Building User-Controlled 3D Models and Animations for Inherently-3D Construction Tasks: Which Tool, Which Representation?

Guy Zimmerman, Julie Barnes, and Laura Leventhal

Department of Computer Science
Bowling Green State University
Bowling Green, OH 43403 USA
+1-419-372-2283
fax +1-419-372-8061
gzimmer@cs.bgsu.edu

Abstract. In this paper, we first define a class of problems that we have dubbed *inherently-3D*, which we believe should lend themselves to solutions that include user-controlled 3D models and animations. We next give a comparative discussion of two tools that we used to create presentations: Cosmo™Worlds and Flash. The presentations included text, pictures, and user-controlled 3D models or animations. We evaluated the two tools along two dimensions: 1) how well the tools support presentation development and 2) the effectiveness of the resultant presentations. From the first evaluation, we concluded that Flash in its current form was the more complete development environment. For a developer to integrate VRML into cohesive presentations required a more comprehensive development environment than is currently available with Cosmo™Worlds. From our second evaluation, based on our usability study, we have made two conclusions. First, our users were quite successful in completing the inherently-3D construction task, regardless of which presentation (Shockwave or VRML) they saw. Second, we found that enhancing the VRML models and including multiple perspectives in Shockwave animations were equally effective at reducing errors as compared to a more primitive VRML. Based on our results we believe that for tasks of the 3D-complexity that we used, Flash is the clear choice. Flash was easier to use to develop the presentations and the presentation was as effective as the model that we built with Cosmo™Worlds and Java. Finally, we postulate a relationship between inherently-3D task complexity and the relative effectiveness of the VRML presentation.

1 Introduction

In recent years, a wide variety of media representations have become accessible via World Wide Web (WWW) browsers. These newer representations have the potential to greatly expand the types of problems and solutions that are deliverable on the Web. Developers of web presentations would seem to face at least two challenges as a direct result: 1) selecting a best fit of technologies to the problem and solution at hand and 2) achieving a balance between the choice of tool to build the presentation

M. Reed Little and L. Nigay (Eds.): EHCI 2001, LNCS 2254, pp. 193–206, 2001.

and the effectiveness of the presentation which can be generated. In general, the designer will likely choose the technologies and tools that have the least cost in terms of resources and training, and at the same time deliver a presentation which is highly effective for the target task and user population.

Making this choice of media representations and tools is often non-trivial; recent advances in computer hardware and software have led to the development of a number of representations including: text, static images, streaming audio/video, computer animation and virtual reality (VR). There are a number of tools that support these formats; some of the tools support only one representation where others may be useful for a variety of representations.

Not all presentations are equally effective for a given task. In our recent work, we have studied the effectiveness of different media combinations for presenting directions for construction tasks. We have found that presentations that include both visual and textual components are more effective than presentations which are either strictly textual or strictly visual. In particular, we have found that textual/visual presentations that incorporate user-controlled 3D models and animations in the forms of simple virtual reality (VR) or three-dimensional (3D) models as visual components can be very effective in the delivery of this type of instruction set. [1]

The development of even the simplest VR models that we have employed is time and resource intensive. In order for such models to have any widespread practical application for the types of tasks that we are studying, it is imperative to have tools that lessen the demands on designer resources. In this paper, we first give some background on the problem domain of current interest: delivery of instructions for an inherently-3D construction task. We argue that presentations that incorporate user-controlled 3D models and/or animations would seem to be a good fit for these types of problems. Secondly, we give a comparative discussion of two tools we used to create presentations for inherently-3D construction tasks: Cosmo™Worlds and Flash. Next we describe the two presentations that we built using these two tools; the presentations included user-controlled 3D models and animations. We evaluate these tools along two dimensions: how well the tool supports presentation development and the effectiveness of the resultant presentations. In our conclusion, we analyze the cost-benefit issues that arise and make recommendations for choosing a tool.

