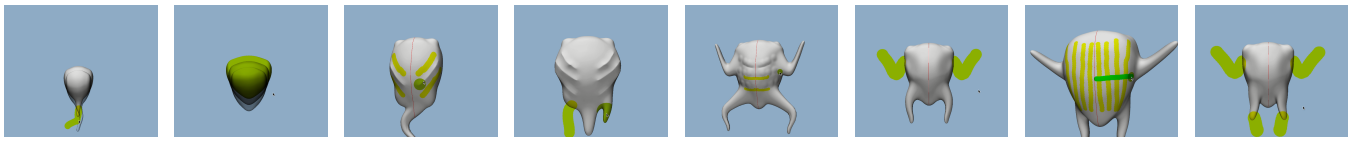
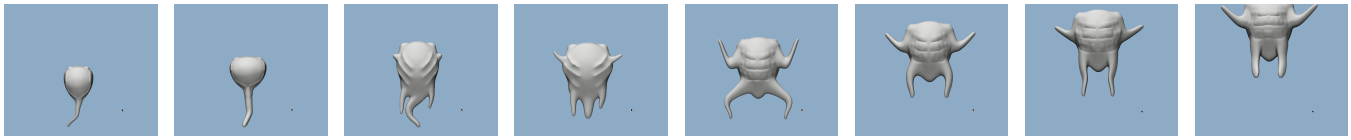
Autocomplete Animated Sculpting

Mengqi Peng[‡] **Li-Yi Wei**[†] **Rubaiat Habib Kazi**[†] **Vladimir G. Kim**[†]
[‡]University of Hong Kong [†]Adobe Research



(a) frame 2 (b) frame 4 (c) frame 6 (d) frame 7 (e) frame 8 (f) frame 10 (g) frame 11 (h) frame 12



(i) frame 2 (j) frame 4 (k) frame 6 (l) frame 7 (m) frame 8 (n) frame 10 (o) frame 11 (p) frame 12

Figure 1: *A tadpole-to-frog authoring sequence with our system.* After forming a plain base shape in the first frame, the user continues to sculpt and manipulate the following shapes with topological and motion changes. Along with user's manual editing, our system can predict what the user might want to do next, as indicated by transparent yellow suggestions for brush strokes (e.g. (a)) and manipulation operations (e.g. (b)). The user can also clone prior strokes as visualized via the green gesture (e.g. (g)). The final shapes with manual and autocompleted edits are shown at the bottom row. Please refer to the accompanying video for live actions.

ABSTRACT

Keyframe-based sculpting provides unprecedented freedom to author animated organic models, which can be difficult to create with other methods such as simulation, scripting, and rigging. However, sculpting animated objects can require significant artistic skill and manual labor, even more so than sculpting static 3D shapes or drawing 2D animations, which are already quite challenging.

We present a keyframe-based animated sculpting system with the capability to autocomplete user editing under a simple and intuitive brushing interface. Similar to current desktop sculpting and VR brushing tools, users can brush surface details and volume structures. Meanwhile, our system analyzes their workflows and predicts what they might do in the future, both spatially and temporally. Users can accept or ignore these suggestions and thus maintain full control. We propose the first interactive suggestive keyframe sculpting system, specifically for spatio-temporal repetitive tasks, including low-level spatial details and high-level brushing structures across multiple frames. Our key ideas include a deformation-based optimization framework to analyze recorded workflows and synthesize predictions, and a semi-causal global similarity measurement to support flexible brushing stroke sequences and complex shape changes. Our system supports a variety of shape and motion styles, including those difficult to achieve via existing animation systems, such as topological changes that cannot be accomplished via simple rig-based deformations and stylized physically-implausible motions that cannot be simulated. We evaluate our system via a pilot user study that demonstrates the effectiveness of our system.

Author Keywords

workflow, autocomplete, clone, sculpting, modeling, animation, user interface

CCS Concepts

•Human-centered computing → User interface design; •Computing methodologies → Shape modeling; Animation;

INTRODUCTION

Sculpting and animating shapes is a common form of content creation, and has found various applications among users with different expertise, such as tutorial and education for novice users, films and games production for expert users. Yet it remains challenging due to the

ACM ISBN 978-1-4503-6708-0/20/04 ...$15.00.
http://dx.doi.org/10.1145/3313831.XXXXXXX

required level of quality and diverse range of effects. Automatic methods such as physical simulation and procedural scripting can achieve high realism, but are limited to specific effects. Keyframe-based sculpting provides complete creative freedom, but requires significant manual labor and artistic expertise. In recent years, time-based or shot sculpting [35, 13] has emerged as a way to propagate manual keyframe edits over an imported animation cache. However, it is mostly a post-process for refining a reasonably complete animated model sequence, which the users still need to obtain via other means, such as simulation, scripting, and capture. As a typical example, users would need to sculpt a static model with ZBrush [41]), import it into Maya [5] for animation planning, then load the animation cache into Mush3D [35] for refinement. Therefore, the whole process requires high expertise and efforts working across multiple tools, imposing a steep learning curve for users.

The keyframe sculpting animation authoring workflows contain two main types of operations: brush (sculpting tools such as drag and clay) and manipulation (such as object translation and rotation). Repetitive and tedious tasks can occur within individual frames (spatial dimension) and across multiple frames (temporal dimension). We present an interactive system to help users, especially novices with limited animated sculpting experience, freely author keyframe sculpting animations with a suite of interactive autocomplete tools, without requiring any external system or data. Our interface functions like traditional desktop and immersive VR keyframe brushing systems [5, 35, 13, 17], with which users can freely brush spatial structures and their temporal changes. Meanwhile, our system records and analyzes their workflows and predicts what they might do in the future, both spatially and temporally, and in real-time. As exemplified in Figure 1, users can brush broad strokes in a new frame to indicate the overall structure, from which our system can predict what they might do next, a feature we refer to as *brush hint*. They can further edit completed frames and automatically propagate the edits to other frames via *edit propagation*, and manually clone workflows from one spatial-temporal region to another via *workflow clone*. Users can accept, partially accept, or ignore these suggestions thus maintain full control.

Similar to existing sculpting tools, we focus on soft/organic instead of hard/CAD-like objects with expressive brushing interface and diverse effects. Our system assists users with repetitive tasks via our autocomplete features for a variety of shape and motion styles, including those difficult to achieve via existing animation systems, such as topological changes (tree growing, cell mutation, frog metamorphosis, and other organic transformations) that cannot be accomplished via simple rig-based deformations and stylized physically-implausible motions that cannot be simulated. It can also interpolate or extrapolate new frames for inbetween and prediction by considering both sculpting strokes and manipulation operations, to produce smoother shape/motion transitions than pure geometry-based interpolations/extrapolations, especially for motion sequences with topological changes.

Method-wise, our main idea is to extend static sculpting [39] towards the temporal dimension, analogous to how 2D hand-drawn animation [53] extends 2D drawing [51]. Specifically, [51] analyzes 2D sketching workflows to provide online painting suggestions, and [53] extends [51] for 2D hand-drawn animations. Similarly, [39] analyzes 3D sculpting workflows to provide brushing strokes predictions, and we extend [39] towards the temporal dimension for 3D sculpting animations. However, a direct extension would not work, due to several fundamental differences between 2D sketching and 3D sculpting. In 2D sketching, brush strokes usually indicate straightforward first-order and final outcomes (stroke to stroke), whereas in 3D sculpting, first-order brush strokes can indicate indirect controls for second-order mesh deformations (stroke to deformation), and later edits can change earlier results. 2D sketching can be placed anywhere in the canvas and thus the predicted strokes are only constrained by prior strokes, while 3D sculpting strokes should start from (*freeform brush*) or adhere to (*surface brush*) the mesh surface in the sculpting canvas. Furthermore, the core method in [53] assumes that the users would follow coherent temporal drawing orders across frames, which might not hold for more general and complex 2D animations. This will become even more so for 3D sculpting due to viewpoint changes and invisible/intermediate strokes often in different temporal orders across frames. Our core algorithm can match sculpting strokes with different explicit properties such as spatial length or temporal order and yet have similar implicit user intentions, and provide more meaningful predictions. In addition to sculpting strokes, this also applies to manipulation operations.

We evaluate our prototype system with a pilot user study, to collect user feedbacks and their animated models. The evaluation indicates that our system can reduce input workload and enhance output quality without restricting the artistic freedom and achievable range of effects.

In summary, the main contributions of this paper include:

- The first suggestive keyframe-based sculpting system that can help users, including novices, create animated models from scratch without requiring external tools or data.
- An intuitive user interface with a set of interactive autocomplete tools (hint, edit propagation, workflow clone, and interpolation/extrapolation) that assist in repetitive tasks across different scales and dimensions.
- A deformation-based optimization framework to analyze recorded temporal sculpting workflows and synthesize predictions.
- A semi-causal across-frames global similarity measurement to support flexible brushing stroke sequences and complex shape changes.

- A pilot user study to evaluate our prototype and create various keyframe sculpting animation sequences.

RELATED WORK

3D animation remains a tedious and challenging task that involves two key aspects: shape and motion. To assist this task, different prior methods are proposed.

Keyframe animation

Keyframe-based animation is a well-established tradition, with manually authored keyframes followed by inbetweening. Keyframing interface provides full control but can be quite tedious to author from scratch. To assist shape posing, various approaches have been proposed, including skeletons and rigs [8, 21], cages [45], stick figures [14], and blend shapes [29]. Keyframe animation has also been recently introduced into VR, such as Quill [17]. Our system inherits this traditional and popular interface design, but maintains the brush-based workflow which tends to be more intuitive than requiring these extra manipulators.

Sketch-based animation

2D sketches can be leveraged for intuitive 3D animation authoring, such as [27, 19, 12]. However, due to depth, these systems focus more on drafting overall movement, may require projective constraints, and provide less fine-grained shape and motion controls. Recent advancements in VR can provide more intuitive sketching gestures for animation authoring [3]. Instead of asking users to provide explicit sketches, our system takes brushing strokes as implicit manipulators to support both coarse (frame-level) and fine-scale (stroke-level) controls, including motion transitions and shape changes.

Suggestive authoring interfaces

Suggestive interfaces can help reduce input efforts and enhance output quality without disrupting user workflows, and have been applied to 2D sketching [33, 28, 51], 2D animations [53, 38], 3D modeling [18, 39, 48, 52], and video-guided VR content creation [49]. Our system follows this route to design a suggestive authoring interface but aims at keyframe sculpting animations, including both shape and motion suggestions. Like 2D stylized animations [25, 24, 26], our scope is freeform stylized animations instead of physically realistic animations. Inspired by prior planar animations works such as [55], we also aim at supporting topology changes with exaggerated deformations across frames, but without requiring external annotations or correspondences for the input to preserve users' natural flow.

OBSERVATIONS AND DESIGN GOALS

In this section, we analyze existing tools to understand the nature and practice for keyframe sculpting. We investigated and studied existing tools with online tutorials, authoring sessions/results, and users' discussions on current workflows in different forums. Furthermore, we also interviewed an expert animator (P_0) with seven years of experience with keyframe sculpting to validate our observations. This helped us better understand the pain points and limitations of existing workflow and motivate our design goals.

Existing tools and workflows

Sculpting tools [41] provide intuitive and freestyle brushing interaction to create organic models for a large community of users with different expertise. However, in keyframe-based sculpting, the temporal dimension adds significant complexity to the task - "it becomes much harder to create sculpting animations" - (P_0).

Desktop time-based sculpting tools [13, 35] facilitate intuitive keyframe-based sculpting, but usually require combining of other tools for the entire process [41, 4, 5]. Regarding existing pipeline, P_0 commented: "Current pipeline is mature for commercial large project production with experienced team collaborations, but I think it is overly complicated for fast prototyping and concept illustration. It is also not friendly for novice users or less-skilled animators, given the steep learning curve and needed tricks".

