History Assisted View Authoring for 3D Models

Hsiang-Ting Chen[1,2], Tovi Grossman[1], Li-Yi Wei[3], Ryan Schmidt[1], Björn Hartmann[4], George Fitzmaurice[1], and Maneesh Agrawala[4]

[1]Autodesk Research [2]The University of Tokyo / JST ERATO Igarashi Design Interface Project
[3]The University of Hong Kong [4]University of California, Berkeley

ABSTRACT

3D modelers often wish to showcase their models for sharing or review purposes. This may consist of generating static viewpoints of the model or authoring animated fly-throughs. Manually creating such views is often tedious and few automatic methods are designed to interactively assist the modelers with the view authoring process. We present a view authoring assistance system that supports the creation of informative view points, view paths, and view surfaces, allowing modelers to author the interactive navigation experience of a model. The key concept of our implementation is to analyze the model's workflow history, to infer important regions of the model and representative viewpoints of those areas. An evaluation indicated that the viewpoints generated by our algorithm are comparable to those manually selected by the modeler. In addition, participants of a user study found our system easy to use and effective for authoring viewpoint summaries.

Author Keywords

3D model; editing history; viewpoint authoring

ACM Classification Keywords

H.5.2 User Interfaces: Interaction Styles

INTRODUCTION

Recent years have witnessed significant progress in 3D modeling and 3D printing. As the technologies are becoming essential to the industries and even people's daily life, the demand of high quality 3D models also increases rapidly. This trend has resulted in a variety of 3D model libraries and websites where 3D modelers can share models they created.

Understanding each model through 3D navigation can be a time-consuming and confusing experience. Thus, to achieve efficient browsing of large 3D model data sets, it is necessary to have visual summaries that can concisely show important aspects of the 3D models. An effective visual summary usually consists of static viewpoints and animated fly-throughs of the model. However, manually creating such views can be a long and tedious process.

CHI 2014, April 26 - May 01 2014, Toronto, ON, Canada

ACM 978-1-4503-2473-1/14/04…$15.00.
http://dx.doi.org/10.1145/2556288.2557009

As a result, automatic viewpoint selection [11, 15] has been an active area of research in the Computer Graphics community. However, the associated algorithms that have been developed typically consider only the final geometric models. Such algorithms may ignore features of the model that an author spent a particular amount of effort on during

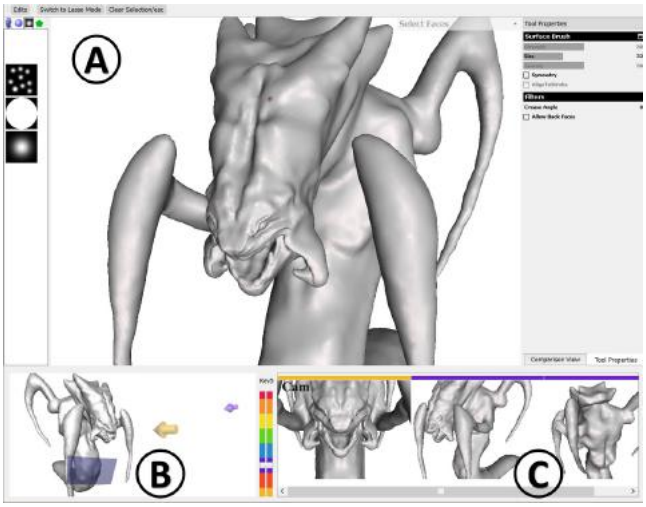

Figure 1. The proposed view authoring environment integrated in a 3D modeling software: (A) the main modeling window, (B) the authoring panel visualizing authored views in the spatial context, and (C) the navigation panel showing the authored views in the temporal sequence.

the modeling process. Furthermore, it is not clear how to integrate these automatic techniques into a modeler's view authoring process in practice.

Alternatively, the HCI literature has investigated techniques for navigating along constrained view paths and view surfaces to obtain effective overviews of 3D models [1, 2]. However, these systems require predefined constrained views. The authoring of such views, either manually or automatically, remains an open problem.

Recent work has demonstrated the utility of instrumenting a 3D modeling environment for visualizing editing operations and model differences [6, 7]. Guided by a set of observations and interviews, we hypothesize that enhancing such instrumentation to also include the camera history could be useful for authoring viewpoint summaries. In particular, the workflows used to create a model may be indicative of the important parts of the models and the viewpoints from which these important parts should be

viewed. By recording and analyzing these workflows, it may be possible to derive effective summary views.

In this paper, we contribute a history assisted view authoring and navigation system for 3D modelers (Figure 1). Our system stores a model's editing and viewpoint history along with discrete versions of that model. By analyzing the history and the model, our system can suggest informative view points, animated view paths and view surfaces of the selected model region. These can be used for automatic visual summaries of 3D objects or in our interactive view authoring environment where modelers can select and customize such summaries.

Our system produces an interactive summary of the 3D model, allowing viewers to easily navigate between authored views and along view paths or surfaces, and even compare different versions from the model's revision history. Alternatively, our system can also automatically arrange the authored views into user-defined layout templates for a static brochure or catalogue-like display.

In a first study, we compare the viewpoints generated by our algorithm to those generated by Secord et al [13], and to manually created views by 3D modelers. The results show that our algorithm selects viewpoints with comparable quality to manual selection. In a second qualitative user study of our interactive system, participants expressed high enthusiasm and a desire to incorporate the system into their daily workflows.

PREVIOUS WORK

3D Navigation

Many existing techniques have attempted to simplify 3D navigation, which is often a challenge for inexperienced users [1, 2]. ShowMotion [2] replaces static camera bookmarks with pre-authored animated shots, and StyleCam [1] supports constrained navigation along pre-authored camera paths. Both works greatly improve users' viewing experience of a 3D model. However, these systems only consider the viewers of the 3D model and it is unclear how a modeler can author the required views. For example, ShowMotion [2] assumes pre-authored viewing motions and StyleCam [1] pre-authored surfaces and paths. Our work bridges the gap by assisting the modeler to author such viewpoints, view paths, and view surfaces.