2 Problem Domain: Instructions for Construction Tasks

The steady increase of processing power on the local desktop has fostered the introduction of a variety of Web-deliverable media formats including: pictures (JPEG, PNG, GIF), animations (Shockwave, Java applets) and audio/video (MPEG, QuickTime, RealAudio). Many of the information representations now widely used on the Web are simply adaptations of existing media formats that have been in use for many years. For example, animation has been used for many purposes from entertainment to education; the delivery of an animation via a computer display, in and of itself, is unremarkable. There are however some research issues that warrant investigation, such as how these various representations (like animations) can be integrated into a cohesive, interactive presentation.

There are also a small number of new technologies, such as 3D modeling and VR, which have truly expanded the types of information representations that are available

via the Web. However, to date most of the widespread use of 3D graphics, particularly on the Web, has been in the "eye-candy" domain, e.g., creating animated logos. There appear to be several reasons for this phenomena. First, there is a general lack of understanding among developers of how to use and integrate 3D graphics effectively for various problem domains. In other words, it is unclear as to what types of tasks and users are best served by presentations that incorporate 3D graphics with other representational forms. [1] Secondly, integrating 3D graphics into presentations is potentially time and resource intensive, due to the limited availability of integrated development tools. [2] Finally, only very recently has the processing power on the local desktop risen to the level necessary to perform real-time 3D processing and thus made this a practical option to Web developers.

It would seem obvious that there is a large class of problems which would benefit from the use of 3D representations of information. That is, there are some problems which seem to be "inherently-3D" in nature and thus a solution to the problem that incorporated 3D graphics would yield an optimal solution. Consider the following problem: *Assemble a toy model car*. This is a non-trivial construction task requiring a series of steps. The car is a real object and it "lives" in 3D space. The pieces of the car are fit together, dynamically, in a multitude of (3D) orientations. While assembling such a car, a person would typically view the car from a variety of (3D) perspectives, choosing vantage points which facilitate the user's understanding of the assembly process. The effectiveness of instructions for assembling a toy car would be enhanced with suitable use of 3D graphics. In particular, given access to a manipulable 3D facsimile of the car, the assembler could compare the facsimile with the actual model car from various perspectives during each step of the assembly.

In this paper, we will refer to problems such as *assemble a model car*, as *inherently-3D construction tasks*. We recognize that the term "inherently-3D" is subjective; in fact, part of our research is to develop metrics to quantify this intuitive notion. We are interested in the delivery of instructions for inherently-3D construction tasks. We observe that the instructions for tasks like assembling a toy car have traditionally been delivered in paper form as combinations of text and still pictures. When such directions are made Web-deliverable they are still typically in the form of text and still pictures. Such instructions are notorious for being difficult for the user to follow. We believe that the effectiveness of directions for inherently-3D tasks, delivered via the Web, could be significantly enhanced using some of the newly available 3D technologies.

To investigate this, we have chosen the construction of origami objects as our inherently-3D construction task. Creating an origami object is similar to assembling a toy car in several ways. Most origami tasks have a number of steps. While assembling an origami object, it is often useful to view it from a variety of perspectives. The paper folding task creates an artifact in the real world. Thus, making an origami object is representative of a very broad class of educational/training problems in which a user is given step-by-step instructions and is expected to build something.

In addition there is a complexity spectrum for origami objects, ranging from simple, basically 2D objects with only a few folds, such as a paper hat, to highly 3D objects such as an origami box. This complexity spectrum provides us with a framework to investigate how the 3D characteristics of the real object are related to the 3D characteristics that are conveyed in the instructions.

Additionally, in terms of a benchmark task for usability research, origami has several useful characteristics. Paper folding is a task which is familiar to most people;

most people have made a simple paper airplane. However, many people do not have explicit experience with traditional origami and thus they would need to follow detailed instructions to actually create an origami (i.e., they need to use our presentation to complete the task). For most people creating an origami object is self-motivating; our users are generally compelled to finish our instruction presentations, because they enjoy the task of building the origami object.