VR brushing animation such as [17] does not require rigging but relies on direct manipulation and brushing to create frame-by-frame animations from scratch, and thus is more intuitive and straightforward for users. However, current VR animation interfaces focus on manual creation of low-resolution and rigid strokes instead of detailed organic objects that can be more easily created in existing desktop tools such as [5].

System and workflow studies also suggest that the tedious brushing strokes and repetitions for 3D sculpting [44] and the multi-departments collaboration for animation production [56], are among the significant reasons that make sculpting animation time-consuming and challenging.

Design goals

Based on these observations and insights, we have formulated the following design goals for our system:

Efficiency We employ a mixed-initiative interface [22] with novel autocomplete tools, providing computational assistance to spatio-temporal tasks while retaining user control and flexibility. We believe that such an efficient workflow would also make keyframe sculpting more accessible to a broader audience.

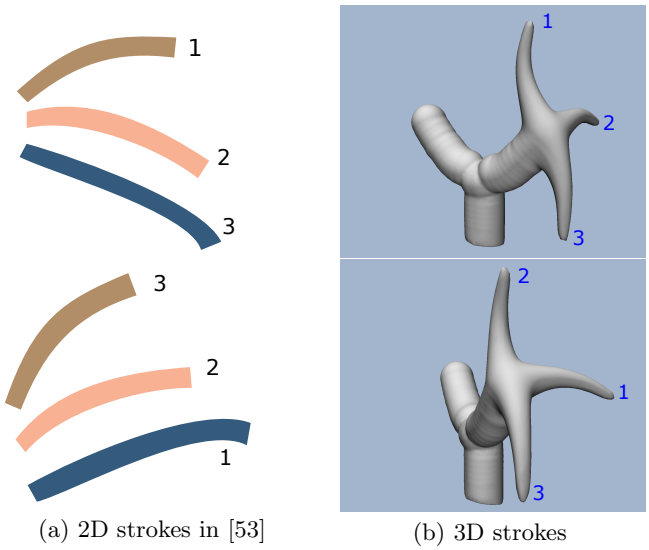
Flexibility Like [13, 35], our system mainly targets at high-resolution organic sculpting animation via ZBrush-like [41] freestyle brushing.

Coherence Like [17], our system is designed to be intuitive and standalone, instead of requiring switching among multiple tools and techniques. We aim to present a new possibility, rather than a replacement, for traditional multi-stage pipelines. Our system is deployed at a desktop environment but the method and features could be applied for VR and other platforms.

Comparing 2D/3D Brushing Workflows

Inspired by predictive 2D sketching animation [53], we aim at a keyframe-based suggestive animation system by recording and analyzing users' workflows.

For static sketching or sculpting, the temporal and spatial editing is often local and coherent [51, 39]. For animation, [53] follows coherent drawing assumptions, which means the workflows such as drawing order (e.g., top-down or bottom-up) and strokes number should be mostly similar for neighboring frames. However, in practice, it is rarely the case [34]. Such an assumption cannot generalize for more realistic and complex animation [23]. This will become even more so for 3D sculpting due to several major differences from 2D sketching detailed below.



(a) 2D strokes in [53] (b) 3D strokes

Figure 2: *Stroke coherence in 2D sketching and 3D sculpting.* The prediction in [53] assumes similar brushing orders across frames and thus could not handle the situation depicted in (a). We relax their assumptions to support more general sculpting practice (b).

3D viewpoint adjustments

If users need to follow the temporal coherent editing assumption, when they are editing the current new frame, besides the brush strokes order, they also need to remember the intermediate viewpoint adjustment history, which is not practical for 3D sculpting as illustrated in Figure 2.

Visible shape changes from invisible user strokes

Sculpting strokes are invisible, what users can visualize are the mesh shape deformations via different brush strokes. With onion-skinning shadow of the previous frame, users can reproduce the content to have similar spatial shape appearance but not necessarily with consistent stroke numbers and orders.

Sculpting strokes preference

Users may use a single long stroke or several short overlapping strokes to achieve similar end effects. This means

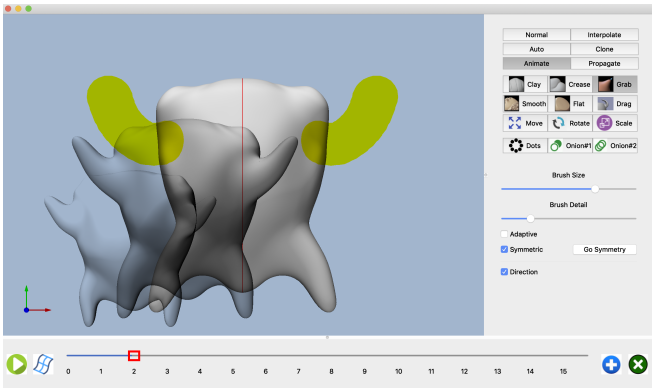


Figure 3: *UI of our system.* Our UI contains the main sculpting canvas (left), a tools widget panel (right), and an animation timeline (bottom). Users place sculpting brushes and manipulate the objects on the main canvas with tools selected from the right widget, and control the keyframe via the timeline frame slider. The main canvas shows a sculpt object in the current frame, and users can decide whether to display the onion skin of the previous frame(s) (rendered in grey with different transparency) for guidance. Our system can predict what the users might want to brush or manipulate next (rendered in transparent yellow) based on recorded workflows.

that a similar feature can be accomplished via different sculpting preferences across frames.

Therefore, a direct extension of [53] will not suffice. Besides manual strokes, another key difference lies in the validity of the predicted strokes, as detailed below.

2D prediction

In 2D, brush strokes directly formulate the visible/final results on the canvas. The predicted strokes are only constrained by prior strokes. The content will be the same as the predicted strokes once accepted, regardless of the prediction quality. There are looser connections among strokes, e.g. strokes do not have to be precisely connected with one another.

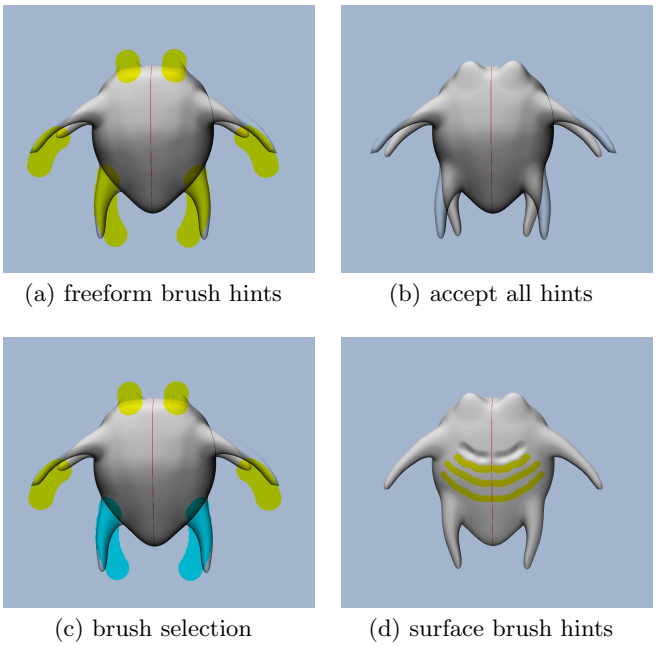
3D prediction

In 3D, brush strokes are the first-order inputs placed by users, what exactly will be generated are second-order mesh deformations. Different from 2D strokes which are loosely constrained by the planar canvas, meaningful and valid 3D sculpting strokes have to start from (*freeform brush*) or adhere to the mesh surface (*surface brush*). And sculpting strokes can carry semantic structural connection meanings, e.g. connected shorter freeform strokes to produce elongated shape deformation.

Hence, we need to consider extra sculpting context to design the prediction algorithm.

USER INTERFACE

The UI of our prototype sculpting animation system, as shown in Figure 3, combines the brush models as in



(a) freeform brush hints (b) accept all hints

(c) brush selection (d) surface brush hints

Figure 4: *Brush hint.* Along with the user's brushing, our system displays both the onion skin of the previous frame (rendered in transparent grey) and the brushing hints (either freeform (a) or surface brush hints (d)) for the current frame (rendered in transparent yellow). The user can accept all hints via the hotkey (b), or select partial hints (c) (selected brushes are rendered in transparent blue), or ignore the hints by continuing brushing.

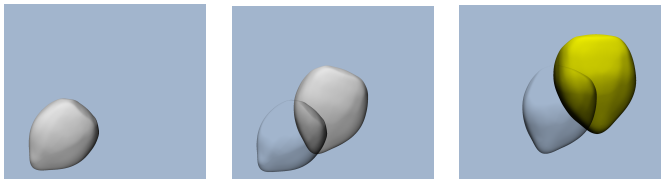
popular digital sculpting tools such as ZBrush [41] and the keyframe/timeline-based digital animation tools such as in Maya [5]. Below we describe the overall process and main features of our system.

Overall process

Starting from the default or imported base shape, users sculpt and manipulate the object in the current frame, and control the timeline to add new frames or edit an existing frame. Meanwhile, our system records and analyzes their workflows, including sculpting brushes and manipulation operations, to provide online suggestions via a set of interactive autocomplete tools. Our system follows the spatial-temporal sculpting brushing behaviors and assumptions to support general sculpting [44, 15]. Some authoring examples are shown in Figure 1 and the supplementary video.

Sculpting and manipulation tools

The surface and freeform brush strokes perform various sculpting operations as in popular sculpting systems (such as clay, drag, grab, crease) to create low-level details and deformations. The manipulation operations (such as rotation, translation, and scaling) facilitate the creation of the overall motion of the object.



(a) frame $F_{\kappa-2}$ (b) frame $F_{\kappa-1}$ (c) hint for F_{κ}

Figure 5: *Manipulation hint.* The user can perform continuous manipulation operations across the frames. Subsequent rotation, scaling, and translation operations occur from $F_{\kappa-2}$ in (a) to $F_{\kappa-1}$ in (b). When adding a new frame F_{κ}, our system suggests the initial shape (rendered in yellow) in (c). The user can accept the manipulation hint via the hotkey or ignore the suggestion.

Interactive autocomplete tools

Following our observations and design goals, we provide a set of interactive autocomplete tools. One tool will be active at a given time. Users can press the button on the right widget panel to select the tool. For all suggestions on the interface, users can fully accept via the hotkey A, partially accept via the brushing selection gesture, or ignore by continuing brushing, thus maintain full control.

Brush hint

Repetitive brushing operations can occur within an individual frame and across multiple frames. Manually performing these operations can be very tedious. Therefore, along with user editing, our system provides brushing stroke hints exemplified in Figure 4. With more strokes, the hints are updated accordingly, including coarser-scale structures (e.g. drag brush) and finer-scale details (e.g. clay brush) for both spatial repetitions within one frame and temporal repetitions across neighboring frames.

Manipulation hint

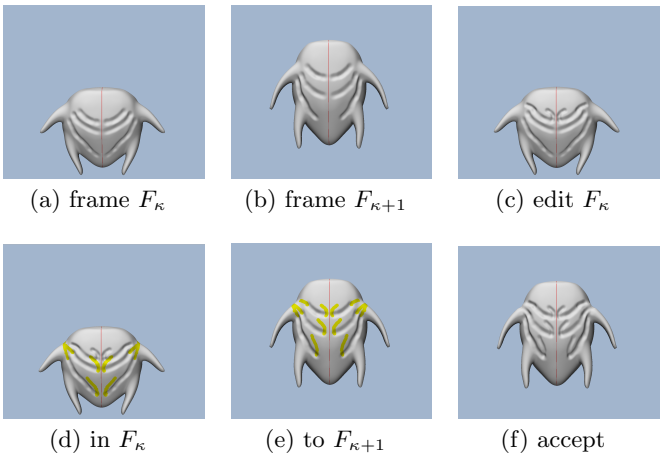
Manipulation operations, such as translation, rotation, and scaling, control the coarser scale object positions and motions across frames. Such manipulation operations can be cumbersome for continuous shape and motion posing across multiple frames. Our system also records and analyzes these manipulation operations and provides suggestions such as extrapolations for bootstrapping new keyframes, as shown in Figure 5.