3D Viewpoint Selection

At the core of our system is the viewpoint selection algorithm. Previous work [11, 15] infers the importance of viewpoints based on the visible geometric attributes of a given 3D surface. Secord et al. [13] summarizes existing techniques, and generates weightings of the various criteria explored in the literature such that the fitted model can predict people's preferred views. By additionally considering the editing history of a 3D model, our approach could potentially create viewpoint selection results that better convey the original author's intentions.

Tsang et al. [17] and Singh et al. [16] mine viewpoint data from users viewing a model after its creation. The collected data is used to summarize viewing behaviors [17] or assist in the future presentation of the model [16]. In contrast we instrument the authoring environment so that viewpoint information can be captured, and informative viewpoints can be inferred as soon as the model is created.

Capturing and Utilizing Workflow Histories

Numerous systems have been proposed to automatically record users' workflows within a software application. Captured interaction history and workflows have been shown to be useful for creating step-by-step tutorials [4, 8] or allowing users to explore and replay editing history of a document [3, 6, 9]. Our work on viewpoint summarization demonstrates a previously unexplored way to utilize captured workflows: to select views and create model summarizations.

INITIAL INSIGHTS AND OBSERVATIONS

A core problem related to our work is the automatic selection of informative viewpoints of a 3D model. Our premise is that the behavior of modelers as they create the model can provide useful cues for viewpoint selection. To better understand the validity of this premise, and gain other insights for guiding our work, we conducted multiple interviews and observation sessions with professional 3D modelers. This section summarizes the details of these sessions and some of the insights gained from them.

Interviews and Observation Sessions

We conducted three separate 45-minutes interview sessions with three 3D modelers. Two of the participants were internal to our institution, and one was externally recruited.

There were five additional observation sessions with the two internal modelers. Each session is 1 hour long of 3D sculpting [12] followed by 20 minutes of discussions. An observer took notes during the sessions, and the modeling processes were screen-captured for later reference.

Cameras Oscillate between Sculpting and Inspection

We observed a camera movement pattern consistent across modelers, where they oscillate between sculpting at camera position A, move to position B for inspection, then move back to position A for further editing.

Such oscillation patterns are a necessity in sculpting tools. Typically, the 3D viewpoint which allows the modeler to precisely position a brush (*sculpting view*) is looking directly down the normal of the sculpting region. Yet the best view for assessing the effect of the brush (*inspection view)* is typically perpendicular to the plane normal.

Since these oscillations are often used for inspection, capturing and analyzing the inflection point of these oscillations may allow us to infer preferred viewpoints for specific areas of the model.

Sculpting Brushes are Short and Localized

We also observed that modelers tend to sculpt with short brush strokes localized to surface features, e.g. eyes and mouth, rather than long winding brushes across an entire surface. Presumably this is because 3D sculpting is largely a process of accumulating small additions and removals of local details. Similarly, modelers tend to work in a spatially- and temporally-coherent fashion.

Both observation and interviews confirmed that modelers tend to focus on one area of the model at a time, before moving on to the next. This indicates that we may be able to segment the 3D model into meaningful parts and to identify important regions, for the purpose of viewing or summarization, based on a modeler's editing history.

Complexity Does Not Imply Importance

Generally it has been assumed that geometric complexity (e.g. curvature variation) correlates well with importance [13, 15]. However, in 3D sculpting, we sometimes observe the exact opposite. Geometrically complex regions such as hair, scales, and textured skin are often quickly created by irregular scribbling of the sculpting brush, whereas very smooth regions that define the overall form (e.g. the shapes of hips, thighs, and shoulders) often require much more careful attention. These smooth "features" are usually marked as unimportant by geometric analysis, but are considered important by the modeler.

View Authoring is an Iterative Process

We observed that when authoring static views, the modelers first quickly selected many candidate views, followed by gradual pruning and refinement. Similarly, when authoring animated views, the modelers first assigned rough control points and then refined the paths and surfaces in detail. Such iterations between selection and refinement happened frequently throughout the view authoring process.

From our interviews we learned that the process could be tedious, spanning from several minutes to more than an hour, if animated views for walk-throughs or fly-bys were included. In general, the modelers welcomed aids for automation but also showed strong demand of some level of manual control over the final results.

OVERVIEW OF OUR SYSTEM

The observations and insights in the previous section motivate us to build a view authoring system that interactively suggests candidate static and animated views to the modeler during the view authoring process.

The core of the system is an automatic view suggestion algorithm driven by the modeler's own editing history. It is based on the observation above that there are significant amount of regularity in the camera movements and sculpting strokes and they may provide useful information for modelers' view authoring that is not readily apparent in the final 3D model. In particular, the oscillating camera movements suggest the existence of representative view clusters while the localized brushes suggest the possibility of segmenting the 3D model into meaningful components.

In the following sections, we first introduce the user interface of our authoring and viewing environments, followed by algorithm and implementation details.

VIEW AUTHORING USER INTERFACE

As a prototype system, we instrumented the 3D sculpting tools in the MeshMixer [12] modeling software for recording the editing history, and integrated the authoring environment inside MeshMixer.

Our view authoring system supports both authoring and viewing modes across three main categories of view widgets: view points, view paths, and view surfaces (Figure 2, as inspired from [1]). As an author, a user can generate a summary of the model through a series of informative viewpoints, constrained view paths and surfaces, and comparison views across versions of the model in its editing history. As a viewer, the user can easily inspect and navigate the model with the aid of these authored views.

System Interface

Figure 1 shows the user interface of our system, which consists of three main components: the *main window*, the *overview panel*, and the *navigation panel*.

Main Window (Figure 1A)

The *main window* (Figure 1A) of MeshMixer is used to display the 3D model. It is also the main area for inspecting the 3D model and for authoring the views.

