Designing a Compelling User Interface for Morphing

David Vronay
Rapid Prototyping Research Group
Microsoft Research Asia
3F, Beijing Sigma Center
No.49, Zhichun Road Haidian District
Beijing 100080, P. R. China
davevr@microsoft.com

Shuo Wang
Industrial Design Department
Academy of Arts & Design
Tsinghua University
34 Middle DongSanhuan Road
Beijing, 100020, P. R. China
Wangs01@mails.tsinghua.edu.cn

ABSTRACT

We present a new user interface for the common morphing tool found in animation packages. Previously this interface has been based on the features of the underlying algorithm, with little regard to how artists actually use this feature. By careful design and analysis of a user study, we were able to design a novel user interface that greatly enhances the usability of the morphing tool for animation. Our improvements come in three areas: First, we replicate the artists' own ad-hoc annotation language and interaction techniques in the user interface. Second, we make the user experience more fluid and editable, to support exploration and iteration. Finally, we use the artists' morph expectations to redesign the morph algorithm itself to be more predictable. We conclude by discussing how our user study technique could help other interface design tasks.

Categories & Subject Descriptors: H.5.2. User-centered design , H.5.2. Prototyping, I.3.4. Graphics Utilities

General Terms: Design, Human Factors

Keywords: User Interface Design, Interaction Design, Prototyping Animation, User Studies, User-Centered Design / Human-Centered Design.

INTRODUCTION

Shape morphing is a popular technique in computer animation. In a shape morph (or simply "morph" for short), the artist is able to draw two shapes and have the system smoothly interpolate between them. Morphing is used for many things, such as transitions between scenes (car turns into boat), transformation of objects (man into wolf) , as well as general animation assistance (dancer with arms down turns into dancer with arms up.)

Our team was tasked to design a compelling morphing user interface for a new drawing program. Based on anecdotal feedback from designers, we knew that the current user interface was not satisfactory. A redesign was indicated.

Existing Techniques

In order to better understand our design problem, it is necessary to know something about the current morphing techniques.

There are a variety of algorithms to perform a morph. While they vary in their implementation details, all of them rely on some user input to establish correspondence between sub-parts of the starting and ending shape in order to compute the morph.

The most common of these techniques is to establish pairs of corresponding vertices on the two shapes, essentially saying "point A on shape 1 maps to point B on shape 2". The different morphing algorithms then use these constraints to inform the morphing process. There are two parts to the process. First is the vertex correspondence problem - generating the correspondence pairs for the remaining unspecified points. Second is the vertex interpolation problem - determining the intermediate position for a point as it transforms from start to finish.

A successful morph realizes both the vertex correspondence and vertex interpolation intended by the artist, while a failed morph results in an unsatisfying or unintended transition. Figure 1, from [3], provides an example.

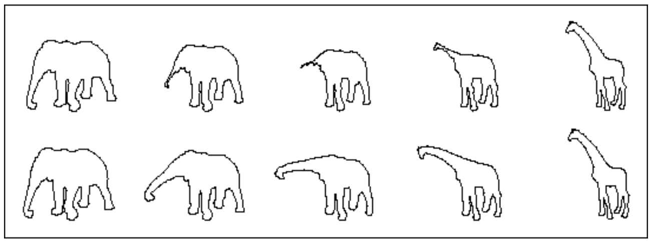
Figure 1: Bad Morph (top) vs. Good Morph (bottom)

Importance of UI in Morphing

Morphing is fundamentally, at its core, an animation shortcut. An animator does not have to use a morph tool – they can always manipulate the shape themselves by hand, specifying the vertex correspondence and interpolation for each and every point along the animation path. Therefore, a morph is only valuable to the user insofar as it is more efficient than the manual alternative.

There are several aspects to efficiency, including number of steps required, difficulty of correctly performing those steps, difficulty of correctly identifying which steps are needed, and predictability of the result. When evaluating the morph UI, we must consider all of these factors.