Edit propagation

Keyframe sculpting involves both coarse and fine scale editing, often in an iterative fashion. Repeating details across frames with coarse edits in shape or motion is extremely tedious. In a 3D interface, the constant viewpoint changes [10] make the process even more cumbersome. The *edit propagation* tool, shown in Figure 6, automates finer-scale additive strokes performed in one frame across multiple spatial regions and temporal frames.

Workflow clone

Clone is a common tool among different interactive content authoring systems, such as image regions [40]. The



(a) frame F_κ (b) frame $F_{\kappa+1}$ (c) edit F_κ

(d) in F_κ (e) to $F_{\kappa+1}$ (f) accept

Figure 6: *Edit propagation.* After manipulating the overall spatial transformation and sculpting initial outlining shapes in (a) and (b), the user starts to add details in one frame (c). Our system can automatically propagate the edits (rendered in transparent yellow), both within the current frame (d) and across multiple frames (e). (f) shows the shape after accepting the suggestions in (e).

method in [51] extended the clone to consider not only static image pixel data but also dynamic 2D sketching workflows, and [39] further considered 3D sculpting workflows. Our system allows users to clone brushing workflows across frames. With our brushing interface, users can specify source and target regions, as well as customized parameters such as orientation and brush sizes. Figure 7 shows an example.

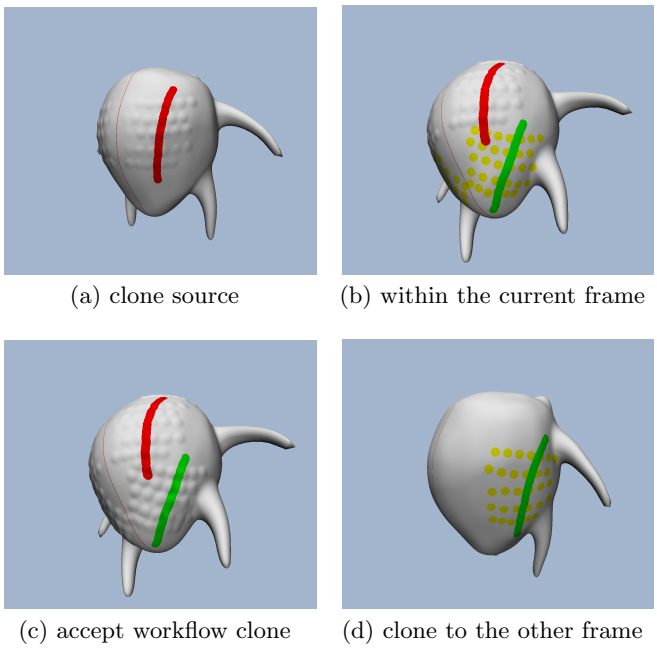
Workflow interpolation

Manually authored keyframes are often too sparse for smooth motion and require the addition of inbetweens, whose generation has been a main focus of research in graphics and image processing [42, 50]. To assist the outputs of smooth sculpting animation, we extend traditional inbetweening methods to consider not only mesh objects but also user workflows. Figure 8 shows an example.

METHOD

Our system analyzes the recorded workflows across frames to synthesize online predictions. We illustrate our method with concrete brush strokes in Figure 9. In order to predict what strokes might be performed for current frame F_κ, we deform strokes in the previous frame $F_{\kappa-1}$ to match the current user strokes in F_κ. Intuitively, user strokes in F_κ act as implicit deformation handles to decide how to deform strokes in $F_{\kappa-1}$ for prediction. To achieve this deformation, we match strokes across frames via a global registration framework, whose core lies in a global similarity to dynamically measure the similarities and update the correspondences.

Our global measurement can capture large, complex shape deformations and movements across frames, such



(a) clone source (b) within the current frame

(c) accept workflow clone (d) clone to the other frame

Figure 7: *Workflow clone.* Users can clone the source in one frame via a red gesture brush (a), to other target regions, either within the current frame (b) or other frames (d) via a green gesture brush. They can re-apply the target gesture if not satisfy with the cloning preview (rendered in transparent yellow). (c) shows the shape if accepting the suggestions for (b).

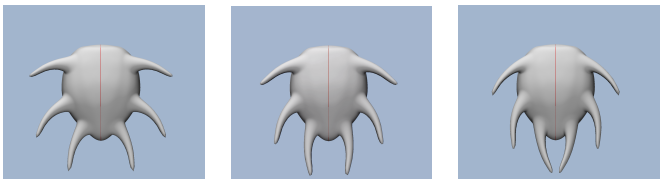
as tree growing, virus mutating, octopus moving, and tadpole-to-frog metamorphosis. This could be viewed as an extension of [39] which only detects local similarities to provide suggestions for single object sculpting. Our measurement can also handle general sculpting practice such as different number and order of strokes across frames.

Let us start with brush representation that will be used in the remaining sections. We represent each brush stroke b as a set of samples $\{s\}$. Each sample s is associated with a set of attributes $u(s)$:

$$u(s) = (p(s), a(s), t(s), m(s)), \tag{1}$$

where $p(s)$ records the sample position; $a(s)$ is a set of appearance parameters such as brush type, radius, pressure, and normal; $t(s)$ indicates temporal parameters including the global frame index, the parent brush stroke index, and the sample index for the relative position within its parent stroke; $m(s)$ records the local deformation across frames.

In the remainder of this section, we will overview our brushing prediction method. We formulate the prediction as an energy-minimization problem, where we optimize for the predicted brushes as a deformation of brushes in $F_{\kappa-1}$ towards F_κ. In order to formulate our energy, we need to find similar brush strokes across frames. For this, we propose the global similarity measurement and sample



(a) F_κ (b) interpolation (c) $F_{\kappa+1}$

Figure 8: *Workflow interpolation.* With user-created keyframes such as (a) and (c), our system can perform workflow-driven interpolations to generate their inbetween frame in (b).

matching techniques. Finally, we describe how our brush prediction could be used for deploying various interactive features on the interface. Algorithm 1 illustrates the

procedure $\{b_\kappa\} \leftarrow \underline{Predict}(\mathbf{I})$
  //$\{b_\kappa\}$: predicted remaining strokes for F_κ
  //$\mathbf{I}$: input keyframes $F_{\kappa-1}$ and F_κ
  $\mathbf{I}$ with deformation model Equations (2) to (5)
  $\mathscr{E}_{rigid} \leftarrow$ Equations (7) and (8)
  $\mathscr{E}_{reg} \leftarrow$ Equation (9)
  $\mathscr{E}_{fitness} \leftarrow$ Equation (10)
  $\mathscr{E}_{conf} \leftarrow$ Equation (11)
  $\mathscr{E}_{sculpt} \leftarrow$ Equation (12)
  $\mathscr{E} \leftarrow$ Equation (6)
  $\{b_\kappa\} \leftarrow \underline{Optimize}(\mathscr{E})$
end procedure

procedure $\underline{Optimize}(\mathscr{E})$
  solver $\leftarrow$ Levenberg Marquardt method
  update sample correspondence of Equation (10)
    backward deform ($F_\kappa \rightarrow F_{\kappa-1}$) via ICP [30, 47]
    forward deform ($F_{\kappa-1} \rightarrow F_\kappa$) via Equation (16)
end procedure

Algorithm 1: *Overview of keyframe prediction algorithm.*

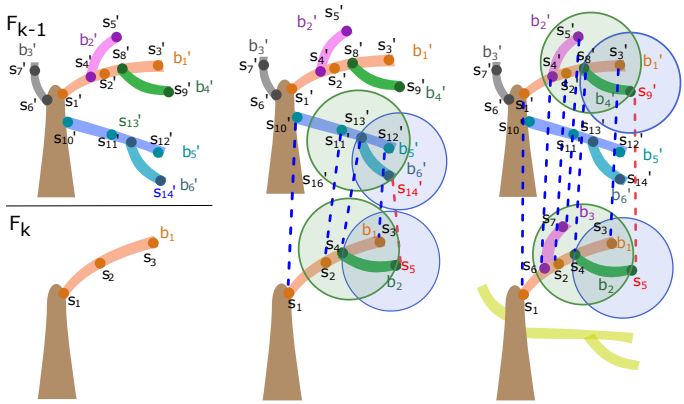
overall flow of our method.

Brush prediction

Sculpting strokes are often coherent across adjacent frames. Thus, strokes in $F_{\kappa-1}$ can guidestrokes in F_κ (Figure 9a). With strokes in F_κ as implicit control handles, we deform strokes in $F_{\kappa-1}$ to predict what else might be performed for F_κ (Figure 9c). We design a deformation framework tailored for keyframe sculpting with both deformation and sculpting constraints. We customize the optimization process to achieve interactive high-quality predictions. Figure 10 illustrates the basic ideas and steps.

Deformation model

Let us start with our deformation model. The deformation parameter $m(s^{\kappa-1})$ in Equation (1), inspired by [47, 53], records the local deformation at s from $F_{\kappa-1}$ to F_κ, via a 3×3 matrix A and a 3×1 translation vector δ.



(a) first stroke (b) more strokes (c) prediction

Figure 9: *Algorithm overview via an example.* The user authors $F_{\kappa-1}$ (top) followed by F_κ (bottom). As the user places a new stroke b in F_κ, our system analyzes how well it matches to each stroke b' in $F_{\kappa-1}$ via their constituent samples $\{s\}/\{s'\}$. These matchings are often ambiguous during the early stages with only a few strokes. For example, with only two strokes in (b), s_5 could match with s'_{14} better than s'_9. With more strokes in (c), the updated global similarities become more robust to suggest s'_9 for s_5. With these matched strokes ($b_1 \leftrightarrow b'_1$, $b_2 \leftrightarrow b'_4$, $b_3 \leftrightarrow b'_2$), we optimize the prediction framework (Equation (6)) to deform the remaining strokes $b'_{3,5,6}$ from $F_{\kappa-1}$ to F_κ for predictions, visualized in yellow (c).

Under the influence of sample $s_i^{\kappa-1}$, the nearby sample $s_j^{\kappa-1}$ is deformed to a new position $q_{ji}^{\kappa-1}$:

$$q_{ji}^{\kappa-1} = A_i(p_j^{\kappa-1} - p_i^{\kappa-1}) + p_i^{\kappa-1} + \delta_i^{\kappa-1} \qquad (2)$$

$s_j^{\kappa-1}$ can be influenced by multiple neighbor samples, thus the accumulated local deformation for $s_j^{\kappa-1}$ is defined by:

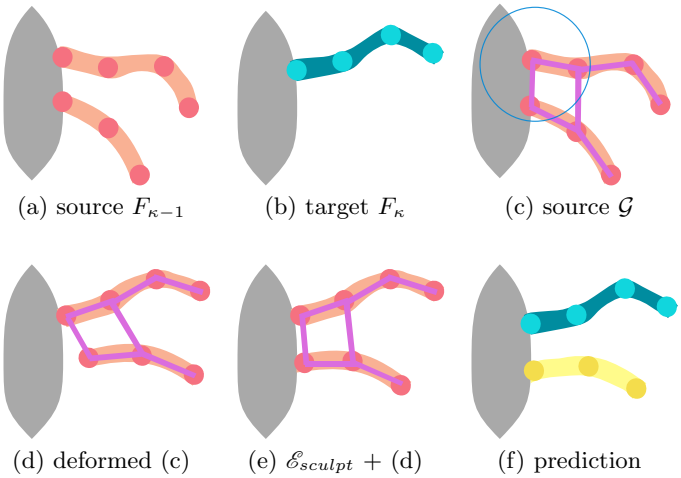
$$q_j^{\kappa-1} = \sum_{s_i \in n(s_j^{\kappa-1})} \omega_i(s_j^{\kappa-1}) q_{ji}^{\kappa-1}, \qquad (3)$$

where $n(s)$ is the local sample neighborhood of s and will be detailed in the next subsection. The weighting $\omega_i(s_j^{\kappa-1})$ is computed via:

$$\omega_i(s_j^{\kappa-1}) = \frac{1 - \left\|p_j^{\kappa-1} - p_i^{\kappa-1}\right\| / \Theta}{\sum_{m=1}^{\ell} 1 - \left\|p_j^{\kappa-1} - p_m^{\kappa-1}\right\| / \Theta}, \qquad (4)$$

where Θ is the distance to the ℓ-nearest sample, the parameter ℓ will be detailed in the implementation section.