Finally, developing presentations to deliver instructions to construct an origami object is a relevant task for the Web. Traditionally, the instructions for foldings are delivered on paper, using text and still pictures. [3] However, numerous Web sites and commercial multimedia products have recently become available; these sites and products convey the directions for origami objects with text, pictures, video and sound. [4]

3 Development Tools

In this paper, we consider the problem of engineering a presentation to deliver instructions for building an origami object, specifically: presentations that include user-controlled 3D models and/or animations. We limited our research to Web-deliverable presentations which integrate the models and/or animations with text and picture versions of the instructions, based on earlier findings that suggest that incorporating visual and verbal information in the presentation aids the user in accomplishing a task.[1] We also required users to be able to control the animations (play, stop, replay). We chose two data formats: VRML and Shockwave. Both VRML and Shockwave permit the creation of user-controlled animations. VRML permits the user to interact directly with the model by rotating and scaling it. Thus the user is able to see the object from any vantage point while a given folding step is being animated. Shockwave animations do not permit the user to manipulate the image, but do permit the display of parallel animations, which allows for a presentation which includes multiple (but static) perspectives. High-end development tools are available for both VRML and Shockwave. Finally, both VRML and Shockwave have Web browser plug-ins, making them Web-deliverable and both can be integrated with text and picture presentations.

In the next two sections, we describe the high-level features of the two tools that we used. In a later section, we will describe how we used the tools.

3.1 VRML Development: Cosmo™Worlds

VRML is a *de facto* Web standard and is arguably the best way to provide interactive, 3D environments in that context. To develop our VRML models we used Cosmo™Worlds which is marketed by Silicon Graphics. In addition to the traditional 3D-modeling tools, Cosmo™Worlds provides GUI tools to manipulate virtually every feature of the VRML specification. The developer can manipulate the scene graph directly using an object browser facility or just by clicking on objects in the development window. VRML nodes can be easily grouped together into a Switch node to allow exactly one of the child nodes to be activated at a time. The keyframe animation mechanism allows the developer to easily animate models and it

automatically chooses the correct VRML interpolation node for a given keyframe sequence. Users can easily set material properties (color) and/or apply texture maps to a given surface. Script node code and I/O events are handled well; Route statements are illustrated visually. One very useful feature is the PEP tool suite. This set of tools allows individual Points, Edges and Polygons (PEP) to be directly manipulated. For example, a selected polygon can be automatically split into two pieces. Finally, an optimization tool is available to reduce the number of polygons, file size and overall complexity.

With all its functionality, VRML still presents some limitations to the problem at hand. Several of the manipulations we wanted to be able to perform on our VRML objects required the functionality of a full-featured programming language. We used Cosmo™Code to develop Java code to manage a user interface and to allow the user to control aspects of the animation. Other researchers have noted the effectiveness of this combination of VRML and Java. In [2], the authors discuss a CAD tool for the virtual assembly of furniture. Client/server applications using Java and VRML are discussed in [5].

We did consider other formats and tools for our work. Two notable candidates were Java 3D and World Up. [8] We rejected Java 3D as being too low-level compared to VRML. World Up, a full-featured 3D system, was rejected for several reasons. It is currently in no sense a Web standard. We also felt it unlikely that many users would have the necessary plug-in. Finally, integrating World Up presentations with HTML was problematic. We did feel that the World Up toolkit may be a better overall development tool in the future for problems of this type, in no small part due to its integrated, object-oriented, scripting feature.

3.2 Shockwave Development: Flash

One of the most popular tools for creating 2D, Shockwave animations for the Web is Macromedia's Flash. Flash provides a development environment for creating animations for Web pages and optimizes the resultant files for fast delivery over the Internet. [6]

The Flash development environment provides the developer with a vector-graphics editor to create graphical objects to populate scenes in Flash animations. The developer creates multiple layers for each scene in order to more easily manipulate the different objects in a scene. Although a scene is composed of multiple layers, hinting at depth, the objects are 2D. The developer must create several scenes as the only way to hint at various perspectives. Shockwave models are not directly manipulable by the user.

A major feature of Flash is its ability to create “tweened” animations. With this feature the developer specifies a starting object, position, shape or color and a final object, position, shape or color and Flash generates the frames in-between the two by interpolation. The developer may specify Shape Hints to guide the interpolation in shape tweening. The developer may specify a Path in a Layer Guide to assist the interpolation in a motion tween.