Overview Panel (Figure 1B, Figure 2)

The purpose of the overview panel is to show all authored views within the spatial context of the current 3D model. Different visualizations are used to represent viewpoints, view paths, and view surfaces (Figure 2).

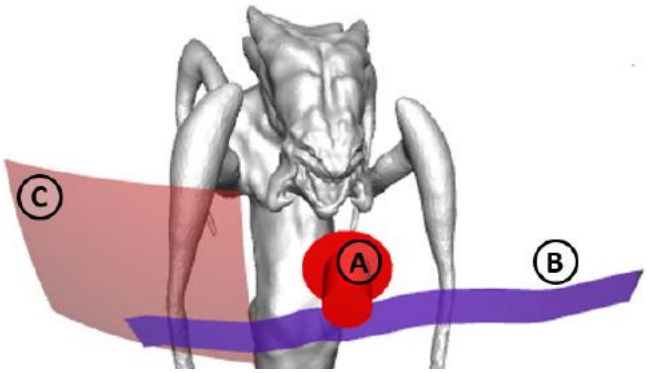

Figure 2. The overview panel contains a proxy of the 3D model and authored views. These can include (a) viewpoints, (b) view paths, and (c) view surfaces.

Navigation Panel (Figure 1C)

The *Navigation Panel* allows the user to navigate between authored views. This panel consists of two parts: a color-coded *version slider* on the left and a thumbnail list of authored views on the right.

The version slider allows users to navigate through previous versions of the model that were manually saved by the modeler or automatically saved by the system based on user-specified elapsed time and operation count. Each version has its own unique color code, and the corresponding model version will be displayed in the main window as the user drags the slider.

The thumbnail list shows the representative images of all authored views with a color-coded stripe indicating the corresponding versions. Double clicking on the thumbnail item loads the corresponding view and version into the main window, and highlights the corresponding view widget (i.e. the viewpoint arrow) in the overview panel.

Authoring Environment

In the following paragraphs, we use a 3D hydra model as an example to walk through the view authoring scenario and explain individual features of our system.

Manual Viewpoint Authoring

The modeler can inspect the 3D model in the *main window* with standard camera controls. When viewing the model from a desired viewpoint, the modeler can store the viewpoint. This authored viewpoint will then be added to both the *overview panel* and the *navigation panel*.

Authoring Region-Specific Viewpoints

Per our initial observation of the iterative authoring process, the modelers might desire a list of view candidates they can choose from and customize. Thus our system provides the function of interactive candidate viewpoint generation for a user-specified region of interest. First, the modeler specifies a region of the 3D model by directly painting the region on the surface of the model using the triangle selection function in MeshMixer (Figure 3). A list of viewpoints associated with the specified region is then displayed in the main window.

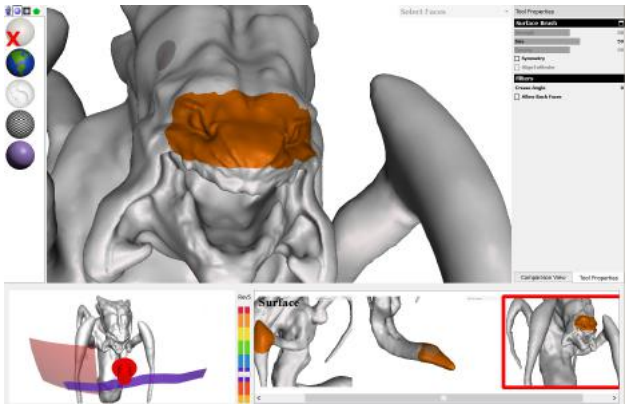

Figure 3. Region-specific view authoring. The orange colored region is the selected region of interest. Our system then suggests viewpoints, view paths, or view surfaces.

Authoring Region-Specific View Paths and View Surfaces

The *region-specific* view authoring mechanism also allows for efficient authoring of 3D view paths and view surfaces. The modeler can first specify a region of interest, as described above, then invoke the context menu to have the system automatically suggests multiple candidate view surfaces and view paths.

Suggested view surfaces are 3D spherical patches that spatially constrain the viewing camera for easier navigation. Suggested view paths are 3D paths consisting of a series of camera viewpoints. Once a candidate is chosen, the modeler can adjust the size of a view surface and the position and length of a path by dragging the mouse.

Automatic Global Viewpoint Suggestion

Alternatively, our system can also fully automatically generate the candidate views of the entire 3D model (Figure 4). The modeler can directly use the generated result to save the view authoring time or select the favored views and customize them.

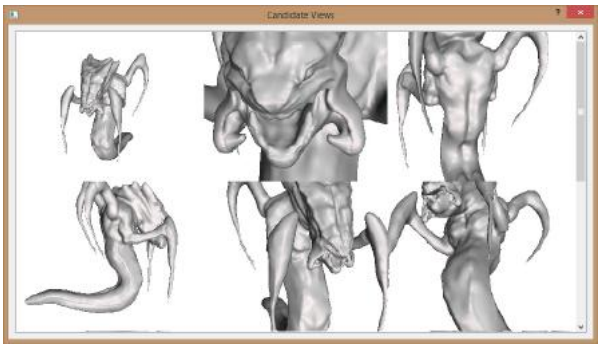

Figure 4. Candidate view-points picked by our automatic viewpoint selection algorithm.

Authoring Views for Different 3D Model Versions

The additional temporal dimension embedded in the editing history enables the modeler to revisit previous model states in the modeling process and associate authored view with the version. The arrow in the overview panel used to visualize the viewpoint is color-coded to represent its associated version.

