3D ANIMATION STREAMING

Martin Hash, Robert Hiromoto, Scott Harrison
University of Idaho, Dept. of Computer Science

Keywords: 3D Character Animation, Internet streaming.

Abstract: In applications where it is imperative that an event occur at a specified time but the data loading requirements cannot be met before the event trigger, like animation distributed over the Internet, it seems appropriate to send the most important data first so that the user at least receives the gist of the information before the application moves on. We offer a framework of graceful degradation of 3D character animation, such that the story narrative is maintained even if data transmission rates are erratic.

1 INTRODUCTION

Delivering data intended to be displayed as a continuous animation over a constricted transmission channel like the Internet may cause hesitation, which usually degrades the visual narrative. We suggest ways to maintain temporal integrity without perceived loss of information. This is accomplished by predictive preloading the data, and the strategic elimination of low-priority objects. We use heuristics that categorize data by desirability. Algorithmic liturgies then rank objects by importance to determine a precedence list (Hash, 2004).

2 GRACEFUL DEGRADATION

The richness of a 3D rendered scene can be measured by how populated it is and how detailed the objects are. The ideal, of course, is to completely deliver the creator's intended vision, but if that is not possible due to a technical consideration such as constricted bandwidth, then our goal is to degrade gracefully, which means that the parts of the scene that are eliminated are not important to the viewer's perception of the story. An example might be two characters conversing in a room that contains furniture and windows to the outside. Gracefully degrading the scene, perhaps the outside scenery could be dropped first, followed by the furniture in the room, and finally the room itself, leaving only the two characters and their dialog to maintain the narrative.

2.1 Precedence

The two values that determine how much of a scene is visible to the viewer are precedence and load time. Precedence can be determined by performing an examination of the data files that describe the animation. Due to the vagaries of Internet transmission, load time can only be roughly estimated by dividing the file size by the predicted data transfer rate. The goal is to minimize the hesitation, or lag, between scenes. There have been extensive studies of how much lag a typical viewer can be expected to tolerate. Viewers become highly critical if the lag reaches a 200-300 ms threshold (DeFanti, 1999).

2.2 Trigger

3D character animation is linearly temporal, meaning it is driven by a clock and events are triggered at prespecified times. Since most animations have varying data load requirements over time, it becomes necessary to smooth out the demand. The temporal trigger of 3D character animation is the beginning of a scene. Once the trigger has occurred, only the data that has been loaded will be displayed, and data needed for the next scene will begin loading. File size is the overriding metric, both in respect to transmission bandwidth and storage size. The list of objects is

Hash M., Hiromoto R. and Harrison S. (2006).
3D ANIMATION STREAMING.
In *Proceedings of WEBIST 2006 - Second International Conference on Web Information Systems and Technologies - Internet Technology / Web Interface and Applications*, pages 457-460
DOI: 10.5220/0001237004570460

ordered by the precedence determined using the concept of importance.

2.3 Definitions

A model is the visual manifestation of a 3D object. Actions specify movement of the models. Models can be further detailed with pictures called images. The choreography describes the temporal relationship of these objects to the story and each other, plus includes ancillary objects, such as cameras and lights. A project contains all of the choreographies, and displays them in linear order to create the story narrative. The most important part of the story is the sound, followed by the models (without images on them), and finally the images that go on the models. A first pass of the project file is performed to identify the models, the actions, the file sizes, and the temporal requirements within the choreography, so that importance can be assigned.

2.4 Object Importance

A model is comprised of many components, such as file size, number of matrices, and actions that are used, but a detailed inspection of over one hundred models indicates that only a few shared components actually indicate importance. After implementing a simple method of parsing files and counting a variety of components, we found the determination of model importance needs to incorporate these notions:

- Models with many actions have increased importance.
- The number of occurrences of a model in the scene increases its importance.
- Images on a model increase its importance.

Again, through inspection, actions had very few components that contributed to importance. Determining action importance needs to incorporate the following elements:

- The number of times an action is used increases its importance.
- Non-reusable actions, called chor (choreography) actions, have increased importance because they contain the default placement of an object in a scene.

Images used on models are subjugate to the model's precedence, meaning images from higher ranking models are all loaded before the next ranking model's images begin loading. Other than that, images precedence is determined by what is most visible on a model. Determining image importance should include:

- "Color" images have more importance than any other type.
- Images used multiple times have more importance.

2.4.1 Weights

Choreographies are parsed to obtain every loadable object and its relationship to other objects. Using the importance criteria above, we describe heuristics that determine precedence ranking among the objects.

The primary sound is identified as the one with the largest file size, and since it is preeminent, it is always loaded first and has no numerical weight. However, ancillary sounds (ranked by decreasing file size) may be outranked by high-ranking models and so must have some weight assigned to them. We will describe the model weight determining heuristic first.