4 An Evaluation of Development Tools

In this section we will discuss and compare the specific development tools that we used to develop our VRML models and Shockwave animations. In particular, we will discuss the amount of effort that was involved in creating the models for our work using VRML and Shockwave

In order to compare the development tools for VRML and Shockwave, we first defined a standard interface look-and-feel for the overall presentation. The left side of the presentation included (top to bottom) a label denoting the current step number, a scrollable text window, and a window containing a still picture. The right side of the presentation contained (top to bottom) a window for the 3D model (VRML) or animation (Shockwave) and controls for the user to manipulate the presentation. These controls included buttons to navigate through the different steps of the presentation and controls to manipulate the 3D model or animation. There were some minor differences in the two presentations. (See Figures 1 and 2.)

Next we defined a benchmark task for our user to perform: build an origami whale. The whale consisted of 12 distinct steps and 25 distinct folds. Steps 1 through 5 were essentially 2D in the sense that they were folds or fold-unfold combinations on a flat piece of paper. In Steps 6 through 12, the folds were 3D, in the sense that the paper model became truly 3D. Steps 1-3, 6 and 9 formed the body, Steps 4-6 and 10 formed the fins, Steps 7-8 and 11 formed the mouth, and Step 12 formed the tail. Steps 6 and 12 included "inverted" or "hidden" folds, in which all of the fold could only be truly seen in 3D. We felt that the whale, by virtue of the 3D nature of the final origami object and the inverted folds, was at least marginally an inherently-3D construction task.

4.1 VRML

Our VRML whale animation models were integrated into our standard interface using the Cosmo™Player plug-in. The control section contained three spin controls that permitted the user to rotate the model about the three standard axes. There was also a size control that permitted the user to scale the model. A start/stop button allowed the user to control the fold animation at each step. Additionally, users could return the 3D model to its original orientation and size for each step by using a reset button.

We created two different VRML versions for the whale building task. The first version was designated "plain VR" (PVR) and, in some sense, served as a control for us in evaluating Cosmo™Worlds and the effectiveness of the VRML presentation. The construction of the first version used a single VRML Shape node type: indexed face set. There were two vertices corresponding to each fold line (the line along which the paper is to be creased); the coordinates of each of these vertices and the faces to which they belong were manually calculated for keyframes of each animation step. All of these values were stored in the indexed face object. A coordinate Interpolator node and a Script node were used to generate an animation sequence to illustrate each fold. In PVR, the fold line had to be inferred from the lighting/shading of the faces involved in that fold. A Java applet was implemented to generate the user interface which allowed the user to manipulate the model and the animation process. The EAI [7] was used to allow the Java user interface to "communicate" with the VRML

model. The VRML model for the first version was created manually with a text editor and required approximately 40 hours to complete. The user interface applet and associated EAI code was developed using Cosmo™Code and required about 10 hours. An additional 5 hours were required to assemble all of the components (text instructions, Java applet user interface and VRML plug-in window) into a coherent set of Web pages; each page corresponded to one step of the construction process.

In the second version, the VRML component was built in Cosmo™Worlds and was designated as "enhanced VR" (EVR). EVR differed from PVR by the addition of explicitly drawn fold lines. In EVR, when the user first initiates the animation of the step, the fold line to be created is "drawn" onto the model and after a brief pause, the animation of the origami fold process begins. It was hoped that the explicit rendering of the fold line would enhance the effectiveness of the VRML model in the representation of each step.

The construction of the second version took more time than the first. This was a direct result of the explicit animated fold line requirement. As in the first version, the basic VRML Shape node used was indexed face set. Because the underlying geometry of the object changed during subsequent steps of the construction process and each new object required additional fold lines, we found it most expedient to create a separate indexed face set object for each step. The addition of the actual fold lines was accomplished using the texture mapping feature in Cosmo™Worlds. However, since there is no integrated facility for creating textures, we created the fold line images using PhotoShop and then imported these into Cosmo™Worlds. This was a non-trivial task since the fold line image geometry had to precisely match the geometry of its corresponding indexed face set. All of the calculations necessary to establish the correct mapping of the image to the corresponding face were done manually. Managing the changing geometry of the indexed face sets was greatly facilitated by the PEP suite of tools in Cosmo™Worlds. For example, creating a fold requires one polygon to become two; the splitting PEP tool made this very easy to do. The additional number of indexed face set objects, combined with the animation of the fold lines, required additional Script node and Java coding to manage the transitions between steps. Fortunately, the Java applet for the control user interface and the Java EAI code from the first version required only modest alterations.