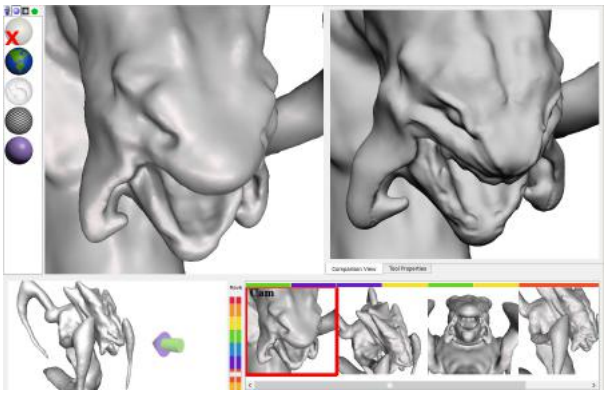

Figure 5. Comparison view function that shows the same viewpoint of two 3D model versions side-by-side. The arrow in the overview panel has two colors representing each version.

Authoring Comparison Views between 3D Model Versions

Our system also provides a *comparison view* function, which allows the modeler to display two different 3D

model versions side-by-side (Figure 5). A context menu is used to toggle the comparison view mode on and off. Once enabled, the user can drag the *version slider* to set the versions being displayed.

In a comparison view, the camera navigation of both versions is synchronized, so that the modeler can easily compare specific areas of the model across versions. This function is particularly helpful when the modeler would like to highlight the evolution and differences of a 3D model in the modeling process. The views authored within these *comparison views* are displayed in the overview panel as a two-color arrow, which corresponds to the color codes of the two versions (e.g. the purple/green arrow in the *overview panel* in Figure 5). Selecting any two-color arrow from the overview panel will bring up the associated side-by-side comparison view into the *main window*.

Editing History Visualization

To further facilitate view authoring, our system can visualize various aspects of the editing history, including time spent on each surface region and the camera paths. By accumulating the sculpting time spent on surface regions we can render a time map on the surface (Figure 6a), while for camera history we create 3D ribbons along the camera paths (Figure 6b). Note that the homogeneous colors around features, such as the eyes, the wings and the chest, support our observation that brush operations tend to be localized. The modeler can author views while the editing history visualization is displayed.

The sculpting history may consist of thousands of operations, thus we also provide a region-specific history filter function (Figure 6b). The modeler can then navigate between the filtered camera positions.

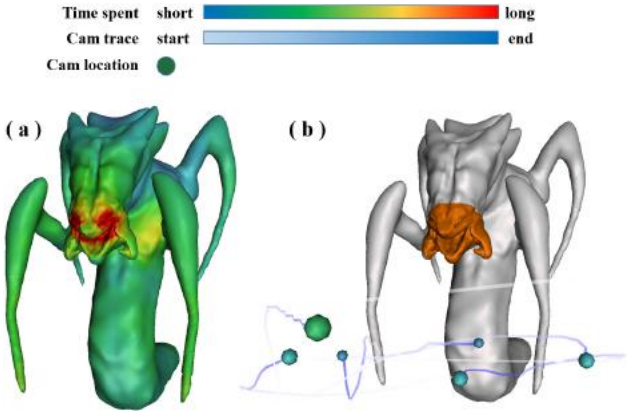

Figure 6. Visualizations of history data. a) A heat-map shows the time spent on each vertex in the model. b) A visualization of the camera history while editing the highlighted face region.

NAVIGATION EXPERIENCE FOR VIEWERS

The authored viewing information and associated representative thumbnail images are stored as meta-data of the 3D model. Figure 7 shows a sample authored result. This can be used to create an interactive navigation environment for other viewers.

In our current prototype, we reuse MeshMixer as the viewer for the viewing information. However, since the information is tool independent, it can be easily adopted by other 3D modeling systems [1, 2].

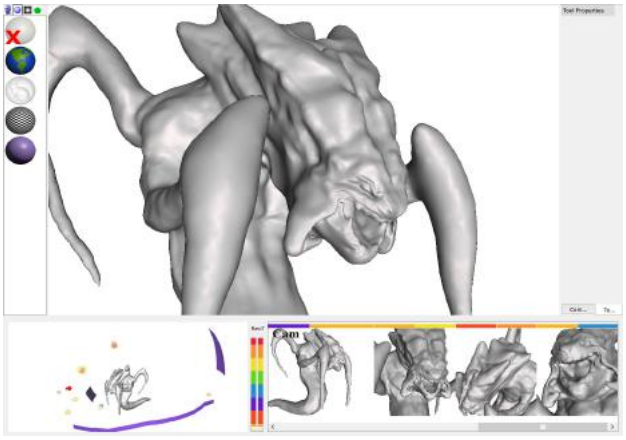

Figure 7. Views authored by the modeler using our system.

A viewer can explore the model using both the *navigation* and *overview panel*s. For example, the viewer can double click on a preview thumbnail in the navigation panel, which will load the associated version and viewpoint into the main window. When the viewer selects a pre-authored path or surface, the user can perform the associated constrained navigations of the model, e.g. follow the authored view path or drag the main camera on the authored surface. This allows the viewer to easily insect the model and experience high-quality views that the modeler has created.

Exporting 2D Image Collages

To further enhance the portability of the authored viewing experiences, our system provides the ability to export the authored views as a collage of representative 2D images. Our system pre-renders the authored views and arranges them according to pre-defined template layouts. The tool is particularly useful in helping modelers showcase their work on printed mediums or devices with low graphics capability. Our system provides two kinds of collages: *summary collage* and *progress collage*.

Figure 8. Summary collages on a tablet and a smart phone.

A *summary collage* presents a collection of rendered views that had been created in the authoring environment (Figure 8). Predefined templates are used for the layout, consisting of equally-sized tiles, or a central primary view

supplemented with smaller secondary views. Users can drag along individual thumbnails to browse views pre-rendered along the authored view-paths and view surfaces.

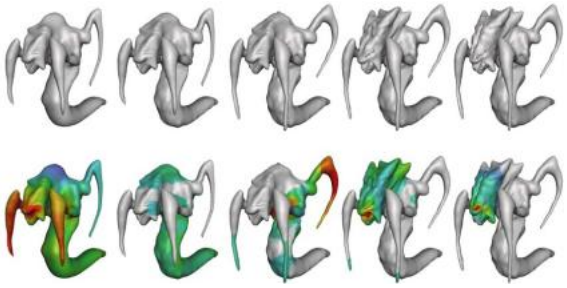

Figure 9. Temporal review using *progress* collage. Top row shows progress of the modeler sculpting a hydra. Bottom row shows time map of each version.