A further factor to consider is that morphing is an aesthetic process. There is no "best" way to morph. While people may generally agree that a particular morph is bad, there is legitimate difference of opinion on whether one morph is better than another. For instance, consider the morph of an elephant into a giraffe. One artist might want the body to shrink while the neck extends, then have legs elongate, and finally have the trunk recoil into the head. Another artist might want all parts to smoothly transform at the same time. As Terry [4] notes, in any artistic process an ideal tool would allow for experimentation and exploration of results.

Critiquing the current Morph UI

Unfortunately, the morph user interfaces found in popular animation programs do not address these requirements. Rather, they simply expose the parameters of the underlying morph algorithm directly.

In terms of the efficiency points mentioned above, the standard morphing fails on all fronts. By focusing only on vertex correspondence, they do not give the user much if any control over the vertex interpolation. This in turn leads to highly unpredictable results. What's worse, the user has no way to determine if more correspondence pairs or better chosen correspondence pairs will help.

Standard morphing also fails in the real of aesthetic exploration. Morphing is normally implemented as an all-or-nothing affair. Either you like the result or you start over. Little support is provided for small edits or tweaking.

RELATED WORK

Abundant literature on 2D shape morphing can be found on in recent decades. Unfortunately these papers are overwhelming focused on the morph algorithms themselves and seldom involve user interface considerations. A representative but by no means exhaustive list would include [1, 3, 5, 7, 13]. To the extent that different algorithms affect the vertex correspondence and vertex interpolation needs of the user interface, we deemed them relevant to our current exploration.

Sederberg et al. give a deterministic solution (based on energy minimization) to the correspondence problem in [1]. Shapira and Rappoport use a star-skeleton representation of polygons to solve both problems [5]. While these techniques help address some of the predictability problems with competing algorithms, they do nothing to address the need for creative control and exploration.

In this paper we have limited our discussion to 2d shape morphing. However, there are other forms of morphing in the literature, such as 3D morphing and image morphing. Like shape morphing, these techniques also require interaction from the user to achieve acceptable results.

Zhong et al. realized this requirement in their 3D mesh morph, and increased the focus on user control in the morphing process by introducing component control instead of vertex control [8]. However, their approach did not seem to come from the needs of perspective end users, and no follow-up user testing was reported. Therefore, we cannot evaluate the success of their component control concept.

USER STUDY

Finding little in the literature to guide us, our first task was to understand in more detail exactly what artists wanted to do with morphing. This would let us understand not only what was wrong with current tools, but also design tools that provided better support.

We sent out a self-administered user study to several dozen participants. The study consisted of a short self-assessment section on age, gender, and computer experience, followed by 12 pages of figures. On each page, the user was presented with two figures labeled A and B and was asked "using any combination of drawings or words, describe how figure A would transform over time into figure B".

The figures were selected to contain at least one example from each of the following classes of transformations:

- matrix transformations, in which one shape is a rotated, sheared, or scaled version of another
- smoothing operations, in which one shape is a curve drawn through the same points that are connected by straight lines in the other
- addition or subtraction of vertices, in which for example a triangle becomes a square
- conceptual transformations, where the intuitive transformation depends on some conceptual knowledge of what the shape represents (such as an elephant into horse)
- addition or deletion of a topological disc, in which two or more items become one
- complex transformations, in which there is no obvious way to map between the samples (for example, a scene of a plane flying through clouds becomes a house).

We chose these figures for three purposes: First, we wanted to determine if subjects deferred to simple mathematic transformations (such as matrix transforms) when they were possible. Second, we wanted to learn as much as possible about the language (both visual and textual) that the subjects used when describing morphs, in the hope that we could then introduce these same concepts into the user interface. Lastly, we wanted to see if there were any fundamental concepts that applied across all of these examples that we could put into the UI. A sample of the different figures on the questionnaire can be seen in figure 2.