Figure 1 shows a screen from our integrated presentation of instructions for Step 6 of the origami whale. Step 6 is a 3D step in which some of the paper is hidden inside of the object. This figure shows the EVR presentation where the 3D model has been rotated by the user in order to view the model from an alternate perspective.

4.2 Shockwave

We also created an integrated presentation of the instructions for folding the whale, using the Flash development tool. We found that in Flash, the animations for the folds were easily created. Motion, shape, and color tweening were all used to create the images of folding paper. In the final Shockwave movie, buttons were included so that the user could interact with the movie. This interaction was limited to 1) starting and stopping the animation, 2) navigating between steps or scenes, and 3) changing viewpoints (selecting an alternate version of some scenes). The user could not directly manipulate the images of the paper model. The user could only replay the Shockwave

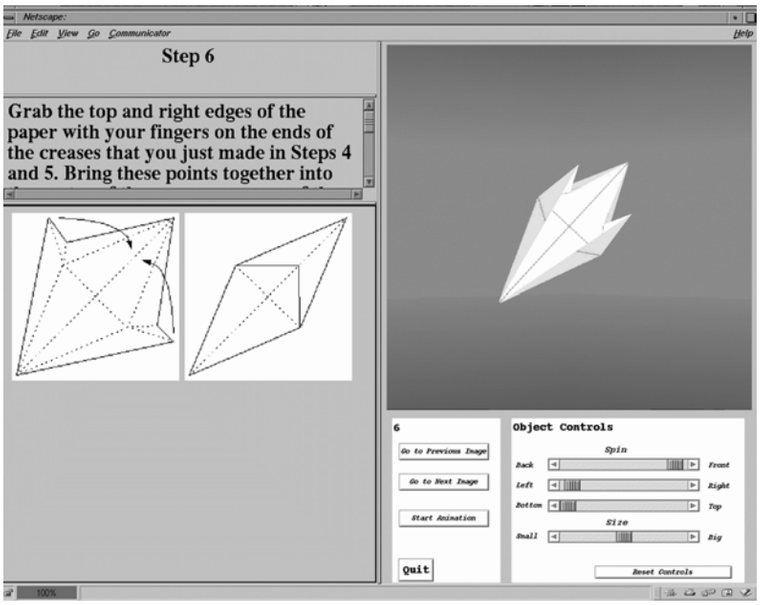

Fig. 1. The Enhanced VRML (EVR) Version of the Interface

animations that had been created. The individual images that we used in Flash were either imported GIF images or images that were created with the Flash drawing tool. The fold lines were drawn as part of these images.

The 3D steps of the folding, particularly Steps 6 and 12 with the hidden paper, were a challenge in Flash. We opted to present a second, parallel perspective of these folds. Hence we were able to show the alternate perspectives of some of the scenes. The user was still restricted to the viewpoints determined to be most meaningful by the developer of the movie.

The final interface of text, pictures and Shockwave animation was itself created in Flash and included a scrollable text window, window for a still picture that showed the folds for the current step, the animated folds, controls and directions for the controls. Figure 2 shows a screen from the Flash presentation. It shows the same step as Figure 1. Because the Shockwave model is itself 2D and only shows one perspective for this step with hidden folds we included a second parallel animation from another perspective.