A *progress collage* is meant to be used for design review and workflow summarization. The *progress collage* displays a series of version snapshots from a fixed viewpoint (Figure 9). In each snapshot the surface is colored using a heat-map based on the accumulated editing time difference from the previous snapshot. This visualization enables users to rapidly assess which areas of the model have been changed between versions.

ALGORITHM AND IMPLEMENTATION

In this section we elaborate more on the instrumentation of MeshMixer and the algorithms used to construct candidate viewpoints, view paths, and view surfaces.

Editing History Instrumentation

The editing history recorded by our instrumented MeshMixer comprises of a complete log of the two most significant interactions in a 3D sculpting tool: interactive sculpting brush strokes, and 3D camera manipulations.

For each sculpting brush operation we store: a unique operation ID, elapsed time of the stroke, a list of the affected vertices, and the current camera viewpoint.

For the camera history we store all intermediate cameras (location, look-at, and up vector) during each camera transformation operation, as well as the elapsed time of the entire camera manipulation.

Region-Specific View Suggestion

Our system interactively suggests candidate views based on the region specified by the modeler. The following paragraphs describe the algorithms that generate candidate viewpoints, view paths, and view surfaces.

Candidate Viewpoint and View Path Construction

Our approach is motivated by the camera oscillations discussed in our initial observations. The captured *sculpting views* and *inspecting views* tend to form dense, disjoint clusters whose weighted centers we interpret as candidate viewpoints for the surface region specified by the modeler. From these candidate views, we interpret the associated oscillating camera traces as candidate view paths.

Given the modeler-specified region, we identify *sculpting views* as those camera positions where the modeler applied a sculpting brush to that region. We then apply spatial clustering to these views to pick representative ones. Our algorithm takes a set C of sculpting-view cameras, and builds an octree based on the camera positions. We then compute the accumulated camera time in each cell by summing the camera times in its sub-tree, and search for the dominant cell bottom-up until we find the one whose accumulated time is the largest among cells at the same octree level and is over 70% (T_1) of the total time. Finally, we calculate the weighted average center (based on time) of the viewpoints in the dominating cell, and select the camera in C that is closest to the weighted center. We denote this camera viewpoint c_{sculpt}, the representative sculpting-view for the specified region.

To identify *inspection views*, viewpoints used by the author to assess the model, we search for camera manipulations which occur between two sculpting operations applied to the same segment. We treat these as inspection operations. For each such camera manipulation, the sequential camera positions create a 3D piecewise-linear curve L. We compute the opening angle at each vertex (camera position) of L, and choose as a turning point, c_{turn}, the vertex with largest opening angle (Figure 10). For a given region, we take the 3D cameras associated with all of the turning points and then apply the same clustering technique described above to extract the inspection-view, $c_{inspect}$, for a segment.

Note that the camera controls between sculpting might not always be inspections. However, the experiment results seem to show that the modeler tend to stay on non-inspection cameras position much shorter and our algorithm is capable of identifying meaningful $c_{inspect}$. Furthermore, the goal is to provide suggested views for users to refine and choose and thus 100% accuracy is not necessary.

Iterating the procedure described above, we can extract multiple sculpting views and inspection views. We then treat these extracting views as candidate viewpoints and the oscillating camera traces these views belong to as candidate view paths of the specified region.

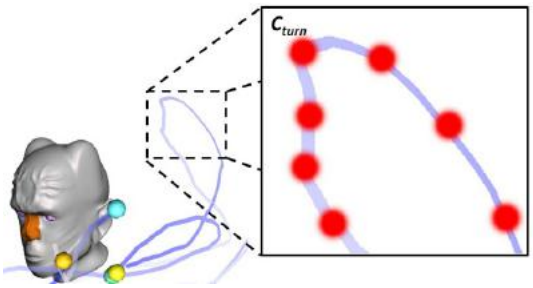

Figure 10. *Cturn* is the turning point of the oscillation camera movement pattern.

Candidate View Surface Construction

We hypothesize that a view surface that covers a cluster of sculpting views would be a good candidate for inspecting the selected region. Given a surface region of a 3D model,

we calculate the average center v_{center} of the surface vertex and obtain its dominating octree cell of sculpting views and representative sculpting-view c_{sculpt} with the algorithm described above. Next, we construct a spherical patch with the center at v_{center}, radius as $\|c_{sculpt} - v_{center}\|$, and spanning angles that cover the octree cell.

Global Viewpoint Suggestion

In addition to suggest views for user specified regions, our system can also automatically segment the 3D model into meaningful surface regions and suggest good viewpoints for each of them. This allows our system to recommend global viewpoints that summarize the entire 3D model.

Per our initial observations, the modelers sculpt sequentially and locally. This leads to a feature-centric approach - we explicitly segment the surface into (potentially overlapping) regions based on the editing history and then rank them by an importance factor.

We first define a segment as a region in which each vertex has been given a roughly similar amount of attention - measured by accumulated sculpting time, by the modeler. We then extract segments via standard greedy region-growing [14]. To create a segment *S*, we first choose as the segment *seed* the vertex v_s with the largest accumulated sculpting time $T(v_s)$. We then incrementally include vertices *v* adjacent to *S* which have a $T(v)$ similar to the average accumulated sculpting time for S, denoted as $T(S)$. Specifically, we add *v* to *S* if $\|T(S) - T(v)\| < T_2 = 0.3*T(S)$. The set S is then removed from the candidate set and we repeat the above process until either the required number of segments are found, or all vertices have been consumed.

Finally, we apply a refinement post-process. For each modeling operation that affects any of the vertices contained in segment S, we grow S to include all vertices of that operation. This is based on the previous observation that brush strokes tend to be local and stay on same feature.