Sculpting strokes/samples deformation across frames should contain global transformation in addition to local deformations. Inspired by [30], we enhance the model with a global rigid transformation defined by a rotation matrix $\mathcal{R}$ and a translation vector $\mathcal{T}$. $\mathcal{R}$ is relative to the center-of-mass χ of the source samples. Thus the locally



(a) source $F_{\kappa-1}$ (b) target F_κ (c) source $\mathcal{G}$

(d) deformed (c) (e) $\mathscr{E}_{sculpt}$ + (d) (f) prediction

Figure 10: *Prediction deformation graph example.* We take $F_{\kappa-1}$ as the source (a) and F_κ as the target (b), with strokes augmented with a graph-based deformation model (c). With user input strokes in F_κ as implicit deformation handles, we deform the source graph $\mathcal{G}$ to predict remaining strokes in F_κ. The deformation constraints alone (Equations (7) to (11)) might not produce valid deformation (d), which can be enhanced by extra sculpting constraint (Equation (12)) shown in (e). The deformed source strokes are visualized as transparent yellow hints on the user interface (f).

deformed $s_j^{\kappa-1}$ is further transformed according to:

$$\varrho_j^{\kappa-1} = \mathcal{R}(q_j^{\kappa-1} - \chi) + \chi + \mathcal{T} \tag{5}$$

Constraints

We formulate the prediction issue as an optimization problem of deforming brushes in $F_{\kappa-1}$ towards F_κ. We assume that similar brushes/samples across frames have been determined as described in the next subsections.

To guide the optimization, we include various constraints:

$$\mathscr{E} = \alpha\mathscr{E}_{rigid} + \beta\mathscr{E}_{reg} + \gamma\mathscr{E}_{fitness} + \eta\mathscr{E}_{conf} + \xi\mathscr{E}_{sculpt} \tag{6}$$

The first four are deformation constraints extended from prior registration methods [47, 30, 53]) while the last constraint is new and tailored for digital sculpting. We detail these individual constraints as follows.

$\mathscr{E}_{rigid}$ penalizes the deviation of each sample transformation from a pure rotation, making the local deformations become as rigid as possible to avoid artifacts:

$$\mathscr{E}_{rigid} = \sum_{s_i^{\kappa-1} \in \kappa-1} Rot\left(A\left(s_i^{\kappa-1}\right)\right) \tag{7}$$

$$\begin{aligned} Rot(A) = (a_1^T a_2)^2 + (a_1^T a_3)^2 + (a_2^T a_3)^2 + \\ (1 - a_1^T a_1)^2 + (1 - a_2^T a_2)^2 + (1 - a_3^T a_3)^2, \end{aligned} \tag{8}$$

where $a_{1,2,3}$ are the column vectors of A.

$\mathscr{E}_{reg}$ serves as a regularizer for the deformation, expecting adjacent samples to agree with one another:

$$\mathscr{E}_{reg} = \sum_{s_i^{\kappa-1}} \sum_{s_j^{\kappa-1} \in n(s_i^{\kappa-1})} \left| q_{ji}^{\kappa-1} - q_{jj}^{\kappa-1} \right|^2 \tag{9}$$

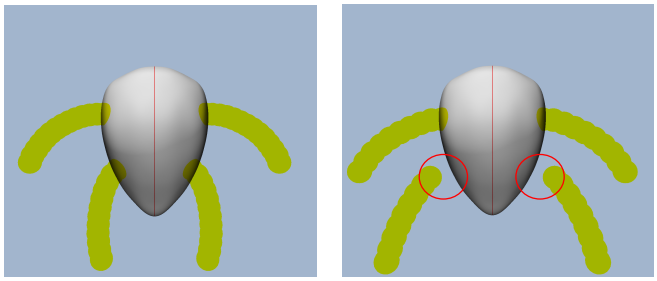
$\mathscr{E}_{fitness}$ pulls the deformation from $F_{\kappa-1}$ towards desired positions in F_κ:

$$\mathscr{E}_{fitness} = \sum_{s_i^{\kappa-1}} \varphi_i^2 \left| \varrho_i^{\kappa-1} - p_i^\kappa \right|^2, \tag{10}$$

where s_i^κ is a sample on a user-deformed stroke in F_κ with position p_i^κ, matched to sample $s_i^{\kappa-1}$ in $F_{\kappa-1}$ with a deformed position $\varrho_i^{\kappa-1}$ according to Equation (5), φ_i is the correspondence confidence weight discussed below.

$\mathscr{E}_{conf}$ targets at maximizing the reliable sample correspondence in Equation (10), where the values of φ_i close to 1 suggest a reliable correspondence via our sample matching, while values close to 0 indicate that no proper matching is determined:

$$\mathscr{E}_{conf} = \sum_{s_i^{\kappa-1}} (1 - \varphi_i^2)^2 \tag{11}$$



(a) valid strokes (b) invalid strokes (red circles)

Figure 11: *3D sculpting strokes semantics.* Only valid predicted strokes can deform the target mesh correctly. Predicted strokes in (a) can deform the shape correctly, while the bottom strokes in (b) are invalid due to disconnections from the target.

The prediction [53] on static 2D canvas only needs to consider brushing strokes. However, with only the above deformation constraints, it is unlikely to produce valid predictions for dynamic objects, as illustrated in Figure 11b. Thus we have specifically designed a sculpting constraint to consider the target shape. Surface strokes adhere to the mesh surface, while freeform strokes start from the surface and grow into the space. Thus for freeform strokes, we include $\mathscr{E}_{sculpt}$ to push the starting samples $\{\tilde{s}^{\kappa-1}\}$ in $F_{\kappa-1}$ to adhere to mesh surface in F_κ:

$$\mathscr{E}_{sculpt} = \sum_{s_i^{\kappa-1} \in \tilde{s}^{\kappa-1}} \left| \varrho_i^{\kappa-1} - \check{p}_i^\kappa \right|^2, \tag{12}$$

where $\check{p}^\kappa$ is the corresponding freeform on-surface sample position for $s_i^{\kappa-1} \in \tilde{s}^{\kappa-1}$. To compute $\{\check{p}^\kappa\}$, we

ignore the non-surface samples of freeform strokes and consider only the on-surface samples to perform surface registration from $F_{\kappa-1}$ to F_κ. The surface registration is performed in the geodesic surface parameterization space [39], and the surface results are then reconstructed back to 3D for $\{\tilde{p}^\kappa\}$.

Sample similarity measurement

Similar to other registration methods [47, 30], our prediction quality highly depends on the accuracy of correspondences between source and target objects, i.e. the dashed line connections for matched samples in Figure 9. For example, with more accurate correspondences from source (Figure 10a) to target (Figure 10b), the input brushing in the target frame will contribute more as implicit deformation handles, which will make the underlying deformation more reliable. However, unlike traditional registration applications, interactive sculpting can have incomplete information (i.e. strokes) in the current frame, and brushing practice can vary among users and authoring stages. To address these challenges, we propose a global similarity measurement to dynamically determine and update correspondences between $F_{\kappa-1}$ and F_κ. Specifically, we measure the similarity based on their neighbors as in the neighborhood-based texture synthesis optimizations [39, 53, 31].

In the remainder of this section, we will start from the local similarity measurement, then proceed to our global one, and finally highlight the core difference from [53].

Sample differential

The differential between s_a and s_b is represented as:

$$\widehat{u}(s_a, s_b) = (\omega_p \widehat{p}(s_a, s_b), \omega_a \widehat{a}(s_a, s_b), \omega_t \widehat{t}(s_a, s_b), \omega_m \widehat{m}(s_a, s_b)), \quad (13)$$

where $\widehat{p}$, $\widehat{a}$, $\widehat{t}$, $\widehat{m}$ represent the sample pair differentials defined in Equation (1), and all are calculated w.r.t. s_b. We compute $\widehat{p}$ via the stroke parameterization as in [39], and other differentials $\widehat{a}$, $\widehat{t}$, $\widehat{m}$ via direct numerical difference (i.e., scalar or vector difference). Corresponding weighting parameters ω_p, ω_a, ω_t, and ω_m are summarized in the implementation section.

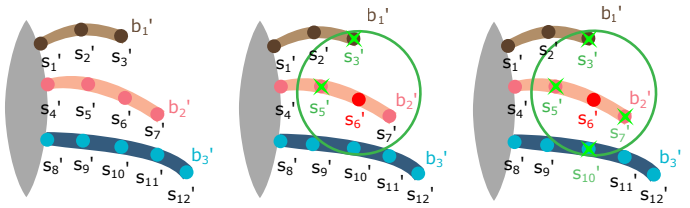
Local neighborhood similarity

We define the local neighborhood $n(s)$ to contain samples of the same brush type with s and fall within its spatial vicinity in the same frame. Unlike [53], we do not enforce $n(s)$ to be temporally causal to gain broader spatial context, as compared in Figures 12b and 12c.

We measure the similarity between $n(s')$ and $n(s)$ by summing up their paired sample differentials:

$$\Delta(s', s) = \sum_{s_j \in n(s)} exp\left(-\left|\widehat{u}(s'_j, s') - \widehat{u}(s_j, s)\right|^2\right), \quad (14)$$

where s_j iterates through $n(s)$, and $s'_j \in n(s')$ pairs with s_j. We follow [53] to measure the similarity via the exponential sample differential with a consistent range;

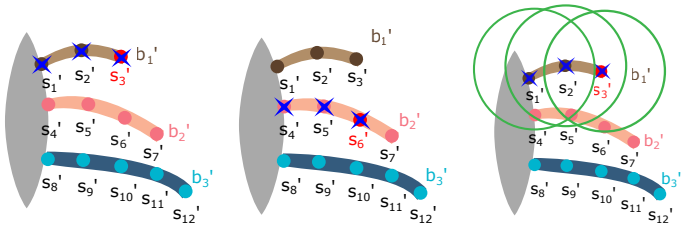


(a) strokes/samples (b) $n(s'_6)$ [53] (c) our $n(s'_6)$

Figure 12: *Brush strokes/samples and local sample neighborhood.* Frame $F_{\kappa-1}$ in (a) contains ordered strokes $b_{1,2,3}$, and samples along each stroke, e.g. $s'_{1,2,3}$ for b'_1. The left grey regions indicate the base shapes. The neighborhood $n(s'_6)$ via [53] in (b) contains only samples placed before ($s'_{3,5}$, marked with green stars). Ours in (c) relaxes the temporal restriction but emphasizes more spatial vicinity, to include $s'_{3,5,7,10}$.

consequently the similarity will be larger with a smaller differential. We determine the pairings by first identifying the pair (s'_j, s_j) with the largest similarity and then exclude them from further consideration, and repeat the process to find the next pair until $n(s)$ runs out of samples.

Global neighborhood similarity



(a) $v(s'_3)$ (b) $v(s'_6)$ (c) $\{n(v(s'_3))\}$

Figure 13: *Predecessor samples and global semi-causal sample neighborhood.* The predecessor samples (marked with blue stars) of s'_3 in (a) and s'_6 in (b) include $s'_{1,2,3}$ and $s'_{4,5,6}$ respectively. Our global semi-causal sample neighborhood N of s'_3 in (c) starts from itself and only goes back along with its predecessor samples. Thus $N(s'_3) = \{n(v(s'_3))\} = s'_{1,2,3,4,5,6}$.