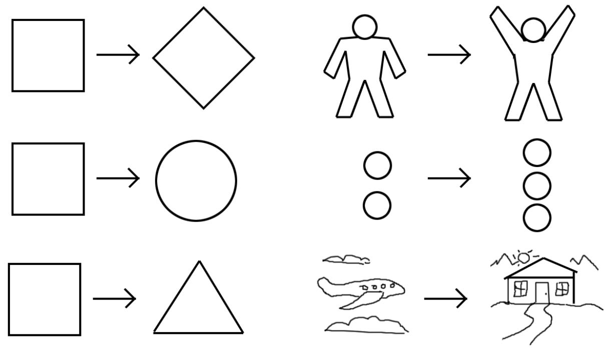

Figure 2: Questionnaire examples

In all, we had 43 responses that were judged acceptable (12 female, 31 male). Of these, 15 had backgrounds in design, while 23 where in computer-related fields. 40% had used morphing software of some kind before. Questionnaire subjects were volunteers and received no gratuity for their participation.

Questionnaire Study Results

Upon analysis of the questionnaires, we determined a number of user principles of morphing.

1) There are many ways to morph between shapes, even very simple ones. Our results included no fewer than three ways of doing even the simplest morph (a + rotated to make an x) and as many as 11 ways of the more complex ones.
2) Most subjects readily identified matrix transformations as the ideal morph when possible. Morphs that could be done with matrix transformations had fewer variants than those that could not.
3) Subjects tended to think of correspondence at a subpart level rather than at the vertex level. For example, users would say things like "the trunk of the elephant there becomes the neck here." Almost invariably, users referred to parts by circling them. Analysis of the circled areas showed that they were not particularly exact – for example, circling more or less than just the leg to indicate a leg.
4) Subjects often wanted temporal control over parts of the morph. For instance, they wanted to specify that one leg morphed, then the next. There was a high correlation between the parts called out for these sequences and the parts called out for correspondence in point 3 above.
5) Subjects often relied on some sort of implied physical properties of the objects when doing their morphs. The physical properties varied from case to case, but included such things as surface tension, gravity, elasticity, conservation of volume, etc. They tended to imagine the objects as if made of clay, water, rubber balloons, and so forth.
6) In addition to physical properties, subjects also often expressed the morph in terms of physical forces acting on the parts. Common forces were pushing, pulling, pinching, growing, shrinking, and flattening. Most subjects indicated these forces and their direction through the use of arrows. (Figure 3)

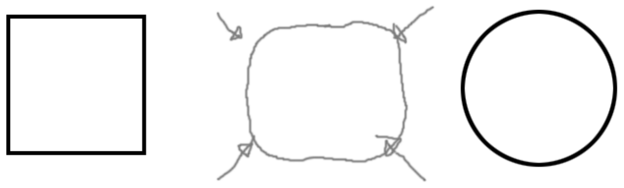

Figure 3: subject draws lines indicating forces

7) Subjects demonstrated the greatest variety of responses in the cases that included the addition and subtraction of topological discs. Indeed, some subjects even skipped the case saying it was "too difficult to work out". The most variation (11 unique cases) was found in the case of two circles transforming into three (figure 4).

Interviews

We supplemented the questionnaire with four free-form face-to-face interviews with experienced designers. The interview subjects were given lunch and were reimbursed for travel expenses.

In our interviews, we were able to confirm many of the principles of morphing listed above. In addition, we learned the following:

1) Artists often wanted to use morphing as a way of animating a figure. For instance, morphing a drawing of a person so that it could have its arms and legs in different positions. This type of morph is very difficult for an algorithm to do accurately without a lot of knowledge of the internal structure and movement constraints of the figure.
2) Artists wanted to be able to put their own signature on a morph. Indeed, a universal complaint of current morphing tools was that all of the morphs looked the same. Artists wanted their version of, say, a square turning into a circle to look different from anyone else's
3) Artists wanted much finer control over the timing and sequencing of their morphs. They wanted to control not only the order of subparts of the morph, but also the acceleration curves. They also wanted to be able to insert other animation into a morph sequence. For example, the elephant might first have its legs change, then it might turn its head to look at its new legs, and then have its head change.