Figure 11. Segments generated by our algorithm.

Given surface segmentation, we then collect the c_{sculpt} and $c_{inspect}$ for each segment, as described above, and treat them as the initial candidate viewpoints.

Discussions

Compared to previous algorithms whose computational cost is related to the geometric properties (e.g. surface curvature, surface variance, and silhouette length), the cost of our algorithm is related to the length of editing history and dominated by the cost of the octree construction. Our prototype system is deployed as a CPU based implementation which constructs the octree in real-time for the models used in the paper (Table 1). For complex models with longer editing history, a faster GPU implementation might be needed to achieve good performance.

The two threshold values in camera clustering (T_1) and region growing (T_2) can be interactively adjusted by the users. The results will be visualized on-the-fly as in Figure 6 and Figure 11. In our current system, we hand-picked the value of $T_1 = 0.7$ and $T_2 = 0.3$ for all results in the paper.

It is also worth noting that previous work [13] assumes a 2 DoF camera model, where cameras lying on a fixed-radius sphere facing the center of the model. On the contrary, our framework extracts cameras from the editing history and preserves the 6 DoF camera model.

VIEWPOINT SELECTION ALGORITHM EVALUATION

The practical benefits of our view authoring system strongly rely on a premise that the history-assisted viewpoint selection algorithm generates viewpoints suitable for understanding the modeling process and the workflow history. In particular, it is important to verify the underlying assumption that good editing views are correlated with good display views.

To check this premise, we compare our automatic global viewpoint suggestion algorithm to two alternatives: viewpoints generated automatically by the state of art method in Secord et al. [13] and viewpoints hand-specified by the original 3D modeler. Note that it is not our goal to outperform the manual selections. Instead, we hope to get close to the quality of manually generated viewpoints with the advantage of a time saving automatic approach.

3D Models

We recruited a 3D modeler to create three 3D models from a sphere using our instrumented version of MeshMixer. Table 1 provided a summary of the 3 models. These models had varying levels of complexity and geometry features. For example, the hydra model had fine-grained details that would have required careful close-up viewing and editing while the squirrel model had a smooth shape suitable to be viewed far away.

name	daemon	squirrel	hydra
preview			
minutes	52	30	55
vertices	67,418	27,902	130,658
editing op	1,566	420	1,022
camera op	738	247	425

Table 1. The 3D models created by a modeler for our study, using our instrumented version of MeshMixer.

Viewpoints

Once the 3D models were created, we generated 5 best views of each model using three different methods: our automatic viewpoint selection algorithm (*History Assisted*), Secord et al.'s automatic algorithm using linear-5b measure [13] (*Secord et al.*), and viewpoints hand-picked by the 3D modeler (*Modeler*).

For the algorithmic approaches we used the 5 top-ranked views for each 3D model. For the manual approach, we instructed the 3D modeler to "assume you are going to showcase your model on a website where one can only upload 5 images per model". It took the modeler about 15 minutes to pick all 15 static images. Figure 12 shows some viewpoints generated by these three methods. Please refer to the appendix file for complete materials.

Participants

To evaluate the viewpoints, we recruited 10 professional 3D modelers for the study, with up to 12 years of experience ($\mu = 5$ and $\sigma = 3.5$).

Procedure

In the beginning of the session, we briefly introduced our system and study to the participants. Then we gave the participants a chance to briefly inspect each of our three models in MeshMixer. Participants then carried out 2 tasks.

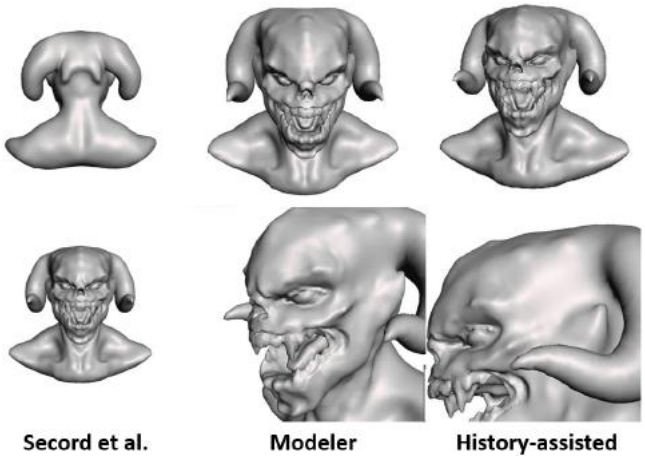

Figure 12. Sample viewpoints generated for the Daemon model for each of the three techniques.

Task 1 – Single View

The goal of the first task was to evaluate the quality of individual viewpoints. For each model, we showed the 15 images generated by the three techniques, one at a time, in random order. For each image, two questions were asked:

1. *The image conveys the information of (a) shape, (b) detail, or (c) shape + detail (pick one from these three)*
2. *I will take a picture of this 3D model from this angle*

Task2 – Multiple View

The goal of the second task was to evaluate the quality of viewpoints under the scenario of multiple image selection. For each model, we showed the participants 15 images at once in a 3×5 grid, and instructed them to pick five, under the assumption that they intended to showcase these models on a 3D forum.

On average, two task took approximately 30 to 40 minutes to finish. In the end of the session, we briefly interviewed the participants about the reasons for their ratings.

Evaluation Results and Discussions

Unless otherwise stated, we analyzed non-parametric participant data (Likert scale in case 1 and image count in case 2) with the Friedman test and pair-wise comparisons with the Wilcoxon signed-rank test.

Figure 13 shows the result of the first question in task 1. Although there is no statistically significant difference among the data, the responses indicated that our approach covers similar combination of shape and detail to the manual approach. These results are consistent with a visual inspection of the viewpoints generated by the three algorithms (Figure 12). It can be seen that the *Secord et al.* viewpoints provide more overall shape yet less detail due to its 2 DoF camera model.