In order to better capture the sample semantics among strokes, we define the predecessor samples $v(s)$ as those sculpted at or before s along the parent stroke. $v(s)$ is temporally causal and remains fixed once computed, as in Figures 13a and 13b.

The local sample neighborhood difference in Equation (14) works on individual samples. In order to capture broader context among strokes, we design a global semi-causal sample neighborhood N as the accumulation of predecessor samples' neighborhoods, as illustrated in Figure 13c. $N(s)$ combines the temporally-noncausal local sample

neighborhood centered at the temporally-causal predecessor samples, thus we call it semi-causal.

We measure the similarity between $N(s')$ and $N(s)$, by accumulating the local sample neighborhood similarity of their predecessor samples $\upsilon(s')$ and $\upsilon(s)$:

$$\Upsilon(s', s) = \sum_{s_\upsilon \in \upsilon(s)} \Delta(s'_\upsilon, s_\upsilon), \tag{15}$$

where s_υ only goes backward and diffuses from s towards the nearby predecessor sample $\in \upsilon(s)$, $s'_\upsilon \in \upsilon(s')$ is the matching sample of s_υ with the same relative diffusion depth. To prevent $\upsilon(s')$ from running out of samples before $\upsilon(s)$, we adaptively set the diffusion length to be the smaller between $||\upsilon(s)||$ and $||\upsilon(s')||$.

Sample matching

Every time the user places a new brush stroke in the current frame F_κ, our system analyzes the global similarity between the strokes in $F_{\kappa-1}$ and F_κ, and updates their matching. The matching quality will be improved with more brushes. Here we discuss how to determine sample pairs $\{(s', s)\}$ across frames.

Normalization

For each s placed into current frame F_κ, we compute its global similarity $\Upsilon(s', s)$ with each s' in previous frame $F_{\kappa-1}$ of the same brush type as s. Since strokes can have different numbers of samples, we divide each $\Upsilon(s', s)$ by $\sum_{s'} \Upsilon(s', s)$ to fall within $(0, 1)$ without being affected by the diffusion depth.

Candidate matching samples

Inspired by [39, 53], an online method is adopted to determine the threshold when choosing the candidate matches $\{s'\}$ for each s. For each s in F_κ, after we compute and normalize its global similarity to $F_{\kappa-1}$, we find the largest global similarity Υ_{max} first, and then collect samples with global similarity over the threshold (set as 35% of Υ_{max}) to its candidate matching samples.

Update matching sample

We decide the matching sample s_i^κ for each $s_i^{\kappa-1}$ via:

$$s_i^\kappa = \arg\max_s (\Upsilon(s_i^{\kappa-1}, s)), \tag{16}$$

where s runs through all candidate matching samples of $s_i^{\kappa-1}$.

Optimization process

We optimize the prediction framework in Equation (6) to compute the predictions. The unknowns of Equation (6) include affine transformations of the deformation graph, global rigid transformation, sample correspondences in Equation (10), and confidence weights φ for each sample.

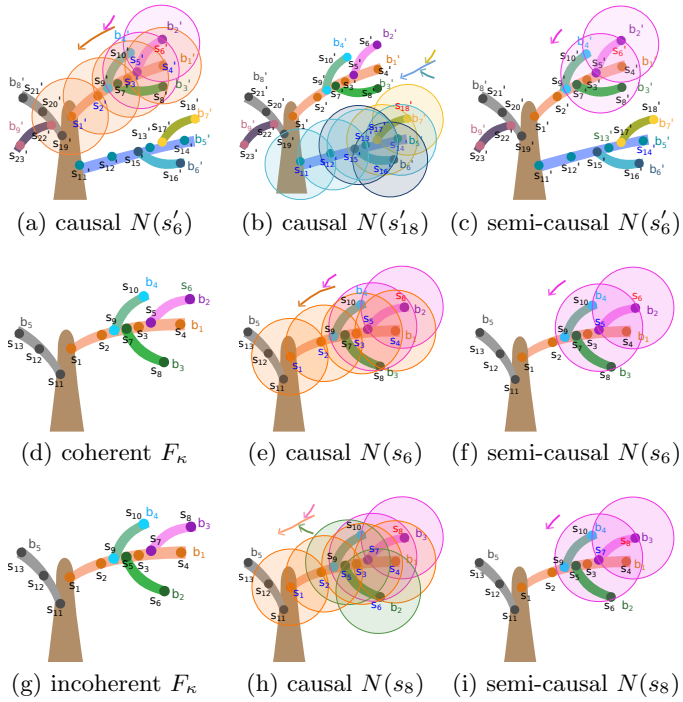
To enhance the prediction quality, we perform a two-pass optimization. In the first-pass, we deform F_κ to $F_{\kappa-1}$ and update the correspondence via ICP as in [30, 47]. With these initial sample correspondences, we deform $F_{\kappa-1}$ to F_κ in the second-pass. For higher accuracy, we update the matched sample of each $s_i^{\kappa-1}$ with our global similarity measurement via Equation (16). $F_{\kappa-1}$ is usually more complete than F_κ which is currently being edited. Thus, in the backward pass, each s_i^κ is likely to find a match in $F_{\kappa-1}$, and thus ICP is a good choice. However, in the forward pass, not all samples in $F_{\kappa-1}$ will have unique matching samples in F_κ. We thus apply Equation (16) to find the best match, with potential duplicating target samples in F_κ for multiple $F_{\kappa-1}$ source samples. The brushing semantic information encoded in the global similarity also provides more accurate and meaningful correspondences, to push the optimization and predictions towards higher quality with users' intentions. We then compute the deformed sample positions via the local deformation model and interpolate samples to form the predicted strokes. More details for the weighting initializations (α, β, γ, η, and ξ) and optimization iterations are summarized in the implementation section.

Difference from [53]

The 2D drawing stroke in [53] adopts a temporally causal global neighborhood. Thus as also pointed out in [53], their method is heavily influenced by the brushing order.

Compared to [53], our global semi-causal sample similarity measurement relaxes the strict temporal causality assumptions, to adapt to more general sculpting animation practice. It is designed to emphasize more on sculpting stroke semantics and aim at capturing larger-scale similarity. The relaxed local neighborhood $n(s)$ takes over more responsibility to capture samples' relationships across different strokes. Though the diffusion path becomes shorter compared to [53], the accumulated enlarged local sample neighborhoods $n(\upsilon(s))$ contribute to capture global structures and relax the original strict coherent temporal strokes order assumption. In contrast, the diffusion paths via [53] are much deeper, which will make it less robust if a certain intermediate strokes break its temporal causality assumption, as illustrated in Figure 14. Furthermore, the broken diffusing routes will propagate to future brush strokes due to its incremental recursive structure. Thus our global semi-causal neighborhood leads to a more robust matching for sculpting strokes samples across frames. One concrete example is shown in Figure 15. Our method can also match input strokes in different numbers and lengths which achieve similar output, as illustrated in Figure 16.

Our global sample similarity depends on the temporal coherence of sample orders within matching strokes. For example, if users place a stroke in the previous frame from left to right, and then change the direction from right to left in the current frame, the global context can be different due to reverse diffusing direction. Fortunately, as statistically analyzed in [44] for sculpting workflows, brushing parameters and preferences are rarely changed and users would prefer repeated brushing. Furthermore, for freeform strokes, the samples ordering within each stroke will naturally be consistent across frames, as they need to start from the mesh shape to perform valid de-



(a) causal $N(s'_6)$ (b) causal $N(s'_{18})$ (c) semi-causal $N(s'_6)$

(d) coherent F_κ (e) causal $N(s_6)$ (f) semi-causal $N(s_6)$

(g) incoherent F_κ (h) causal $N(s_8)$ (i) semi-causal $N(s_8)$

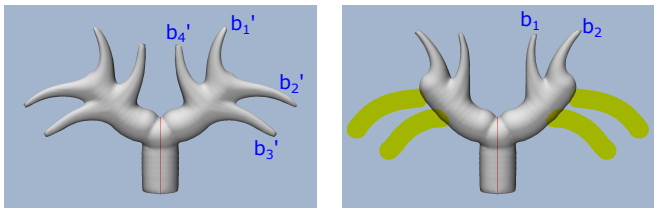
Figure 14: *Causal neighborhood in [53] versus our semi-causal neighborhood.* After finishing $F_{\kappa-1}$, the user continues to author F_κ. (a) to (c) visualize $F_{\kappa-1}$ while (d) through (i) visualize F_κ. In the middle row (d) to (f), F_κ and $F_{\kappa-1}$ have the same strokes order, and ideally, s_6 should match s'_6. In the bottom row (g) through (i) the strokes are placed in a different order, and ideally, s_8 should match s'_6. The method in [53] accumulates the neighborhood $N(s)$ recursively in temporally causal order, as shown in (a), (b), (e), (h). The accumulated recursive path (arrows in different colors correspond to diffusing branches along with different brush strokes) is indicated right above the diffusing neighborhoods (shown in circles in the same colors as the corresponding branch strokes). This works when $F_{\kappa-1}$ and F_κ have similar stroke orders (e.g., (a) and (e)) but not otherwise (e.g., (a) and (h)). And s_8 in (h) will match with s'_{18} in (b) incorrectly. Our semi-causal neighborhood, which accumulates (temporally-non-causal) local neighborhoods from (temporally-causal) samples from the same stroke, can work for both cases, both (f) and (i) can match (c) and decide the correct s'_6 for s_6 and s_8.

formations. Thus our method works well for general brushing preference.

Implementation

Software environment and hardware platform

Our prototype is developed via C++ and Qt 5.12 under macOS 10.14 Mojave with a Wacom Intuos Pen Tablet to support pressure-sensitive brushing input.



(a) frame $F_{\kappa-1}$ (b) frame F_κ

Figure 15: *Semi-causal global similarity example.* The user placed the brushing strokes in $F_{\kappa-1}$ in the order shown in (a). Though the user did not follow the same brushing order in F_κ (b), our semi-causal global neighborhood can still lead to meaningful prediction via a more robust matching to alleviate the need of strict brushing orders.

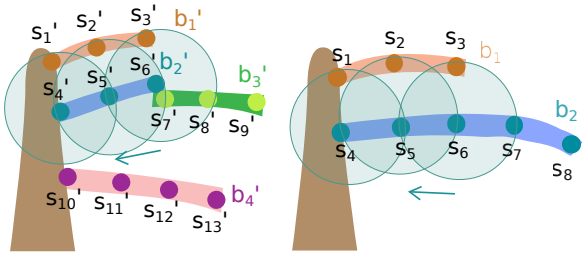


Figure 16: *Matching strokes in different numbers and lengths.* In the first frame (left), there are four strokes $b'_{1,2,3,4}$. In the current frame (right), the user inputs two strokes $b_{1,2}$. Note that the two short strokes $b'_{2,3}$ and the long stroke b_2 achieve similar deformation effects. Our method adaptively sets the diffusion length to be the smaller predecessor set size between $v(s)$ and $v(s')$. So the accumulated global neighborhood would suggest s_6 matches with s'_6; similarly, s_7 matches with s'_8.

Mesh

Interaction efficiency for sculpting is a significant concern [9], even more so for animated sculpting. We adopt a triangle-based mesh representation with the adaptive subdivision option as in Dyntopo [7]. To ensure real-time mesh updates with different sculpting brushes, we employ the octree structure to support instant local mesh deformation.