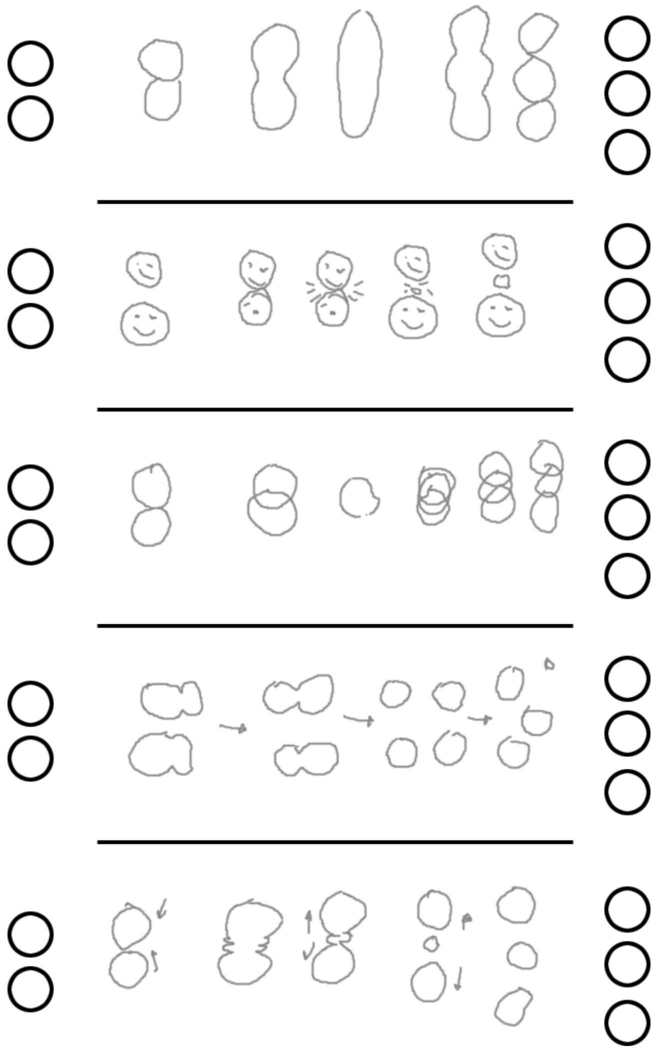

Figure 4: A variety of ways of morphing two circles into three. Note the narrative aspects of several of the cases.

Temporal Reversibility

One fascinating observation in the results is that, due to the assumption of physical properties and simple forces, often the user expectation for a morph will not be time-reversible. That is, A will not morph into B the same way that B will morph into A.

A common example of this is three small circles transforming into one large one. In the forward direction, subjects preferred the small circles to inflate and join, leaving a small hole in the middle that would eventually fill in. In the reverse direction, however, users prefer that the large circle pinch in from the sides and eventually pinch off into three smaller bits. (see Figure 5)

This is particularly significant because all current morph algorithms are time reversible. Indeed, many cases of failed user expectation in morph can be traced to this underlying cause.

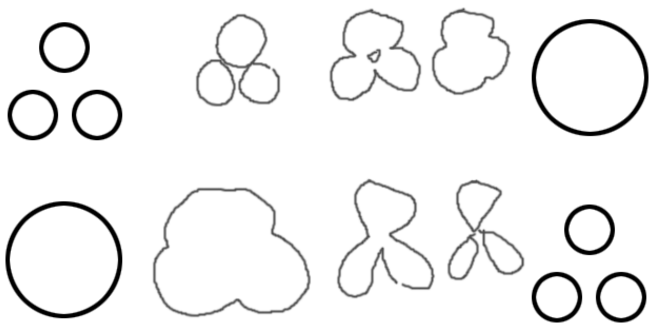

Figure 5: A to B is different from B to A

Primary Features

Another interesting observation was that in a morph of any complexity, users tended to identify a single key feature to "get right", making all other features secondary. What this feature was varied from subject to subject, but the tendency was consistent.

For example, in the case of a 3 morphing into a 4, some subjects would focus on the hole in the four as the primary feature, while others focused on the corner (see Figure 6).

