SYNCHRONIZING USER INTERACTIONS WITH MULTIPLE PARAMETRIZED THREE-DIMENSIONAL WEB GRAPHICS

Martin Kraus
Computer Graphics & Visualization Group, Technische Universität München, Boltzmannstraße 3, 85748 Garching, Germany

Keywords: Three-dimensional graphics, Web graphics, Parametrized graphics, Interactive graphics, Interactive illustrations, Synchronized graphics.

Abstract: The synchronization of a single user's interactions with multiple parametrized graphics in a single web page is an important and powerful feature of interactive illustrations. In this work we present an interface to specify this kind of synchronization and its implementation in a Java applet for rendering and interacting with three-dimensional web graphics. Our design allows authors of web pages to synchronize the parameters of groups of interactive graphics within a web page by specifying identical text labels for all applets of the same group. Further labels may be specified to synchronize changes of the viewpoints and/or of the magnification. This design allows for multiple, independently synchronized groups of graphics on the same web page as well as any combination of synchronized parameters, viewpoints and magnification factors. Only three additional applet parameters are necessary, which are not restricted by any syntax.

1 INTRODUCTION

The support of web browsers for Java applets has led to the development of many interactive web graphics for illustration purposes. While most of these graphics are restricted to two dimensions, there are also many applets for three-dimensional illustrations. Moreover, there are several frameworks for the design of interactive illustrations that address the problem of reducing the associated development costs, which are usually significantly higher than the costs for static illustrations.

One particular approach to the design of interactive illustrations are "parametrized graphics" (Kraus, 2008). In this case, the underlying scene depends on a set of parameters and is rebuilt (usually from scratch) whenever any of the parameters is changed by the user or due to an automatic animation. In general, this approach simplifies the development of interactive illustrations dramatically as shown, for example, by the "Wolfram Demonstrations Project" (Wolfram Research, 2008). Another example for this approach is the related Java applet "LiveGraphics3D" (Kraus, 2002), which serves as the test bed for our work.

Traditionally, many static illustrations accompany a single main text. On the other hand, the development of interactive illustrations and graphics is usually too costly to produce a similar amount of interactive illustrations. Moreover, some viewers for interactive graphics are restricted to a single graphics at a time. Thus, in most cases, only a single interactive illustration is included in each document. Therefore, the problem of synchronizing multiple interactive illustrations in the same document has not received much attention in the past.

However, the reduction of development costs enabled by parametrized graphics already resulted in large collections of interactive illustrations and is therefore very likely to lead to web pages and electronic documents that include multiple interactive illustrations. Synchronizing these illustrations is not always necessary; however, there are several important applications of synchronized graphics, such as multiple views from different points of view, overview+focus techniques, multiple "filtered" views, consistent step-by-step illustrations for variable parameters, and sliders, dials, virtual trackballs, or other 2D and 3D widgets, which provide alternative interfaces for manipulating parameters of interactive graphics.

While all these techniques can also be implemented within a single graphics, the synchronization of multiple graphics is more general and simplifies the development of both, the viewer and the ac-

Kraus M. (2009).
SYNCHRONIZING USER INTERACTIONS WITH MULTIPLE PARAMETRIZED THREE-DIMENSIONAL WEB GRAPHICS.
In *Proceedings of the Fourth International Conference on Computer Graphics Theory and Applications*, pages 401-404
DOI: 10.5220/0001773104010404

tual content. Moreover, side-by-side comparisons of graphics embedded in hypermedia documents are often preferable for presentation purposes and can only be achieved by synchronizing multiple graphics.

In this work, we developed an interface to specify the synchronization of multiple illustrations by analyzing basic properties and requirements of the synchronization mechanism. This allowed us to reduce the interface to the definition of a few (optional) labels per graphics. These text labels are not subject to any syntax; thus, no syntax errors are possible and authors do not have to learn any syntax rules or restrictions. Instead, the labels may be chosen arbitrarily and are only tested for equality. We also present a prototypical implementation of this interface within the Java applet LiveGraphics3D.

2 RELATED WORK

A survey of approaches to interactive illustrations in electronic documents has been published by Kraus (Kraus, 2008) with particular emphasis on the implementation of parametrized graphics within the Java applet LiveGraphics3D (Kraus, 2002), which renders *Mathematica* graphics (Wolfram, 2003) with several restrictions. The “Wolfram Demonstrations Project” (Wolfram Research, 2008) provides a web platform to archive and distribute more general interactive parametrized graphics than supported by LiveGraphics3D, i.e., arbitrary *Mathematica* graphics (Wolfram, 2003).

Communication between Java applets, which is necessary for synchronization, has been discussed, for example, by Andrew Meckler (Meckler, 1997) and has been implemented in various Java applets for interactive web graphics, e.g., “Jmol” (Jmol, 2008) and “Modern CA” (ModernCA, 2008).