Manipulation hint

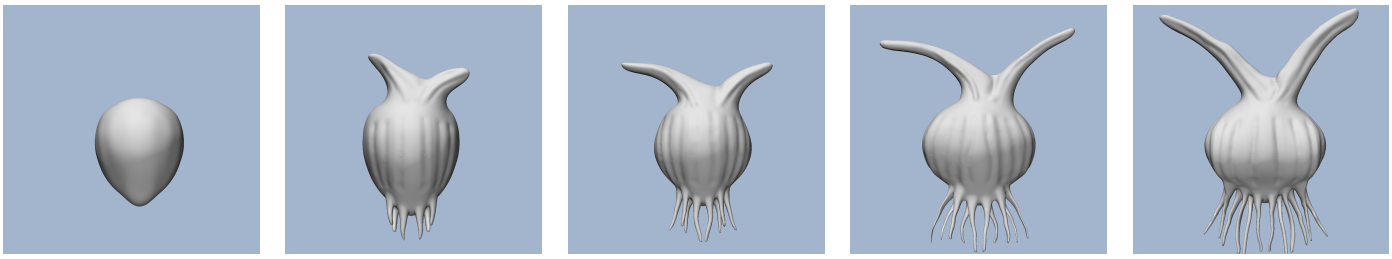
The predicted initial mesh shapes, when starting a new frame, are transformed via the extrapolated manipulation hints, and rendered in light yellow.

Brush hint

The predicted brush strokes are rendered in transparent yellow. Users can partially select hints via the selection brush, rendered in transparent blue.

Workflow clone

The source and target gesture brushes are rendered in red and green respectively. Since gesture brushes are



(a) plain base shape (b) frame 1 (c) frame 2 (d) frame 3 (e) frame 4

Figure 17: *Target sculpting animation task (onion growing).*

usually specified with different lengths and orientations, we normalize the gesture stroke parameterization w.r.t. the gesture arc length.

Workflow interpolation

To produce the workflow-driven interpolated shapes (e.g. Figure 8b), we utilize the optimized transformations via our prediction framework. The optimized transformation suggests how to fully deform $F_{\kappa-1}$ towards F_κ; for the inbetween frame, we deform $F_{\kappa-1}$ with halved transformations, rendered in light purple.

Neighborhood

Inspired by [31], for sample neighborhood similarity in Equation (14), to prevent $n(s')$ from running out of samples before $n(s)$, we set the size of $n(s')$ to be 6, and $n(s)$ to be 4. The ℓ parameter in Equation (4) is set to the full sample neighborhood size 6.

Optimization

We optimize this nonlinear least-squares problem of our prediction framework using the Levenberg Marquardt algorithm. For this sparse system matrix, we solve the normal equations in each iteration via a direct solver that adopts sparse Cholesky factorization to enhance performance. We alternate two steps until convergence: update correspondence Equation (10) with transformations fixed, then optimize transformations with correspondences fixed. The A_i in Equation (8) is initialized as an identity matrix. The unknowns are jointly determined by the optimization solver. We follow a heuristic to automatically adapt the optimization weights in Equation (6), to initially favor a rigid alignment, and then lower the stiffness to increase deformation as the optimization progresses. Specifically, we initialize α, β, γ, η, and ξ to be 1000, 100, 10, 100, and 10 respectively. α, β, and η are halved whenever $|\mathscr{E}_i - \mathscr{E}_{i-1}| < 10^{-3}(1 + \mathscr{E}_i)$, where $\mathscr{E}_i$ is the total cost of i-th iteration, until $\alpha < 1$, $\beta < 1$, and $\eta < 1$. Other weights remain fixed.

Other parameters

For Equation (13), we set ω_p, ω_a, ω_m to be 1. To observe the brush stroke topology for neighborhood matching, we set ω_t to be 10 when samples belong to the same brush stroke and set to 0 otherwise.

USER STUDY

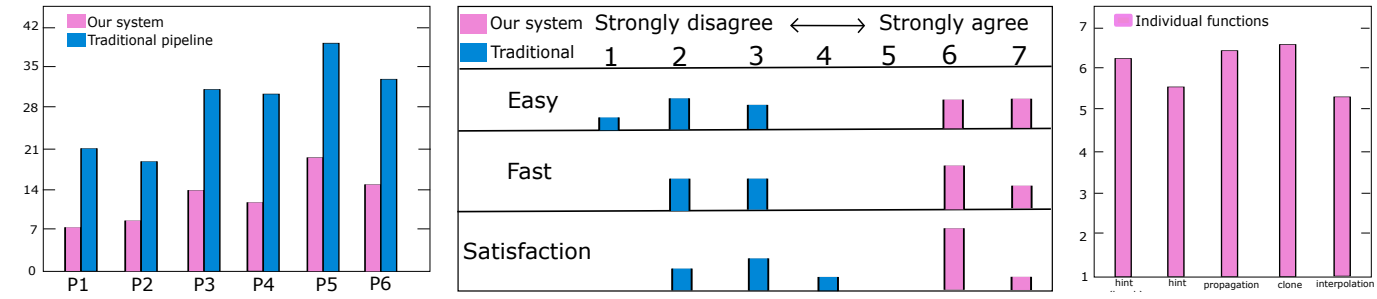
We conducted a pilot user study to gain insights on how our system compares with the traditional fully manual keyframe sculpting pipeline. We invited two professionals ($P1$,2) and four novices ($P3$,4,5,6) with different levels of sculpting animation experiences as participants. All tasks were conducted on a 13-inch laptop with a Wacom tablet. The study consisted of four sessions: tutorial, target session, open creation, and final interview. The first author provided guidance to all participants, observed their behaviors, took notes, recoreded their feedback, and collected animation results.

Tutorial (30 minutes)

The goal is to help participants familiarize themselves with sculpting animation. The participants were given an overview of our system, as well as the traditional pipeline composed of modeling followed with animation, including the popular tools for each stage (ZBrush and Blender for the modeling, Maya and Mush3D for the animation). For the extra tutorial, since sculpting animations often contain topological changes, we taught all users how to handle topological changes on the interface of traditional pipeline, while our system does not require any extra tutorial. The first author provided the same amount of guidance to all participants.

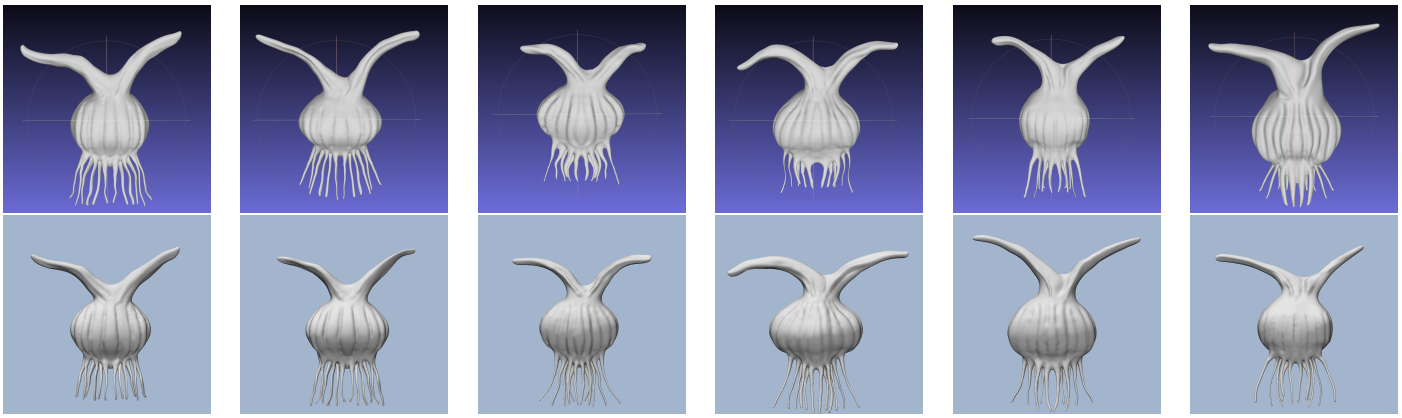
Target session (1 hour)

The goal is to measure and compare the objective performance and subjective experience of our system compared to the existing pipeline. The participants were asked to reproduce a target sculpting animation sequence (Figures 17b to 17e) in two separate conditions (traditional pipeline and our system). The target animation consists of an onion life cycle [32], a typical example of sculpting animations with organic shape, motion, and topological changes, with complexity suitable for the user study. In the traditional baseline condition, participants were allowed to use whichever tools they felt comfortable with. Specifically, the professional users ($P1, 2$) completed the baseline condition via the traditional pipeline as they normally do. Participants were provided with the necessary input base mesh file (Figure 17a) to be imported as the start point for both conditions. The condition orders were fully counterbalanced among participants, and there was no time limit for each condition.



(a) target session complete time (minutes) (b) traditional pipeline vs our system (c) features of our system

Figure 18: *User study results.*



(a) P1 (21/7 min) (b) P2 (18/8 min) (c) P3 (30/14 min) (d) P4 (29/12 min) (e) P5 (38/20 min) (f) P6 (32/15 min)

Figure 19: *Resulting artifacts (ending frame) in the target session using traditional pipeline (top row) and our system (bottom row) by the participants.* The number indicates the authoring time required to complete the whole target session (traditional/our).

Open session (20 minutes)
The goal is to observe user behaviors and identify merits and issues of our systems. Participants were encouraged to explore various functions and perform open-ended sculpting animations using our system.

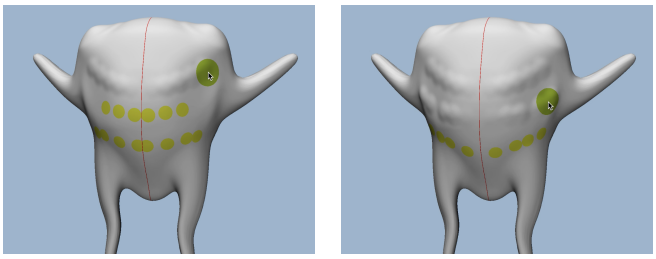
Interview (10 minutes)
The goal is to collect feedback from each participant on different aspects of our system, including different features, overall satisfaction, and open-ended feedback.

RESULTS

Quantitative measurement
All participants ($P1$ to $P6$) were able to complete the target session under both conditions, as shown in Figure 18a. The outcomes are shown in Figure 19. On average, the target session took 28 minutes for the traditional pipeline, and 12 minutes for our unified system, indicating the core design goal *efficiency* is realized. The ratio of completed time by traditional pipeline versus our system was 2.33. We observed that novice users tended to struggle with topological changes across frames (commonly produced using sculpting animations) using traditional condition, and required extra training during the tutorial session. In contrast, the participants did not need an extra tutorial for our system, indicating that our system is easier to learn.

Subjective feedback
Figure 18c summarizes the subjective feedbacks about the individual features of our system. Overall, the participants responded positively to the features of our system. The users commented that the suggestions provided by our system were fast and efficient ($P1$,2,3,4,5,6), interesting ($P5$), easy to learn and use ($P5$,6).

In the post-study questionnaire, when asked "which one do you prefer for sculpting animation for the target session", all participants preferred our system compared to traditional workflows for the given task. Participants were also asked to "score the two conditions to perform the target sculpting animation sessions" based on a 7-Likert scale (the higher the better). As demonstrated in Figure 18b, participants found our system easier and faster, and they felt more satisfied with the sculpting animation created by our system compared to the tradi-



(a) initial suggestions (b) updated suggestions

Figure 20: *User iterations with system suggestions.* Our system might not produce suggestions that the users would like as in (a), for which the users can ignore and continue brushing. Our system can then incorporate user edits to update the suggestions in real-time (b).

tional pipeline, e.g. "The yellow suggestions are fast, the brushes are more suitable for the target sculpting details, no need for platform switching, such topology changes are not easy to handle within maya-like toolset" ($P1$). Participants ($P2$,3,4,6) commented that they preferred our unified system instead of switching around different tools with different interaction designs via traditional pipeline, such as "I can reduce manual input with autos in a single interface" ($P2$) and "I prefere one independent system, easy to learn with basic brushing knowledge and experience" ($P6$), which indicates our design goal *coherence* is realized. Participants also liked the inherent ZBrush/Blender-like high-quality sculpting tools ($P1$,2,3), suggesting our goal *flexibility* is fulfilled; and suggestive visualizations to reduce workload ($P1$,2,3,4,6), indicating our design goal *efficiency* is achieved. Participants also favored our keyframing manipulation with 2D-cell animation-like onion skin and commented it to be more straightforward than traditional condition ($P1$,2,5), e.g. "The previous frames' shadows are helpful as guidance, they are straightforward without extra setups" ($P2$). We observed that novices with limited animation experience required more time to get familiar with the traditional interfaces and they commented it was much easier to get familiar with our unified interface for animation authoring, such as "the navigation is similar to Blender so I do not need to change the sculpting navigation practice" ($P3$) and "the interface and icons are not complex compared to traditional one" ($P5$). Participants also suggested that we can try to combine with Maya when designing more advanced features to support sculpting and non-sculpting object animations, e.g. "adding similar keyframe curve of maya/mush3d to control density" ($P1$); and more widgets for feature options, e.g. "add a slider widget for users to adjust the interpolated frames" ($P2$).