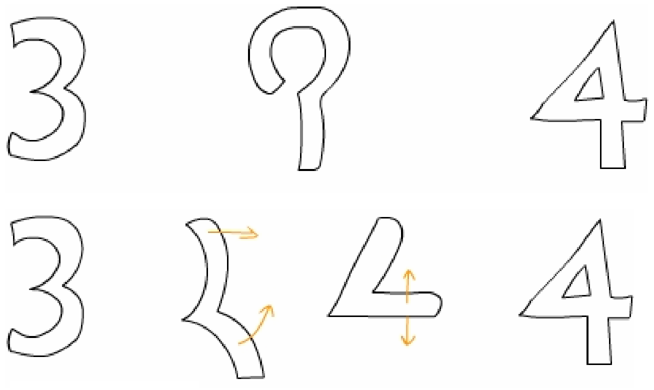

Figure 6: Primary Features (top is hole, bottom is corner

Users tended to limit their morph description to this primary feature, and provide little or no specification to the other parts of the transformation. When shown morphs where the primary feature met expectations but differed in their handling of secondary features, subjects exhibited no preference (and in many cases noticed no difference) between the cases.

USER INTERFACE DESIGN

Based on these findings, we undertook our redesign on the morph user interface. Our design focused on six key areas.

Smart Onion-Skin

In our design, we did not start from a blank slate, but rather repurposed several concepts common to animation user interfaces. In particularly, we used onion-skinning to show the transformation between the first and last frame.

Onion-skinning is the process of showing a sequence of frames on top of each other, each one at a higher degree of transparency and (optionally) a greater offset than the last. Like many systems, ours allowed the user to control the transparency and the offset as desired.

Our onion skinning was "smart" in that the key frames were emphasized, while only in-between frames were faded. Key frames included the first and last frames as well as any frames in-between for which the user made specific edits or adjustments. (Figure 7)

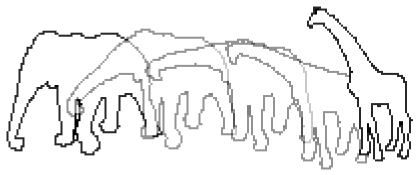

Figure 7: Smart Onion Skin

The idea of onion skinning with variable levels of influence is not novel. The idea is in fact common among traditional paper animators, who often interleave a few sketches done with ink with many in-between frames done with light pencil.[12]

Sub-Area Specification and Manipulation

The primary differentiating factor in our design is the elimination of the specification of individual vertices for correspondence pairs. Instead we rely on sub-area specification.

Sub-areas are initially automatically generated by the system. We do this by computing visually significant differences in the shape's local curvature. In particular, we detect changes of curvature, combining consecutive areas until the areas total at least 10% of the shape's total area. (See Figure 8)

Sub-areas are computed for both the start and end shape and are indicated with circles. Users can establish a correspondence between sub-area by selecting first one area and then the next. Users can also define their own sub-area or delete system-defined ones. Sub-areas are specified by selecting a range of points with a freehand lasso-style tool. In the event users still want to specify correspondences at the vertex level, the system provides for sub-area that consists of a single point, which can be specified by a click.

Sub-areas need not be specified in pairs. The user can define them in whatever order he or she desires. For instance, one could draw all of the sub-areas on the first shape and then define all of the areas on the second. Correspondences do not have to be set between all sub-areas, and the number of sub-areas on the shapes need not match. Any uncorrelated areas will be automatically solved by the computer. This allows for users to concentrate on the areas that they care about while giving just general hints or even ignoring the others they are less concerned about.