4.3 Evaluation

In terms of development, Cosmo™Worlds has a complete set of features for generating objects, color and animations; as a tool for creating stand-alone VRML worlds it is excellent. However, in order to create a presentation incorporating VRML, we found this tool to be inadequate. In particular, the lack of a facility for creating custom texture maps was limiting. We were also hampered in generating our presentation because we had to use Java to create and to manage the integrated text,

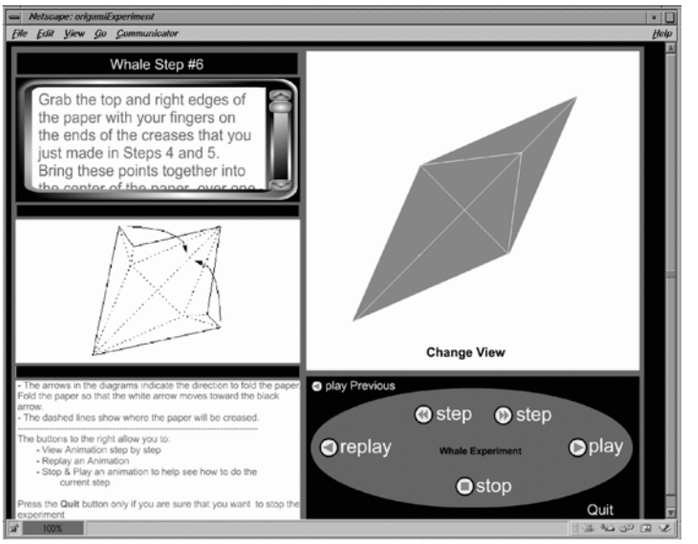

Fig. 2. The Flash Version of the Interfaces

picture and VRML components. We note that this is not a deficiency of Cosmo™Worlds *per se*, but rather points to a vacuum in the set of tools that are available for delivery of VRML over the Web.

Flash was quite easy and fast for the application at hand. To develop the animation, we had a keyframe of the initial image of the paper at the current step, several keyframes of intermediate images, and a final frame. Flash tweened these images into our animation. Color interpolation and shape tweening can give the effect of a paper fold. The step buttons permitted the user to step through our intermediate keyframes. The Flash development environment includes a draw tool and the capability of importing GIF images, so it was also quite easy for us to build the keyframes. Finally, we were able to create our entire integrated presentation in Flash, rather than using a second tool, such as Java.

In evaluating the two development tools, Cosmo™Worlds and Flash, we conclude that Flash in its current form is the more complete environment. By contrast, Cosmo™Worlds supports only the development of stand-alone VRML. For a developer to integrate VRML into cohesive presentations requires a more comprehensive development environment than is currently available.

5 An Evaluation of the Presentations

We concluded in the previous section that the Flash development tool is the more complete tool for developing integrated Shockwave animations as compared to Cosmo™Worlds for developing integrated VRML presentations. In other words, from the developer's perspective, it is likely to be more work to develop an integrated VRML presentation. In this section, we consider whether that extra work is

worthwhile. We compare the effectiveness of the VRML and Shockwave presentations for users who are trying to complete an inherently-3D construction task.

5.1 Methodology, Users, and Materials

The task for our users was to construct an origami whale following the standardized presentation and interface look-and-feel that we described in the previous section. Users saw a presentation of instructions for folding a whale, in one of three treatments: (1) PVR: containing text, still-images, and plain VRML (no fold lines), (2) EVR: containing text, still-images, and VRML (enhanced renderings with fold lines), and (3) FLASH: containing text, still-images, and Shockwave animations. User performance was measured and assessed by the number of correct folds and the number of error folds in the whale that each user created.

There were 24 users for this study, with eight per treatment. Users were sophomores and juniors enrolled in computer science classes at Bowling Green State University. All were highly computer literate. Each user viewed only one of the three presentations, PVR, EVR or FLASH. The instructions were presented with a Silicon Graphics O2 computer with a 17 inch monitor. Users viewed the presentations with *Netscape 4.0*.

5.2 Procedure

All of the users performed the task in the CHIL lab (Hayes 227, Bowling Green State University). Users arrived at the lab and completed consent materials. Next the users received training. For the users who would see the PVR or EVR presentation, they first completed an interactive training session about VRML and the tool set for the 3D-model presentation in this version of the instructions. Users in the FLASH conditions saw an interactive tutorial that illustrated the animation controls. This training took about 8 minutes to complete, unless the users had questions.

All users then received training in paper folding from computerized instructions. In this phase of the training, users folded a stylized paper airplane which had five steps and eight folds. The computerized instructions were presented in whichever of the three treatments that the user was to receive.