Sample results

Figure 21 presents some sample results created by our study participants during the open session. The results show that our standalone system can support creating sculpting animation sequences from scratch with continuous topological changes (such as a growing tree in Figure 21c or transforming virus in Figure 21b), spatial movements (such as Figure 21d), larger-scale deformations (such as the octopus animations in Figure 21a which may require external manipulators for rigging in traditional condition), and surface details changes (such as Figure 21e).

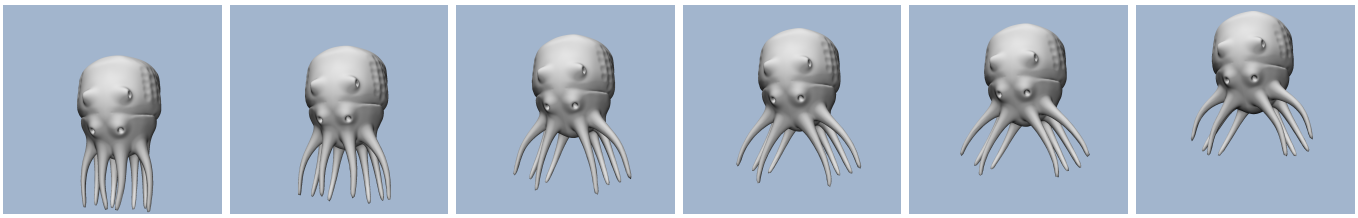
CONCLUSIONS, LIMITATIONS, AND FUTURE WORK

Brush strokes have been demonstrated as a flexible and intuitive means to author 2D sketches, 2D animations, and 3D sculptures. A main goal of this work is to continue and expand this line of inquiry for 3D animation, which naturally requires high artistic/technical skills and manual labors, even more so than 2D sketching, 2D animation, and 3D sculpting. Our autocomplete system provides opportunities to reduce these expertise and efforts barriers for users with different backgrounds while respecting their freedom of exploration and individual expression of styles.

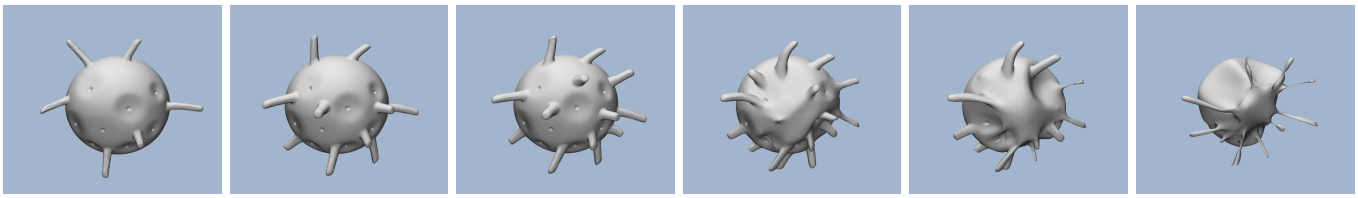
Like other autocomplete systems, our method might not always suggest what the users would like to do, due to the inherent complexity and subjective nature of artistic creation, such as Figure 20a during the animation authoring for Figure 1. For such cases, users can ignore the suggestions and continue brushing, and let our system incorporate the latest inputs to update the suggestions with higher quality in real-time, as illustrated in Figure 20b. This can be an effective way to balance automation and customization for assistive freestyle content creation. A pilot study indicated that our system can help users create compelling animated objects through an intuitive sculpting interface without requiring other data or tools. We plan to engage with more users to elicit feedback and create more varied outputs to further explore and expand our design and method, such as displaying multiple possible solutions [54, 20] and incorporating machine learning [6].

Our prototype is deployed in a desktop environment, but the proposed features and methods are general and independent of the platforms. VR sculpting can be a particularly interesting platform to pursue given the immersive brushing experience. A potential work is to extend the scope to handle spatial strokes like in Medium [36] for more freeform topological changes, such as adding components and punching holes. Supporting tangible feedback [2, 1] is also of interest for VR sculpting.

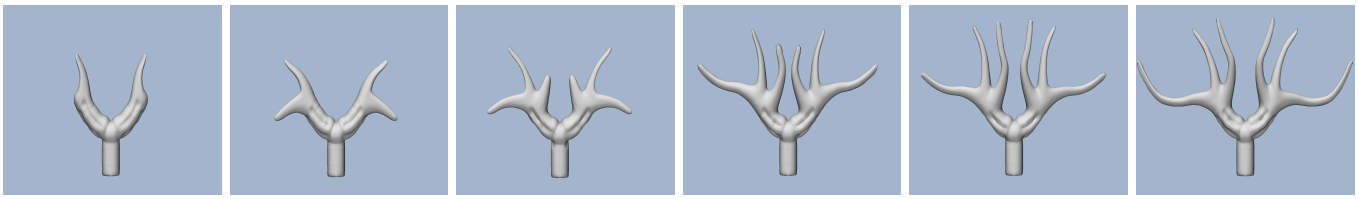
We focus on shape and motion for this project, and plan to consider textures and colors [16, 48] for a more feature-complete authoring environment. One potential direction is to apply our autocomplete algorithms beyond a single user session for applications such as tutorials (leveraging expert workflows to help novices) [11] and collaborations (combining workflows from multiple users and multiple sessions) [37, 43, 9]. Another direction is to predict UI commands and automatically free up screen space by



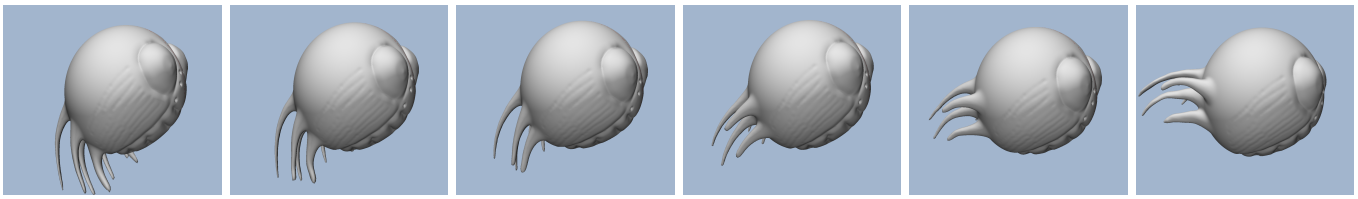
(a) swimming octopus: 62/83(15+68) manual/autocomplete strokes, 11 min



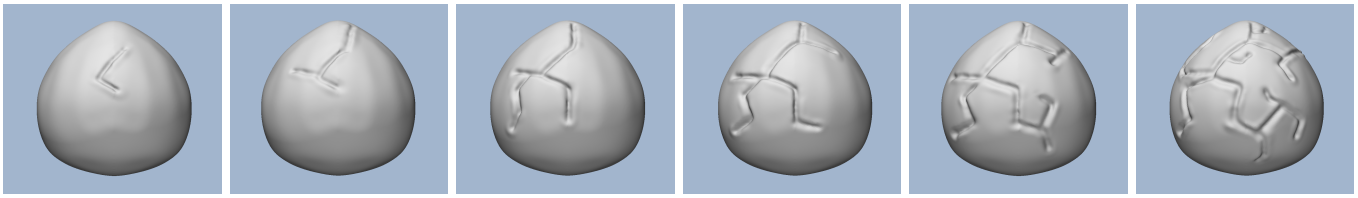
(b) mutating virus: 86/112(18+94) manual/autocomplete strokes, 12 min



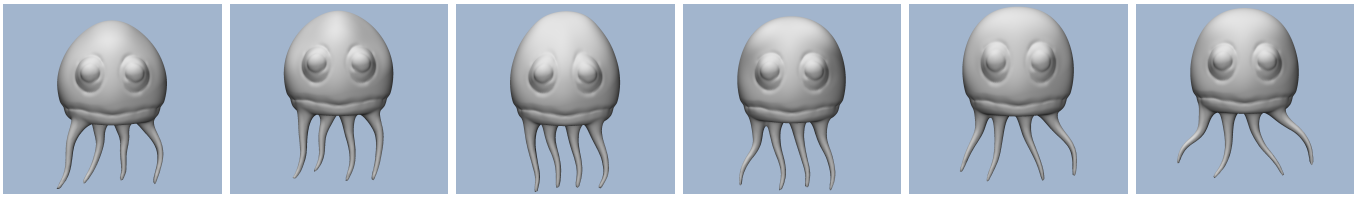
(c) growing tree: 45/89(0+89) manual/autocomplete strokes, 7 min



(d) flying insect: 92/80(20+60) manual/autocomplete strokes, 12 min



(e) fracturing egg: 31/55(0+55) manual/autocomplete strokes, 8 min



(f) wandering jellyfish: 35/42(12+30) manual/autocomplete strokes, 7 min

Figure 21: *Sample outputs.* Results shown here are example sculpting animations produced by our system. Each sequence is denoted with the following statistics: number of manual brushes, number of autocomplete brushes (within-frame hints via [39] + our across-frame hints), and sculpting animation time in minutes. Please refer to the accompanying video for animations.

hiding inactive UI elements, towards a dynamic and less cluttered UI [46].

Acknowledgements

We would like to thank our user study participants for their time and feedback, and the anonymous reviewers for their valuable suggestions. Mengqi Peng has been funded by a HKU postgraduate scholarship, and began this project during her internship with Adobe Research.

REFERENCES

[1] Rahul Arora, Rubaiat Habib Kazi, Tovi Grossman, George Fitzmaurice, and Karan Singh. 2018. SymbiosisSketch: Combining 2D & 3D Sketching for Designing Detailed 3D Objects in Situ. In *CHI '18*. Article 185, 15 pages. DOI: http://dx.doi.org/10.1145/3173574.3173759

[2] Rahul Arora, Rubaiat Habib Kazi, Fraser Anderson, Tovi Grossman, Karan Singh, and George Fitzmaurice. 2017. Experimental Evaluation of Sketching on Surfaces in VR. In *CHI '17*. 5643–5654. DOI:http://dx.doi.org/10.1145/3025453.3025474

[3] Rahul Arora, Rubaiat Habib Kazi, Danny Kaufman, Wilmot Li, and Karan Singh. 2019. MagicalHands: Mid-Air Hand Gestures for Animating in VR. In *UIST '19*.

[4] AutoDesk. 2019a. 3dsMax. (2019). https://www.autodesk.com.hk/products/3ds-max/overview

[5] AutoDesk. 2019b. Maya. (2019). https://www.autodesk.com/products/maya/overview

[6] AutoDesk. 2020. TinkerCAD. (2020). https://www.tinkercad.com

[7] Blender Foundation. 2019. Blender. (2019). https://www.blender.org/

[8] Nestor Burtnyk and Marceli Wein. 1976. Interactive skeleton techniques for enhancing motion dynamics in key frame animation. *Commun. ACM* 19, 10 (1976), 564–569.