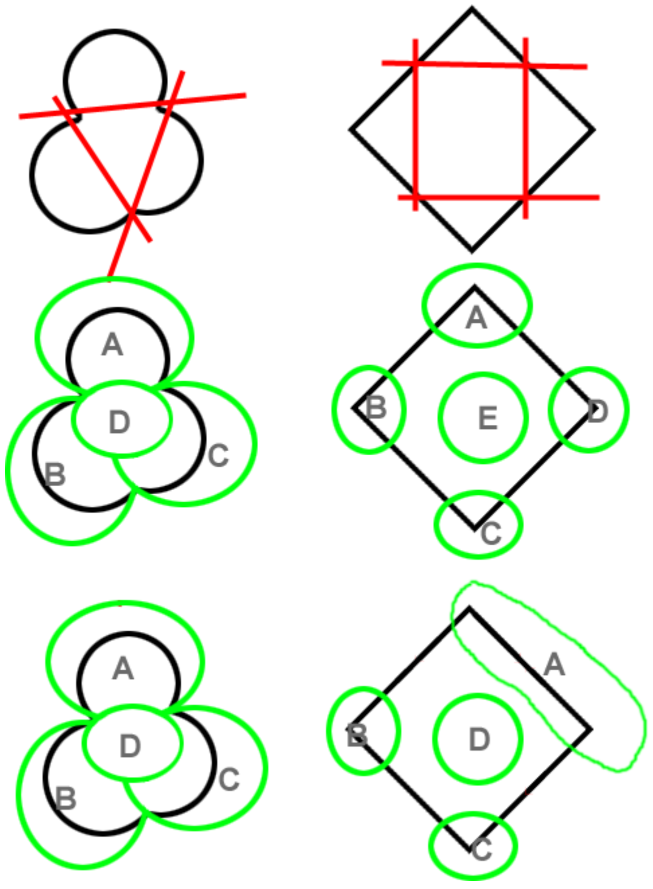

Figure 8: sub-area correspondence

At calculation time, vertex correspondence is computed by using the weighted distance of a particular point from the middle of its sub-area. Points in the center 50% of the sub-area's path are affected solely by their sub-area, while points on the extreme edge area are jointly affected by the sub-areas on either side. This preserves the fuzzy nature of sub-areas while still giving users more specific control when they want it.

Hierarchical Timeline Sequencing

Because users demanded more control of the morphing process, we introduced the notion of a hierarchical timeline object. Initially, the morph is represented on the timeline as a single object. However, this object can be expanded to show the morphs of each sub-area. (Figure 9)

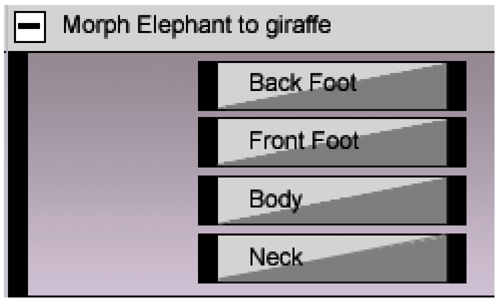

Figure 9: Sub-areas of a morph on timeline

The user can then perform temporal scaling, sequencing, and filtering operations on either the morph sequence itself of the subparts. In figure 10, we can see the user has made the sub-areas morph in series rather than in parallel. The user has also added some acceleration to the vertex interpolation, giving the changes more of a "pop".

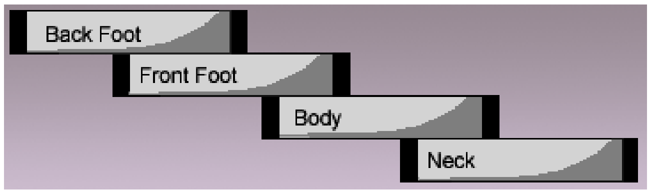

Figure 10: Sub-area scheduling in the timeline

At any particular point in the timeline, the user can manipulate the shape manually. This will introduce a new key frame, and the morph will be recomputed to go from the previous key frame (or start) to the new key frame, then to the following key frame (or end). The timing of these key frames can be manipulated on the timeline or deleted.

Multiple Variations

Our studies showed that they are many possible ways to morph two shapes. Rather than attempt to come up with some sort of compromise "best" morph, we expose all computed possible morph variations to the user and allow them to easily switch between them.

In order to implement this, we do not have a single morph algorithm, but rather several different algorithms, each of which is adept at handling different morph cases. This allows us to have different algorithms that mimic different sets of physical properties (i.e., rigid body morph, fluid volume morph, matrix transformation morph, etc). The algorithms themselves indicate if they have one or more reasonable novel solutions to the current morph, and those that do are added to the result set. "Reasonable" and "novel" are at the discretion of the algorithm implementer.