Next the experimenter explained that origami is the ancient art of Japanese paper folding. Users were given a piece of special origami paper and were told to fold the whale by following the presented instructions. There were no time limits for how long subjects were given to fold the whale.

5.3 Results

Our dependent variables were the number of correct folds and number of error folds that subjects made. In order to assess the correctness of the whale folds, each existing fold was graded by three criteria:

- Was the fold in the right place?
- Was the direction of the fold correct?
- Was the fold the right size?

In order to be a correct fold, the fold had to be correct on all three of these criteria. Some subjects made a correct fold and then recreased the same fold in a slightly different position. We scored both of these folds as correct.

The subjects in all of the treatments did very well. The average percentage of correct folds was 86.5% across the three treatments and there was no statistical difference by treatment. In other words, on average all groups eventually got more than 86% of the folds right.

We conducted a between-subjects MANOVA of total number of correct folds and total number of error folds, by presentation condition. The MANOVA was significant (Roy's Greatest Root = 0.496 ($F(2,21) = 5.2$, $p < 0.01$)). A univariate ANOVA of the total number of correct folds by presentation condition was not significant. A univariate ANOVA of the total number of error folds by presentation condition was significant ($F(2,21) = 4.6$, $p < 0.02$). The average number of error folds by treatments was PVR, 10.0, EVR, 4.9, and FLASH, 4.3. Post-hoc tests (Fisher's PLSD) indicated that, when considered pairwise, PVR was significantly different from both EVR and FLASH. EVR and FLASH were not significantly different from each other.

In other words, both EVR and FLASH reduced the number of error folds as compared to PVR. Neither EVR or FLASH improved the likelihood that the final whale would be correct, but did reduce the number of errors along the way. The baseline presentation (PVR) was the least effective of the three, although it was still quite effective. Adding explicit fold lines in EVR or having the 2D images with multiple perspectives and fold lines in FLASH were even more effective.

From our usability study we drew two conclusions. First, our users were quite successful in folding the whale, regardless of which presentation they saw. This result is consistent with our previous studies which indicate that presentations that include both visual and textual information are likely to be more useful than either visual or textual information alone. Second, we found that enhancing the VRML models with fold lines and including multiple perspectives in Shockwave animations were equally effective at reducing folding errors as compared to more primitive VRML.

6 Summary and Conclusions

In this paper, we identified a class of problems that we termed inherently-3D construction tasks. Our previous research has indicated that when delivering instructions for tasks of this type on the Web, multiple media (visual and textual) are more effective than a single type of media. User-controlled 3D models and animations are examples of visual presentations that can be used effectively with text. We considered two tools for developing 3D models or simulated 3D models for the Web: Cosmo™Worlds to generate VRML models and Flash to generate Shockwave animations. We evaluated these two tools in terms of effectiveness of the development environment and effectiveness of the presentation that could be developed with each tool.

We conclude that in terms of the development environment, Flash can be used to quickly develop a simulated 3D animation, because it provides an integrated environment for developing for the Web. Also, Flash does not require the developer to create an actual 3D model to manipulate. By comparison, development of 3D models in VRML is much more difficult. While Cosmo™Worlds provides a VRML

environment, it lacks features for integration as compared to Flash. The VRML models themselves are more complex. For example, in VRML to show fold lines in our model, we were forced to superimpose a texture map on our images. In Flash, lines are simply a part of the image.

In terms of the effectiveness of the resultant presentations, we found that both FLASH and EVR were equally effective as compared to PVR, although all three treatments had a very high success rate for folding the origami whale. We believe that this was because both FLASH and EVR better presented the 3D aspects of the object, via multiple perspectives or fold lines, respectively.

Which tools should a developer select when delivering instructions for inherently-3D tasks? Based on our results we believe that for tasks of the inherent 3D-complexity of our origami whale, Flash is the clear choice. For our whale, Flash was easier to use to develop the presentations and the presentation was as effective as the model that we built with Cosmo™Worlds and Java.

