time during the experiment.

To evaluate the precision of the segmentation maps created by participants, we evaluate the accuracy of segmentations as the proportion of correctly classified pixels on the UV map. A pixel was deemed correctly classified if it was marked with the right material type from three possible options or if all material values were zero for regions without an associated material.

5.2 Results

Segmentation Accuracy Table 1 shows a box plot of the accuracy of participant-generated segmentation results. On average, participants achieved an accuracy of 85.95% ± 2.75 for the Broccoli 3D model with our interface, surpassing the baseline's average accuracy of 67.57% ± 6.05. For the Watergun 3D model, our interface resulted in an average accuracy of 95.28% ± 0.97, while the baseline recorded 76.65% ± 8.48. The statistical significance of these differences was confirmed with a Mann-Whitney U test, following a Shapiro-Wilk test suggesting a non-normal distribution of the results. The significance levels were recorded at $p < 0.01$ for the Broccoli model and $p < 0.01$ for the Watergun model (with p-values adjusted using the Benjamini–Hochberg procedure for multiple comparisons). These results indicate that participants produced significantly more accurate segmentations for both models using our interface compared to the Blender baseline.

Figure 6 shows the top-performing segmentation results from the user study as well as the output of SAM3D [16].

A detailed analysis of the black boxes in Figure 6 indicates that our method achieves better alignment along texture boundaries and reduces painting errors compared to the baseline. The results for both models highlight that the ease of painting the 3D model greatly affects the performance of both interfaces. Overall, our semi-automatic user-guided technique greatly enhanced the precision of segmentations.

Preference on User Interface In our survey, fifteen participants indicated they preferred our interface while one participant preferred Blender, i.e. almost all participants favored our interface over the baseline.

Number of scribbles We examine the number of added scribbles with our interface to assess manual effort. On average, our participants used 24.38 ± 7.96 scribbles on average with our interface for the Broccoli 3D model, compared to 30.94 ± 8.01 scribbles with the baseline. For the Watergun 3D model, an average of 12.44 ± 4.52 scribbles were input with our interface, against 29.56 ± 8.42 for the baseline. These differences are statistically significant, as confirmed by a Mann-Whitney U test, following a Shapiro-Wilk test which suggested a non-normal distribution of the results. The significance levels were recorded at $p < 0.05$ for the Broccoli model and $p < 0.01$ for the Watergun model (with p-values adjusted using the Benjamini–Hochberg procedure for multiple comparisons).

Table 1: Segmentation accuracy: The results show that our approach enhances segmentation accuracy compared to the baseline. $*$ indicates statical significance ($* < 0.05$)

3D Model	Proposed method	Baseline (Blender)
Broccoli	**85.95**% ± 2.75*	67.57% ± 6.05
Watergun	**95.28**% ± 0.97*	76.65% ± 8.48

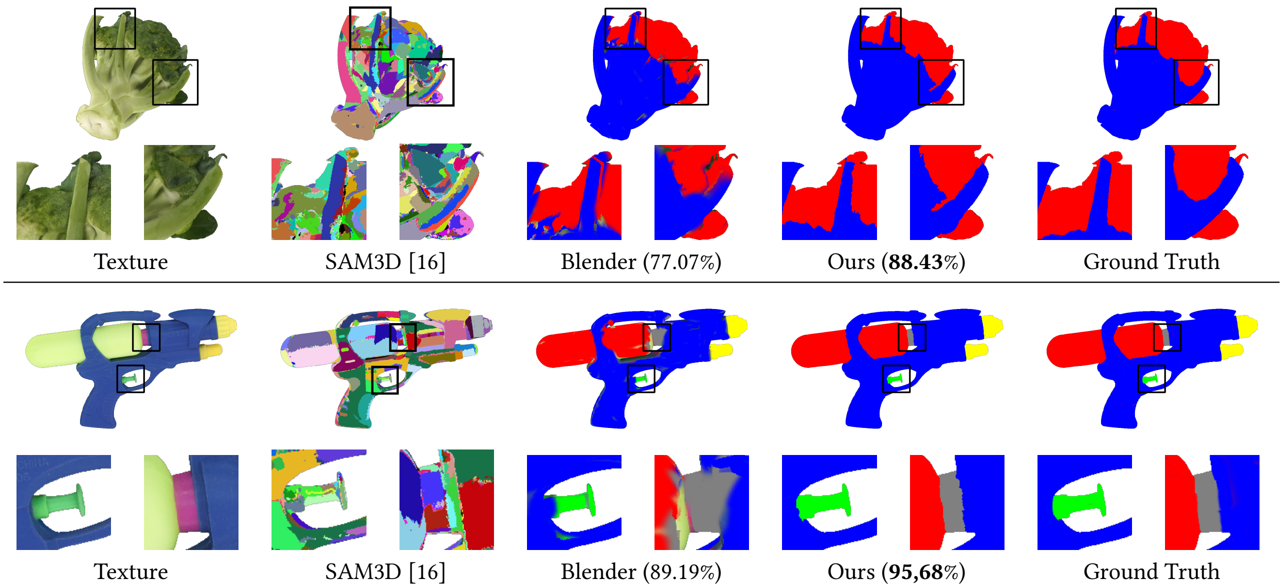

Figure 6: Segmentation results on user study. We visualize the model which shows the best segmentation accuracy in each condition. We highlight the improvement over the blender by the black boxes.

These findings suggest that our interface allows users to complete segmentation tasks with less effort.