The different possible morphs are indicated directly on the timeline entity for the morph, as well as on the editing halo of the shape (figure 11). The user can switch between them easily. If the user has edited the details of one variation's timeline, those edits stay with that morph version. The user can even add their own version by copying an existing algorithm if they so desire. This allows for maximum exploration with no pressure to commit to a design too early.

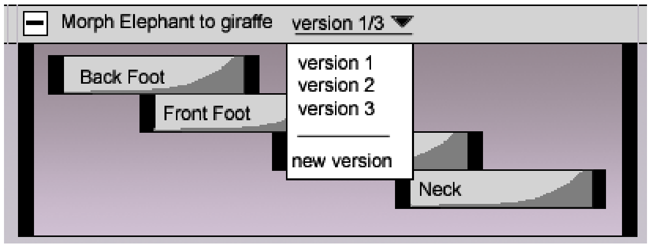

Figure 11: Variations

Morph Description Language

Finally, in an effort to aid in the visualization of different morphs, we overlay small arrows near the sub-areas to indicate the general direction of movement that area will experience. This allows the user to quickly discard cases that he or she knows will not be relevant. (Figure 12)

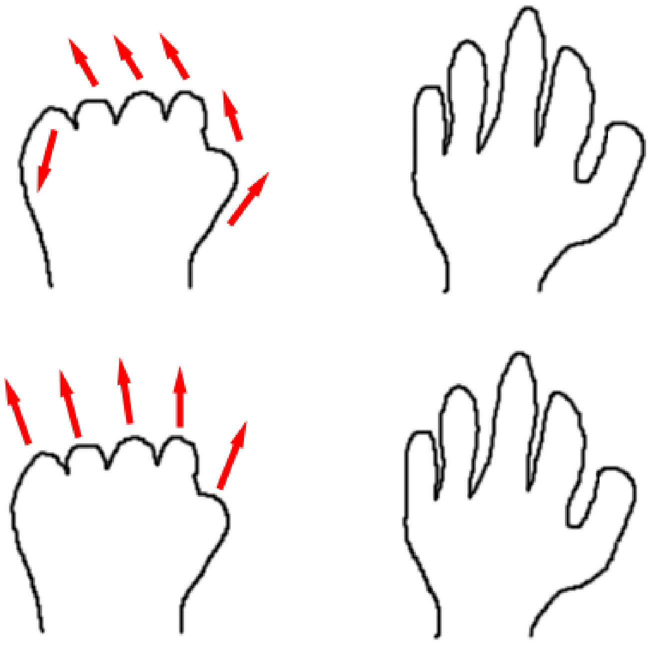

Figure 12: Arrows indicating vertex interpolation allow the user to select the desired morph quickly

Initially, we explored specific types of indicators for different forces such as pinch, pull, stretch, etc. In the end, we found that a simple suggestion of direction was sufficient. We also considered making these arrows editable by the user – allowing them to create arrows themselves to control morph direction and velocity. However, early feedback indicated that the arrows were not sufficient for this task. They cluttered the direct manipulation area of the UI. The hints were too vague to be useful - users who wanted precise control preferred to manipulate the morphs using the timeline directly.

EVALUATION

We have shown sketches and simulations of our new morph user interface to several professional designers, including two of the designers in the original study, and have received positive feedback. Subjects were able to understand the new capabilities and make correct predictions.

We have also had discussions with the development team about our findings. Together, we are trying to develop a set of algorithms which can produce the types of results required.

NEXT STEPS

We are in the process of developing a prototype system that can demonstrate all of the features of our new UI.

As soon as this is completed, we will conduct further usability studies to make sure that our design satisfies users' needs.

As it happens, our design had implications that went beyond the scope of a simple morphing UI. In particular are the notions of improved drawing primitives for character animation and an improved timeline representation.