[9] Claudio Calabrese, Gabriele Salvati, Marco Tarini, and Fabio Pellacini. 2016. CSculpt: A System for Collaborative Sculpting. *ACM Trans. Graph.* 35, 4, Article Article 91 (July 2016), 8 pages. DOI: http://dx.doi.org/10.1145/2897824.2925956

[10] Hsiang-Ting Chen, Tovi Grossman, Li-Yi Wei, Ryan M. Schmidt, Björn Hartmann, George Fitzmaurice, and Maneesh Agrawala. 2014. History Assisted View Authoring for 3D Models. In *CHI '14*. 2027–2036. DOI: http://dx.doi.org/10.1145/2556288.2557009

[11] Hsiang-Ting Chen, Li-Yi Wei, Björn Hartmann, and Maneesh Agrawala. 2016. Data-Driven Adaptive History for Image Editing. In *I3D '16*. 103–111. DOI:http://dx.doi.org/10.1145/2856400.2856417

[12] Byungkuk Choi, JP Lewis, Yeongho Seol, Seokpyo Hong, Haegwang Eom, Sunjin Jung, Junyong Noh, and others. 2016. SketchiMo: sketch-based motion editing for articulated characters. *ACM Transactions on Graphics (TOG)* 35, 4 (2016), 146.

[13] ChronoSculpt. 2019. ChronoSculpt. (2019). https://www.lightwave3d.com/chronosculpt/.

[14] James Davis, Maneesh Agrawala, Erika Chuang, Zoran Popović, and David Salesin. 2003. A Sketching Interface for Articulated Figure Animation. In *SCA '03*. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 320–328. http://dl.acm.org/citation.cfm?id=846276.846322

[15] Jonathan D Denning, Valentina Tibaldo, and Fabio Pellacini. 2015. 3dflow: Continuous summarization of mesh editing workflows. *ACM Transactions on Graphics (TOG)* 34, 4 (2015), 140.

[16] Marek Dvorožňák, Wilmot Li, Vladimir G. Kim, and Daniel Sýkora. 2018. ToonSynth: Example-Based Synthesis of Hand-Colored Cartoon Animations. *ACM Transactions on Graphics* 37, 4, Article 167 (2018).

[17] Facebook Quill. 2019. Quill. (2019). https://quill.fb.com/

[18] Lubin Fan, Ruimin Wang, Linlin Xu, Jiansong Deng, and Ligang Liu. 2013. Modeling by drawing with shadow guidance. *Computer Graphics Forum* 32, 7 (2013), 157–166.

[19] Martin Guay, Rémi Ronfard, Michael Gleicher, and Marie-Paule Cani. 2015. Space-time Sketching of Character Animation. *ACM Trans. Graph.* 34, 4, Article 118 (July 2015), 10 pages. DOI: http://dx.doi.org/10.1145/2766893

[20] Paul Guerrero, Gilbert Bernstein, Wilmot Li, and Niloy J. Mitra. 2016. PATEX: Exploring Pattern Variations. *ACM Trans. Graph.* 35, 4, Article 48 (July 2016), 13 pages. DOI: http://dx.doi.org/10.1145/2897824.2925950

[21] Fabian Hahn, Sebastian Martin, Bernhard Thomaszewski, Robert Sumner, Stelian Coros, and Markus Gross. 2012. Rig-space Physics. *ACM Trans. Graph.* 31, 4, Article 72 (July 2012), 8 pages. DOI:http://dx.doi.org/10.1145/2185520.2185568

[22] Eric Horvitz. 1999. Principles of mixed-initiative user interfaces. In *CHI '99*. 159–166.

[23] Jazza. 2014. Tutorial: Pose to Pose Animation - Keyframes and Motion Breakdown. (2014). https://youtu.be/VcBwDHx_oDk

[24] Rubaiat Habib Kazi, Fanny Chevalier, Tovi Grossman, and George Fitzmaurice. 2014a. Kitty: Sketching Dynamic and Interactive Illustrations. In *UIST '14*. 395–405. DOI: http://dx.doi.org/10.1145/2642918.2647375

[25] Rubaiat Habib Kazi, Fanny Chevalier, Tovi Grossman, Shengdong Zhao, and George Fitzmaurice. 2014b. Draco: Bringing Life to Illustrations with Kinetic Textures. In *CHI '14*. 351–360. DOI: http://dx.doi.org/10.1145/2556288.2556987

[26] Rubaiat Habib Kazi, Tovi Grossman, Nobuyuki Umetani, and George Fitzmaurice. 2016. Motion Amplifiers: Sketching Dynamic Illustrations Using the Principles of 2D Animation. In *CHI '16*. 4599–4609. DOI: http://dx.doi.org/10.1145/2858036.2858386

[27] Junggon Kim and Nancy S Pollard. 2011. Direct control of simulated nonhuman characters. *IEEE Computer Graphics and Applications* 31, 4 (2011), 56–65.

[28] Yong Jae Lee, C. Lawrence Zitnick, and Michael F. Cohen. 2011. ShadowDraw: Real-time User Guidance for Freehand Drawing. *ACM Trans. Graph.* 30, 4, Article 27 (July 2011), 10 pages. DOI: http://dx.doi.org/10.1145/2010324.1964922

[29] John P Lewis, Ken Anjyo, Taehyun Rhee, Mengjie Zhang, Frederic H Pighin, and Zhigang Deng. 2014. Practice and Theory of Blendshape Facial Models. *Eurographics (State of the Art Reports)* 1, 8 (2014), 2.

[30] Hao Li, Robert W. Sumner, and Mark Pauly. 2008. Global Correspondence Optimization for Non-rigid Registration of Depth Scans. In *Proceedings of the Symposium on Geometry Processing (SGP '08)*. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 1421–1430. http://dl.acm.org/citation.cfm?id=1731309.1731326

[31] Chongyang Ma, Li-Yi Wei, and Xin Tong. 2011. Discrete element textures. In *ACM Transactions on Graphics (TOG)*, Vol. 30. ACM, 62.

[32] Kazakova Maryia. 2020. Onion life cycle. (2020). https://www.shutterstock.com/image-vector/onion-life-cycle-growth-stages-seeding-738951745.

[33] Yusuke Matsui, Takaaki Shiratori, and Kiyoharu Aizawa. 2017. DrawFromDrawings: 2D Drawing Assistance via Stroke Interpolation with a Sketch Database. *IEEE transactions on visualization and computer graphics* 23, 7 (2017), 1852–1862.

[34] Stuart Mealing. 1994. *The Art and Science of Computer Animation* (1st ed.). Intellect Books, Exeter, UK, UK.

[35] Mush3D. 2019. Mush3D. (2019). https://mush3d.com/.

[36] Oculus. 2019. Medium. (2019). https://www.oculus.com/medium/

[37] Onshape Inc. 2015. OnShape. (2015). https://www.onshape.com/.

[38] Priyanka Patel, Heena Gupta, and Parag Chaudhuri. 2016. TraceMove: A Data-assisted Interface for Sketching 2D Character Animation. In *Proceedings of the 11th Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications: Volume 1: GRAPP (GRAPP 2016)*. SCITEPRESS - Science and Technology Publications, Lda, Portugal, 191–199. DOI:http://dx.doi.org/10.5220/0005672501890197

[39] Mengqi Peng, Jun Xing, and Li-Yi Wei. 2018. Autocomplete 3D Sculpting. *ACM Trans. Graph.* 37, 4, Article 132 (July 2018), 15 pages. DOI: http://dx.doi.org/10.1145/3197517.3201297

[40] Patrick Pérez, Michel Gangnet, and Andrew Blake. 2003. Poisson Image Editing. *ACM Trans. Graph.* 22, 3 (July 2003), 313–318. DOI: http://dx.doi.org/10.1145/882262.882269

[41] Pixologic. 2019. ZBrush. (2019). http://pixologic.com/zbrush/

[42] William T. Reeves. 1981. Inbetweening for Computer Animation Utilizing Moving Point Constraints. *SIGGRAPH Comput. Graph.* 15, 3 (Aug. 1981), 263–269. DOI: http://dx.doi.org/10.1145/965161.806814

[43] Gabriele Salvati, Christian Santoni, Valentina Tibaldo, and Fabio Pellacini. 2015. MeshHisto: Collaborative Modeling by Sharing and Retargeting Editing Histories. *ACM Trans. Graph.* 34, 6, Article Article 205 (Oct. 2015), 10 pages. DOI: http://dx.doi.org/10.1145/2816795.2818110

[44] Christian Santoni, Claudio Calabrese, Francesco Di Renzo, and Fabio Pellacini. 2016. Sculptstat: Statistical analysis of digital sculpting workflows. *arXiv preprint arXiv:1601.07765* (2016).

[45] Thomas W Sederberg and Scott R Parry. 1986. Free-form deformation of solid geometric models. *ACM SIGGRAPH computer graphics* 20, 4 (1986), 151–160.

[46] Siemens. 2019. Adaptive UI in NX. (2019). https://blogs.sw.siemens.com/nx-design/New-Adaptive-UI/

[47] Robert W. Sumner, Johannes Schmid, and Mark Pauly. 2007. Embedded Deformation for Shape Manipulation. In *ACM SIGGRAPH 2007 Papers (SIGGRAPH '07)*. ACM, Article 80. DOI: http://dx.doi.org/10.1145/1275808.1276478

[48] Ryo Suzuki, Koji Yatani, Mark D. Gross, and Tom Yeh. 2018. Tabby: Explorable Design for 3D Printing Textures. In *Proceedings of the 26th Pacific Conference on Computer Graphics and Applications: Short Papers (PG '18)*. Eurographics Association, Goslar, DEU, 29–32. DOI: http://dx.doi.org/10.2312/pg.20181273

[49] Balasaravanan Thoravi Kumaravel, Cuong Nguyen, Stephen DiVerdi, and Björn Hartmann. 2019. TutoriVR: A Video-Based Tutorial System for Design Applications in Virtual Reality. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. ACM, 284.

[50] Brian Whited, Gioacchino Noris, Maryann Simmons, Robert W. Sumner, Markus Gross, and Jarek Rossignac. 2010. BetweenIT: An Interactive Tool for Tight Inbetweening. *Computer Graphics Forum* 29, 2 (2010), 605–614. DOI: http://dx.doi.org/10.1111/j.1467-8659.2009.01630.x

[51] Jun Xing, Hsiang-Ting Chen, and Li-Yi Wei. 2014. Autocomplete Painting Repetitions. *ACM Trans. Graph.* 33, 6, Article 172 (Nov. 2014), 11 pages. DOI:http://dx.doi.org/10.1145/2661229.2661247

[52] Jun Xing, Koki Nagano, Weikai Chen, Haotian Xu, Li-yi Wei, Yajie Zhao, Jingwan Lu, Byungmoon Kim, and Hao Li. 2019. HairBrush for Immersive Data-Driven Hair Modeling. In *UIST '19*. 263–279. DOI:http://dx.doi.org/10.1145/3332165.3347876

[53] Jun Xing, Li-Yi Wei, Takaaki Shiratori, and Koji Yatani. 2015. Autocomplete Hand-drawn Animations. *ACM Trans. Graph.* 34, 6, Article 169 (Oct. 2015), 11 pages. DOI: http://dx.doi.org/10.1145/2816795.2818079

[54] Loutfouz Zaman, Wolfgang Stuerzlinger, Christian Neugebauer, Rob Woodbury, Maher Elkhaldi, Naghmi Shireen, and Michael Terry. 2015. GEM-NI: A System for Creating and Managing Alternatives In Generative Design. In *CHI '15*. 1201–1210. DOI: http://dx.doi.org/10.1145/2702123.2702398

[55] Yufeng Zhu, Jovan Popović, Robert Bridson, and Danny M. Kaufman. 2017. Planar Interpolation with Extreme Deformation, Topology Change and Dynamics. *ACM Trans. Graph.* 36, 6, Article 213 (Nov. 2017), 15 pages. DOI: http://dx.doi.org/10.1145/3130800.3130820

[56] Susan Zwerman and Jeffrey A Okun. 2012. *Visual Effects Society Handbook: Workflow and Techniques*. CRC Press.