3 SYNCHRONIZING PARAMETRIZED GRAPHICS

There are various ways of specifying synchronizations of interactive graphics, which differ significantly in the syntactical overhead as well as in their expressive power regarding the specification of different kinds of synchronizations, e.g., master-slave relations or selective synchronization of particular state variables.

In this section, we first discuss some features of the synchronization of parametrized graphics, which are particularly important since they determine the required expressive power of an interface for the specification of synchronized graphics. Based on this discussion, we develop the structure of our proposed interface.

3.1 Properties of the Synchronization

3.1.1 Symmetric vs. Asymmetric Synchronization

If the interface only allows for the specification of symmetric synchronizations of parametrized graphics, no “master-slave” relation can be specified, where one graphics, the “master,” is manipulated and a second graphics, the “slave,” follows the changes while manipulations of the “slave” are not forwarded to the “master.” However, in almost all actual scenarios symmetric synchronizations are sufficient as can be shown by considering the following two cases:

1. The “slave” cannot be manipulated. In this case, there is no difference between a symmetric synchronization and a “master-slave” synchronization because the “slave” never produces events that have to be synchronized.
2. The “slave” can be manipulated. In this case, there is a clear difference between a symmetric synchronization and a “master-slave” synchronization. In particular, there are some disadvantages of the latter: Firstly, the states of the “slave” and the “master” may be different, i.e., unsynchronized. And secondly, if the states of the two graphics are different and the “master” is manipulated, the state of the “slave” changes discontinuously due to the synchronization. Thus, this scenario is usually highly undesirable.

We conclude that the “master-slave” synchronization of parametrized graphics is unnecessary since it is either indistinguishable from a symmetric synchronization or results in inconsistent states and discontinuously changes of the state of the “slave.”

3.1.2 Independently Synchronized Groups

Suppose there are four graphics A, B, C, and D in a single document. (See also Figure 1.) In many cases, only a subset of all graphics should be synchronized, which might, however, consist of more than two graphics. It is also conceivable that multiple subsets should be synchronized independently, for example, it might be necessary to synchronize the pair $\{A,C\}$ independently of the pair $\{B,D\}$. Thus, it is necessary to allow for the specification of any number of independently synchronized subsets of any size.

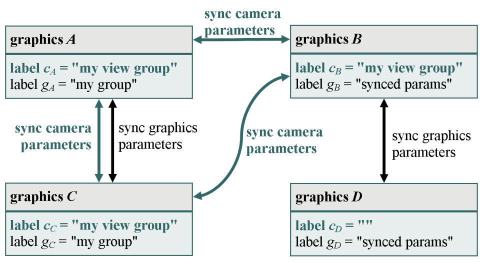

Figure 1: In this example, graphics A, B, and C synchronize camera parameters since they specify identical camera labels $c_A = c_B = c_C$; graphics A and C synchronize graphics parameters (because $g_A = g_C$) and graphics B and D synchronize their graphics parameters independently ($g_B = g_D$).

However, it is not necessary to be able to specify intersecting subsets, e.g., the pairs $\{A,B\}$ and $\{B,C\}$, because the specification of intersecting subsets can always be replaced by specifying the union of these subsets, in our example $\{A,B,C\}$, due to the implied synchronization of all elements of the intersecting subsets.

3.1.3 Synchronized State Variables

Depending on the functionality offered by a renderer for parametrized graphics, several groups of state variables should by independently synchronizable, for example, camera parameters, parameters of the graphics, or a time parameter for animations.

However, in several scenarios it is preferable to synchronize only a subset of the camera parameters. For example, the field of view should usually not be synchronized between a focused view and the corresponding overview while other camera parameters should be synchronized.

The synchronization of parameters of multiple graphics requires the specification of corresponding parameters of these graphics. However, assuming that the author of a web page has full control over the included graphics, the easiest way to specify correspondences between parameters is naming them identically in all included graphics.

For animations, it is preferable to introduce a time parameter with a user-specified name. Thus, the time variable of an animation can be synchronized with any other parameter of another graphics just by choosing identical names.

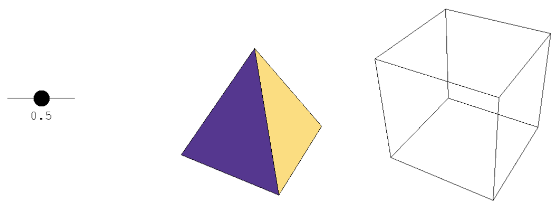

Figure 2: Initial rendering of three synchronized applets.

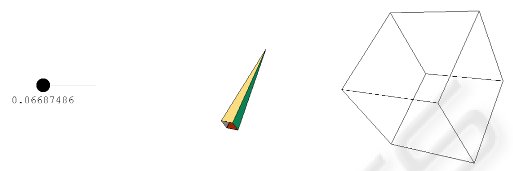

Figure 3: Rendering of the three applets in Figure 2 after the point in the left-most applet has been dragged to change the variable x and the right-most applet has been rotated to change the camera parameters. The applet in the middle is synchronized with both changes.