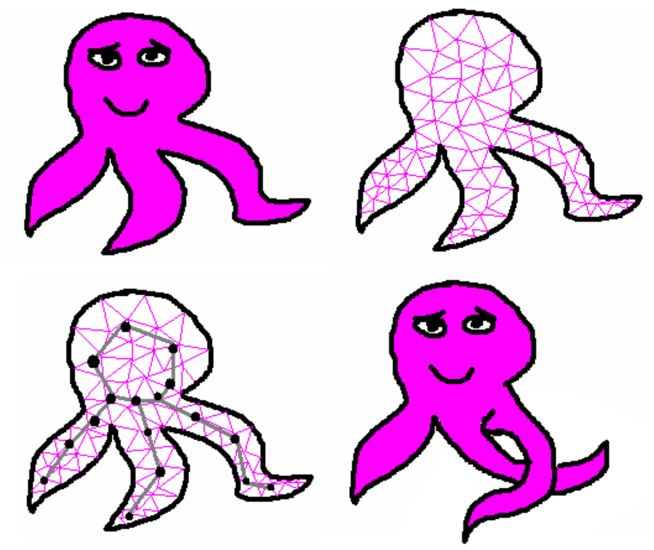

Figure 13: Skinned 2D Polygons

To help the character animation problem, we are developing a new graphics primitive known as a skinned polygon. A skinned polygon is a 2D polygon with a bone structure inside of it – basically a simplified version of the boned animation primitives common in 3D animation packages. The bones are connected with joints, which can have constrained angles of movement. The user can then do animation by simply moving the bones, and the system will handle transforming the skin around them. (Figure 13) Such a system can handle many of the cases that are extremely difficult in shape morph, such as overlapping segments and velocity-related wiggle.

We are also investigating a new timeline representation that is better suited for displaying hierarchical segments and for working rough in time in general. The current timeline common in animation tools is insufficient for these tasks, forcing the animator to work very exactly and giving no support for loose work, exploration, and multiple versions.

ACKNOWLEDGMENTS

We would like to acknowledge the assistance of Professor Xiaobo Lu, Baining Guo, YingQing Xu, and Harry Shum for their support of this work.

REFERENCES

1. Wolberg, G. Image Morphing Survey. *The Visual Computer*, 14, 8/9, 1998
2. Marc Alexa, and Wolfgang Müller. Visualization by Meta-morphosis, *IEEE Visualization '98 Late Breaking Hot Topics Proceedings*, 1998
3. Marc Alexa, Daniel Cohen-Or and David Levin, As-Rigid-As-Possible Shape Interpolation, SIGGRAPH 2000, New Orleans, LA USA
4. Michael Terry, and Elizabeth D. Mynatt, Recognizing Creative Needs in User Interface Design, *C&C'02*, October 14–16, 2002
5. Shapira, M. and A. Rappoport. Shape Blending using the Star-skeleton Representation. *IEEE CG&A*, 15:44-50, 1995.
6. Wolfgang Müller, and Marc Alexa, Using Morphing For Information Visualization, *NPIV 98* Bethesda MD USA
7. Vitaly Surazhsky and Craig Gotsman, Controllable Morphing of Compatible Planar Triangulations, *ACM Transactions on Graphics*, Vol. 20, No. 4, October 2001, Pages 203–231.
8. Yonghong Zhao, , Hong-Yang Ong, Tiow-Seng Tan and Yongguan Xiao, Interactive Control of Component-based Morphing, *Eurographics /SIGGRAPH Symposium on Computer Animation* , 2003
9. Adobe Illustrator, Software made by Adobe, Inc. http://www.adobe.com/illustrator
10. Macromedia Flash, Software by Macromedia, Inc. http://www.macromedia.com/flash
11. Thomas Sederberg, Eugene Greenwood. A Physically-Based Approach to 2D shape blending, *Proc. SIGGRAPH 1992*, p.25-34
12. Frank Thomas, The Illusion of Life: Disney Animation, *Hyperion Press,* 1995, 575 pages
13. Zhang, Y.A Fuzzy Approach to Digital Image Warping. *IEEE Computer Graphics and Applications*, pp. 33-41, 1996