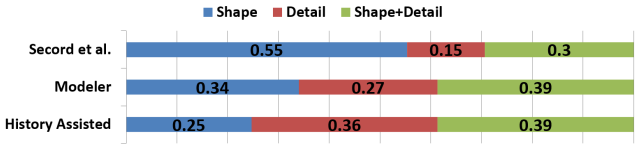

Figure 13. The result of first question in task 1. Each graph bar shows the percentages of participant labeling.

Figure 14 shows the result of the second question. The average rating for the *history assisted* viewpoints (3.24) is significantly higher than the *Secord et al.* viewpoints (2.80) ($p<0.05$). There was no significant difference between the *modeler* and *history assisted* methods.

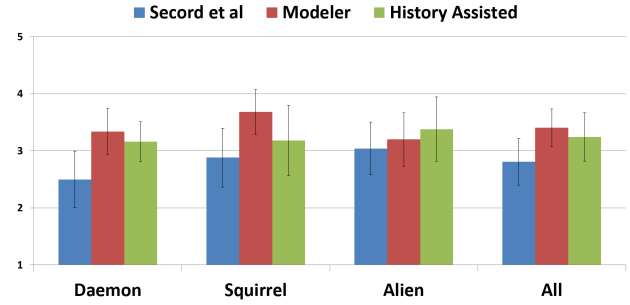

Figure 14. Average rating (from 1 to 5) for viewpoints in task 1. Error bars show 95% CI.

For task 2, we measured the average number of images picked from the three different methods. The averages were 1.3 (*Secord et al.*), 1.833 (*modeler*), and 1.867 (*history assisted*). Our algorithm had the highest rate of being picked by participants, but there were no significant differences among the three groups ($p=0.38$).

In summary, the results showed that good views from the editing history are indeed correlated with good display views. The viewpoints generated by our algorithm are similar in quality to those handcrafted by the 3D modeler,

with the advantage of being automatic. As such, our algorithm does provide a suitable method for automatically selecting viewpoints, as used in our view authoring system.

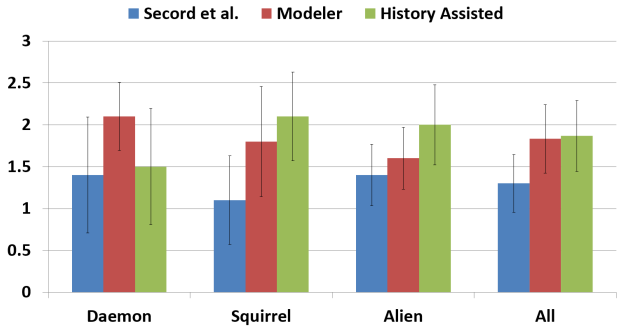

Figure 15. Average number of selected images in task 2. Error bars show 95% CI.

In addition, our study demonstrates the limitations of using existing automated techniques for assisting modeler's viewpoints authoring process. While Secord et al. [13] has been shown to be effective for modeling crowd preferences, its simplified 2 DoF camera model might miss certain viewpoints preferred by the modeler, e.g. close-up views or views not pointing to the center of objects.

In the post-interview, we found that even when two participants gave the same view an equal score, the provided reasons could differ dramatically. For example, for the bottom right image of Figure 12, a participant rated it as 5 because it showed the shape of the horn, while another rated it as 5 because it showed a clear silhouette of the head. It suggested that people could observe the same 3D model for different personal purposes, and it is difficult, if not impossible, to have a single automatic algorithm to cover everyone's preference. This reiterates the findings in our initial interview sessions, where the 3D modelers welcomed the automatic assistance on the view authoring while also considered the manual adjustment/modification over the final result necessary.

VIEW AUTHORING SYSTEM EVALUATION

The second user study was designed to provide a qualitative evaluation of the entire view authoring system, and to gain insights, observations, and feedbacks.

Participants and Materials

To evaluate the authoring system, we recruited three professional 3D modelers, two computer graphics researchers, and one architecture researcher. We used the same 3D models and editing histories in Table 1.

Procedure

The evaluation sessions began with a 20 minute walkthrough of the view authoring system. We first demonstrated individual features of our system, and then asked the participant to complete related short tasks. There were a total of 24 short tasks, such as "add a view path for the body region", and "add a comparison view for the head region for version 3 and version 5".

After the walkthrough and short tasks, we replayed the sculpting video of the "hydra" model (~1hr) at 5× speed to the participants. We then asked them to author views of the model using viewpoint, comparison view, view path, and view surface, at least once.

We replayed the modeling process before the view authoring session to allow participants to get a sense of the modeler's workflow and the viewpoints that were used. We did not ask the participants to perform the 3D sculpting on their own, mainly because it could take too much time.

At the end of the study, an interview was conducted and the participants filled out a questionnaire regarding individual features of the system.

Evaluation Results and Discussions

Overall, the participants were enthusiastic about the system, and considered it a potentially useful component in a modeler's toolset. Participants were all able to complete the provided tasks without difficulty.

Figure 16 summarizes the subjective ratings of our system. Overall, responses were quite positive. When asked about their favorite functions, participants were enthusiastic about the access to previous versions, the comparison view function, and the time map. In particular, the comparison view function received very positive comments from the professional modelers. P2 commented that when looking for critiques from peer modelers or asking for help on detailed topology issues, it is common for 3D modelers to post before-and-after image pairs on a forum.

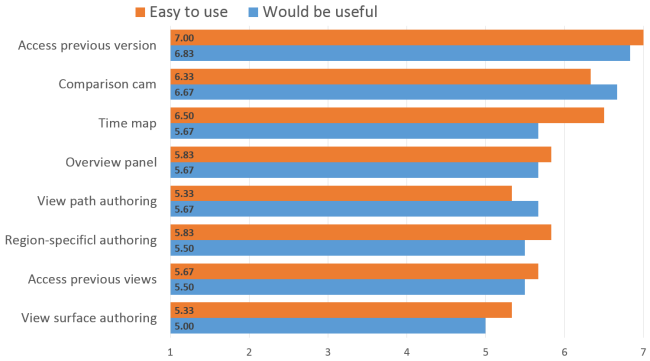

Figure 16. Subjective rating on system features.