3.2 Specifying Synchronized Graphics

Based on the requirements discussed in the previous section, we conclude that for each document a set of non-intersecting sets of synchronized graphics has to be specified for each group of synchronized state variables (e.g., camera parameters, graphics parameters, or time).

We propose to specify these sets by specifying a label for each group of state variables and each graphics. If the labels of two (or more) graphics for the same group of state variables are identical, these state variables are synchronized. An example with two groups of state variables (camera parameters and graphics parameters) is illustrated in Figure 1.

There are several advantages of this interface in comparison to previously presented interfaces for the specification of synchronizations (Jmol, 2008; ModernCA, 2008):

- The specification is completely symmetric: all graphics are treated the same way.
- For each group of state variables, any number of non-intersecting sets of graphics can be specified without the need to specify lists of graphics.
- There is no need to name graphics nor to reference graphics by any other means.
- There is no possibility for syntax errors or any other inconsistencies (if labels are not identical, there is just no synchronization).

Furthermore, the design is easily extensible for additional groups of state variables by introducing additional labels and no additional syntax has to be learned by authors of web pages.

4 IMPLEMENTATION IN A JAVA APPLET

We implemented our proposed interface within the Java applet LiveGraphics3D (Kraus, 2002). From several use cases we decided that three groups of state variables may be synchronized independently: graphics parameters, which are called “independent variables” in LiveGraphics3D (Kraus, 2002); camera parameters, which include the *Mathematica* (Wolfram, 2003) options `ViewPoint` and `ViewVertical`; and a magnification parameter, which is specific to LiveGraphics3D. One text label may be specified for each of these groups and for each applet by means of three applet parameters: `SYNCHRONIZED_VARIABLES_GROUP`, `SYNCHRONIZED_VIEWPOINT_GROUP`, and `SYNCHRONIZED_MAGNIFICATION_GROUP`. A example with three applets is illustrated in Figures 2 and 3.

On initialization, each applet builds three lists of applets within the same document that specify the same label in one of the three applet parameters for synchronization. If such an applet was found, it is appended to one or more of the three lists and requested to add a reference back to the former applet. This is necessary because applets can be initialized in any order by the Java virtual machine. During runtime, each change of the graphics parameters or the camera parameters of one applet is sent to all applets in the corresponding list of synchronized applets.

4.1 Synchronization of Time Variables

A particular problem is posed by the synchronization of time variables, which are incremented continuously by all animated applet; thus, a smooth, symmetric synchronization is difficult to achieve between two animated applets. Moreover, in the case of the synchronization of a time variable with a directly manipulated parameter, it is preferable to establish an asymmetric master-slave synchronization, where the directly manipulated parameter acts as the master. We solved these problematic situations by stopping any animation as soon as its time variable is set to another value due to a synchronization. This strategy turned out to provide the most intuitive user experience among the tested approaches.

5 CONCLUSIONS AND FUTURE WORK

This work presents a particularly simple interface for the specification of synchronizations of user interactions with multiple parametrized graphics and a prototypical implementation in a Java applet. There are many applications for this kind of synchronization, which will become more important as interactive illustrations become more popular in the web and in other electronic documents.

The proposed interface is only a small part of any complete framework for interactive illustrations; however, it can contribute significantly to the success and usability of any such framework.

Future work consists in providing a collection of basic widgets, e.g., sliders, dials, 2D and 3D point selectors, virtual trackballs, etc., and to research the specification and applications of remote synchronizations of parametrized graphics.

REFERENCES

Jmol (2008). Jmol Interactive Scripting Documentation. Retrieved July 24, 2008, from http://chemapps.stolaf.edu/jmol/docs/?ver=11.6#sync.

Kraus, M. (2002). Parametrized Graphics in Mathematica. In *Proceedings of the 2002 World Multiconference on Systemics, Cybernetics, and Informatics (SCI 2002)*, volume XVI, pages 163–168.

Kraus, M. (2008). From Parametrized Graphics to Interactive Illustrations. In Borwein, J. M., Rocha, E. A. M., and Rodrigues, J. F., editors, *Communicating Mathematics in the Digital Era*. A.K. Peters, Ltd.

Meckler, A. (1997). Java and Inter-Applet Communication. *Dr. Dobb’s Journal*.

ModernCA (2008). Modern Cellular Automata Author Documentation. Retrieved July 24, 2008, from http://www.collidoscope.com/modernca/author/ documentation.html.

Wolfram, S. (2003). *Mathematica: A System for Doing Mathematics by Computer*. Wolfram Media, 5th edition.

Wolfram Research (2008). Wolfram Demonstrations Project. Retrieved July 24, 2008, from http://demonstrations.wolfram.com/.