Completion Time For the Broccoli 3D model, participants completed the segmentation task within 5 minutes using both interfaces. For the Watergun 3D model, participants averaged 3.68 minutes to complete the task with our interface, compared to 5 minutes with the baseline. Furthermore, 12 participants finished within 5 minutes using our interface, with the other 4 participants mainly using the last $1\tilde{2}$ minutes to check the segmentation result with the 3D view.

6 Conclusion

We proposed an interactive texture segmentation method for 3D scanned models, combining automatic segmentation with minimal manual input to balance efficiency and user control. In a preprocessing phase, the technique identifies small subsegments in the model, which are subsequently clustered by users using intuitive scribbles on the 3D model view. Evaluations confirmed that our method significantly improves segmentation accuracy and quality compared to conventional manual techniques using standard 3D computer graphics software. Additionally, our method produces more detailed segments compared to fully automatic algorithms that segment UV texture maps. In future work, we would like to reduce subsegment precomputing time to enable immediate segmentation of models upon loading in the application. Furthermore, we plan to implement additional tools to offer further flexibility in the segmentation process, e.g. splitting subsegments and editing their contours.

References

[1] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. 2012. SLIC Superpixels Compared to State-of-the-Art Superpixel Methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34, 11 (2012), 2274–2282. https://doi.org/10.1109/TPAMI.2012.120

[2] Faraz Faruqi, Ahmed Katary, Tarik Hasic, Amira Abdel-Rahman, Nayeemur Rahman, Leandra Tejedor, Mackenzie Leake, Megan Hofmann, and Stefanie Mueller. 2023. Style2Fab: Functionality-Aware Segmentation for Fabricating Personalized 3D Models with Generative AI. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*. 1–13.

[3] Clifton Forlines and Ravin Balakrishnan. 2009. Improving visual search with image segmentation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Boston, MA, USA) *(CHI '09)*. Association for Computing Machinery, New York, NY, USA, 1093–1102. https://doi.org/10.1145/1518701.1518868

[4] Haoyu Guo, He Zhu, Sida Peng, Yuang Wang, Yujun Shen, Ruizhen Hu, and Xiaowei Zhou. 2023. SAM-guided Graph Cut for 3D Instance Segmentation. *arXiv preprint arXiv:2312.08372* (2023).

[5] Takeo Igarashi, Naoyuki Shono, Taichi Kin, and Toki Saito. 2016. Interactive Volume Segmentation with Threshold Field Painting *(UIST '16)*. Association for Computing Machinery, New York, NY, USA, 403–413. https://doi.org/10.1145/

2984511.2984537

[6] Wataru Kawabe, Taisuke Hashimoto, Fabrice Matulic, Takeo Igarashi, and Keita Higuchi. 2023. Interactive Material Annotation on 3D Scanned Models leveraging Color-Material Correlation. In *SIGGRAPH Asia 2023 Technical Communications* (Sydney, NSW, Australia) *(SA '23)*. Association for Computing Machinery, New York, NY, USA, Article 18, 4 pages. https://doi.org/10.1145/3610543.3626170

[7] Lei Ke, Mingqiao Ye, Martin Danelljan, Yifan Liu, Yu-Wing Tai, Chi-Keung Tang, and Fisher Yu. 2023. Segment Anything in High Quality. In *NeurIPS*.

[8] Yujin Kim, Eunyeoul Lee, Yunjung Lee, and Uran Oh. 2024. Understanding Novice's Annotation Process For 3D Semantic Segmentation Task With Human-In-The-Loop. In *Proceedings of the 29th International Conference on Intelligent User Interfaces* (Greenville, SC, USA) *(IUI '24)*. Association for Computing Machinery, New York, NY, USA, 444–454. https://doi.org/10.1145/3640543.3645150

[9] Theodora Kontogianni, Ekin Celikkan, Siyu Tang, and Konrad Schindler. 2023. Interactive object segmentation in 3d point clouds. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2891–2897.

[10] Carina Liebers, Marvin Prochazka, Niklas Pfützenreuter, Jonathan Liebers, Jonas Auda, Uwe Gruenefeld, and Stefan Schneegass. 2024. Pointing it out! comparing manual segmentation of 3d point clouds between desktop, tablet, and virtual reality. *International Journal of Human–Computer Interaction* 40, 19 (2024), 5602–5616.

[11] James MacQueen et al. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, Vol. 1. Oakland, CA, USA, 281–297.

[12] Valentin Z Nigolian, Takeo Igarashi, and Hirofumi Seo. 2019. INVANER: INteractive vascular network editing and repair. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. 1197–1209.

[13] Zhongzheng Ren, Aseem Agarwala, Bryan Russell, Alexander G. Schwing, and Oliver Wang. 2022. Neural Volumetric Object Selection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 6133–6142.

[14] Patric Schmitz, Sebastian Suder, Kersten Schuster, and Leif Kobbelt. 2022. Interactive Segmentation of Textured Point Clouds. In *Vision, Modeling, and Visualization*, Jan Bender, Mario Botsch, and Daniel A. Keim (Eds.). The Eurographics Association. https://doi.org/10.2312/vmv.20221200

[15] Jingying Wang, Haoran Tang, Taylor Kantor, Tandis Soltani, Vitaliy Popov, and Xu Wang. 2024. Surgment: Segmentation-enabled Semantic Search and Creation of Visual Question and Feedback to Support Video-Based Surgery Learning. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. 1–18.

[16] Yunhan Yang, Xiaoyang Wu, Tong He, Hengshuang Zhao, and Xihui Liu. 2023. Sam3d: Segment anything in 3d scenes. *arXiv preprint arXiv:2306.03908* (2023).

[17] Zhongyi Zhou and Koji Yatani. 2022. Gesture-aware interactive machine teaching with in-situ object annotations. In *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology*. 1–14.