One professional modeler (P1) particularly liked the time map function and commented that it would be very useful for workflow management. He stated that he usually works on multiple projects at the same time, and sometimes he might overlook some detailed regions on a 3D model. The time map function could serve as a good reminder about progress. Another modeler (P2) also commented that the combination of the time map and version slider could be very useful for an art director to track modelers' progress.

However, participants did express concern about learning so many new functions in such a short session. We believe this may have contributed to the lower scores for the view

authoring functions, as they were less familiar concepts to the participants. Still, participants did comment that the regions-specific authoring techniques would be very useful in certain scenarios. One 3D modeler (P1) commented that the region-specific authoring function is particularly useful for parts that are difficult to inspect, e.g. the tongue inside the mouth, and the architect (P6) commented that the automatic construction of local view paths would be a time-saver for detailed inspection.

Compared to their previous view authoring experiences in 3D modeling software, the responses were quite positive. Participants were excited about the additional temporal information provided by our system and thought it would be beneficial to have candidate views suggested during the authoring process.

DISCUSSIONS AND FUTURE WORK

We focus mainly on 3D digital sculpting users and tools. We are interested in extending our methodology to other categories of 3D modeling tools. Our work does rely on the assumption that the 3D modeling tool has been instrumented at a very detailed level. The internal data model in MeshMixer is a straightforward triangle mesh; however the geometric representation in other tools might be more complicated.

Currently our system is designed for scenarios where the modeler sculpts a single 3D object from scratch. However, some 3D objects or 3D scenes (such as mechanical assemblies or architectural models) incorporate a large number of objects or import 3D assets from a database. Our instrumentation needs to be extended to handle deletion and creation of parts, copy-and-paste, and so on. A merge algorithm that can handle editing histories from multiple objects might also be needed.

Finally, we focus our current method only on the editing and viewing history of a 3D model. However, this does not mean that the model geometry information should be ignored. We believe even more effective view selection and navigation systems can be designed by considering both the final model and its editing history. We hope our work can inspire future work in this fruitful research direction.

CONCLUSION

We propose a history assisted view authoring system for 3D models. Our system provides modelers the ability to traverse and author the camera views, paths, and surfaces across different model versions. We introduce the *region-specific view authoring* technique where candidate views are automatically calculated based on the specified surface region. At the core of the view authoring system is a unified algorithm framework for 3D model segmentation and automatic view selection based on the editing history. Our user studies show that the quality of the viewpoints generated by our algorithm is comparable to ones manually selected by the modeler, and the authoring system elicits a high level of interest from potential end-users.

REFERENCES

1. N. Burtnyk, A. Khan, G. Fitzmaurice, R. Balakrishnan, and G. Kurtenbach. (2002). Stylecam: interactive stylized 3d navigation using integrated spatial & temporal controls. *ACM UIST*. 101–110
2. N. Burtnyk, A. Khan, G. Fitzmaurice, and G. Kurtenbach. (2006). Showmotion: camera motion based 3d design review. *ACM I3D*. 167– 174
3. H.T Chen, L.Y Wei, and C.F Chang. (2011). Nonlinear revision control for images. *ACM Trans. Graph*. 30, 4, Article 105
4. P.Y Chi, S Ahn, A, Ren. B, Hartmann, M. Dontcheva, W. Li(2012). MixT: Automatic generation of step-by-step mixed media tutorials. *ACM UIST*. 93-102.
5. M. Christie and P. Olivier. (2009). Camera control in computer graphics: models, techniques and applications. *ACM SIGGRAPH ASIA Courses*. 3:1–3:197
6. J. D. Denning, W. B. Kerr, and F. Pellacini. (2011). Meshflow: Interactive visualization of mesh construction sequences. *ACM Trans. Graph*. 30, 4, Article 66.
7. J. D. Denning and F. Pellacini. (2013). Meshgit: diffing and merging meshes for polygonal modeling. *ACM Trans. Graph*. 32, 4, Article 35.
8. F. Grabler, M. Agrawala, W. Li, M. Dontcheva, and T. Igarashi. (2009) Generating photo manipulation tutorials by demonstration. *ACM Trans. Graph*. 28, 3, Article 66.
9. T. Grossman, J. Matejka, and G. Fitzmaurice (2010) Chronicle: Capture, exploration, and playback of document workflow histories. *ACM UIST*, 143–152.
10. A. Khan, B. Komalo, J. Stam, G. Fitzmaurice, and G. Kurtenbach. (2005). HoverCam: interactive 3D navigation for proximal object inspection. *I3D*. 73-80.
11. C. H. Lee, A. Varshney, and D. W. Jacobs. (2005). Mesh saliency. *ACM Trans. Graph*. 24, 3, 659–666.
12. R. Schmidt and K. Singh. (2010). MeshMixer: an interface for rapid mesh composition. *ACM SIGGRAPH Talks*, Article 6.
13. A. Secord, J. Lu, A. Finkelstein, M. Singh, and A. Nealen. (2011). Perceptual models of viewpoint preference. *ACM Trans. Graph*. 30, 5, Article 109.
14. A. Shamir. (2008). A survey on mesh segmentation techniques. *Computer Graphics Forum*, 27(6):1539–1556.
15. P. Shilane and T. Funkhouser. (2007). Distinctive regions of 3d surfaces. *ACM Trans. Graph*. 26, 2, Article 7.
16. K. Singh and R. Balakrishnan. (2004). Visualizing 3d scenes using non-linear projections and data mining of previous camera movements. *AFRIGRAPH*. 41–48.
17. M. Tsang, N. Morris, and R. Balakrishnan. (2004). Temporal thumbnails: rapid visualization of time-based viewing data. *AVI*. 175–178.