In retrospect, we believe that the construction task did not have enough inherently-3D features to benefit from the additional realism and functionality that the VRML provided. We feel that as 3D-complexity of the task increases, then the extra overhead of VRML would be justified. Figure 3 shows this postulated relationship. We speculate that the whale folding task was at the threshold or cross point. In our future research, we intend to explore this issue further by studying origami objects that have more hidden and inverted folds than the whale did. We are also currently in process of trying to link a cognitive measure with specific characteristics of the origami object as a way to quantify "inherently-3D."

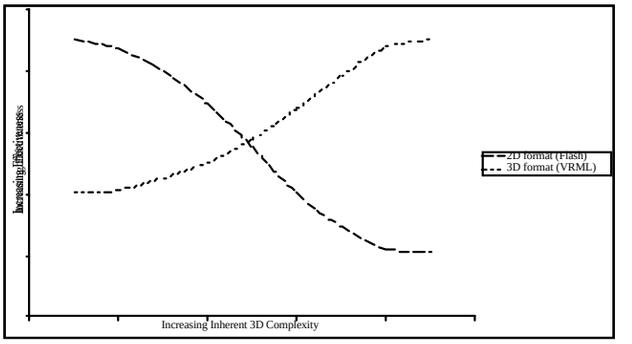

Fig. 3. Postulated presentation effectiveness as a function of 3D complexity

We also conclude that future development environments for VRML models could greatly benefit by incorporating some of the features of the Flash development tool. Specifically, we note that future VRML development tools should support an integrated approach to:

- HTML
- VRML 3D modeling
- VRML Script Node programming and ROUTE statements
- Java (for user interface)

- EAI (for Java-VRML interaction)
- Image manipulation: (for textures, etc)

Acknowledgements

We thank Keith Ebare, Lisa Weihl, Tom Stoltz Brian Ameling, and the conference reviewers for their assistance on this project.

References

1. Leventhal, L., Barnes, J., Zimmerman, G., Weihl, L.: Multiple Web Representations of Instructions on a Procedural Task: An Empirical Study. Submitted to: *Behaviour and Information Technology*. (2000)
2. Nousch, M., Jung, B.: CAD on the World Wide Web: Virtual Assembly of Furniture with BEAVER. The Fourth International Conference on the Virtual Reality Modeling, Language and Web 3D Technologies (February 1999) http://www.clab.de/vrml99/home.html
3. Honda: The World of Origami, Japan Publications Trading Co., Tokyo (1965)
4. Cassady & Greene: Origami: The Secret Life of Paper (1996) ISBN 1-56482-104-8
5. Jung, B., Milde, J.: An Open Virtual Environment for Autonomous Agents Using VRML and Java. The Fourth International Conference on the Virtual Reality Modeling, Language and Web 3D Technologies (February 1999) http://www.clab.de/vrml99/home.html
6. Macromedia, Inc.: Macromedia Flash 4: Using Flash. Macromedia, Inc., San Francisco (1999)
7. Marrin, C., Couch, J.: Specification of the VRML/EAI Interface. The web 3D Consortium. Online resource. (January 1999) http://www.web3d.org/WorkingGroups/vrml-eai/
8. Sense8 Corporation: World Up User's Guide. Sense8 Corporation, Mill Valley, CA (1997)

Discussion

J. Willans: Do you have data on how subjects used each of the representations during construction?
G. Zimmerman: Subjects flipped between representations, especially with difficult manipulations. Text was the least favoured. We have video and interaction logs, but we have not analyzed the data yet.

N. Graham: Many assembly tasks involve both hands. Do you have ideas about how to adapt a system like yours to these tasks?
G. Zimmerman: We have a long wish list of techonologies we'd like to investigate, e.g. immersive virtual reality. We do not see a solution to the problem of requiring a mouse click to move to the next step.

P. Curzon: A common problem following construction instructions is that you think you understand what to do, and only realize several steps later that you went wrong. Do you think that giving the user the ability to construct the object virtually on the

screen would be a good use of 3D as it would allow the system to give feedback on errors?

G. Zimmerman: That's not an area we have looked into. The aim was just to see how a web delivered set of instructions could make things easier.
J. Willans: A related question is one that they have been asking at Nottingham University in the UK, which related to what are virtual environment good for?
G. Zimmerman: This is one of the questions we are trying to answer also.