Models should be loaded by rank, where rank is determined by the sum of the weights of the components of the model. As a simple implementation, the components that comprise the weights are given unit values. For example, every action used on a model increases its weight by 1, but if a model is used more than once, it only gets 1 additional unit (no matter how many times it is used), and if the model has images on it, the weight is also incremented by only 1 (regardless of the number of images). All the models in the scene are weighted in this manner and ranked in descending order.

Actions should also be loaded by rank, where rank is determined by the sum of the weights assigned via our approach. If an action is used more than once, it should have more weight, so one unit is added. The action that places the model in the scene, called a *chor*, seems to be very important in practical implementation, so to distinguish a chor action as the most important action, it receives a weight of three, which is one more than a multiply used action.

Images should be loaded last, by rank and in model order. After our inspection there seemed to be no contributory component of images that affected weighting other than the logical deduction that an image used more than once was more efficient as to load time.

Table 1 summarizes our deduced loading order and weights of the models and actions contained in five example choreographies.

Table 1: Weighting.

Object	Weight	Order
Choreography		1st
Sound		
Largest		2nd
Ancillary	10	
Multiple use	+1	
Model		3rd
Actions	1 each	
Chor Actions	3 each	
Multiple use	+1	
Image	+1	
Action		4th
Chor Action	3	
Multiple use	+1	
Image		5th
Multiple use	+1	

2.5 Experiments

In our example choreographies, models essential to the story indeed had the most weight. Optimal weightings are difficult to define because of their coarse granularity; however, this is acceptable because in our cases of two similarly weighted models, no differentiation was possible because both models were equally important to the story narrative.

No unjustified weightings occurred in our examples but pathological cases could easily be imagined. For example, a highly articulated model with lots of individual actions could outweigh an essential model, and it becomes difficult to distinguish essential models when there are many similarly weighted models to choose from.

2.6 Precedence List

From the heuristics described above, the Precedence list like that shown in Table 2 was created for each choreography. “Size” is in bytes, and “Play Time” is in milliseconds. The values without units are simple counts described by their respective headings.

3 PRELOAD TIME

Preload time is the loading that occurs before the playback begins. Preload time is determined by accumulating sequential, non-zero choreography wait times, where:

$$wait_n = \sum_{i=1}^{i=n} load_i - \sum_{i=0}^{i=n-1}(play_i + wait_i)$$

Where $play_0 = 0$, $wait_0 = 0$, *and* n *=choreography.*

Table 2: Sample precedence list (truncated).

Choreography	Size	Play Time
Frog and Butterfly*	**5000**	**20000**

Sounds	Size	Uses
Power of Juju	**169937**	1

Model	Images	Size	Uses	Actions	Rank
Tak	274	**207462**	1	19	27

Actions	Size	Uses	Weight	Rank
chor action	**1172**	1	3	3
show hand bones	**1964**	2	1	2
tongue roll	**41**	1	1	1
tongue curl	**1915**	1	1	1
oval eyes	**327**	1	1	1
hair control	**61458**	1	1	1
feather	**3583**	1	1	1

*Only bold-faced items are actually included in the project as part of the Precedence list.

The largest accumulation is the preload time. For example, in Table 3a, the sum of the wait times for the “Toys” and “Frog” choreographies is less than the wait time for the “Juju” choreography, therefore the wait time for the “Juju” choreography is the preload time. Table 3b is an example of how a simple reordering of the choreography play sequence results in a large variation in preload time.

Table 3b is an example of how a simple reordering of the choreography play sequence results in a large variation in preload time.

Table 3: Example preload times: a), b).

Name	Size (KB)	Play (S)	Load (S)	Wait (S)*
Toys	1038.8	9.4	6.93	6.93
Frog	2408.3	20.0	16.06	6.66
Dance	885.4	5.8	5.90	0
Juju	13922.3	286.0	92.85	72.95
Drive	1279.1	87.0	8.53	0

Preload Time a): 72.95 S

*Effective Internet transfer rate of 1.5 Mbits/second

Name	Size (KB)	Play (S)	Load (S)	Wait (S)*
Toys	1038.8	9.4	6.93	6.93
Juju	13922.3	286.0	92.85	83.45
Dance	885.4	5.8	5.90	0
Drive	1279.1	87.0	8.53	0
Frog	2408.3	20.0	16.06	0

Preload Time b): 90.38 S

In the case where there is not adequate load time available, which could occur due to a lowering of the effective Internet transmission rate during play time, the Precedence list determines what is truncated. Another important consideration that is taken into account by the preload calculation is that in later choreographies many of the objects are already available: these objects are assumed loaded, and will not contribute to the preload time calculation.

4 CONCLUSION

Though our evaluation is equally applicable to all kinds of networks, the constricted transmission bandwidth of the Internet most notably benefits from the concepts of preloading and Precedence lists. As technology improves, transmission bandwidth will increase but it has been our experience that the demands placed on the medium by the viewer’s expectations will similarly increase, so these strategies will remain relevant.

REFERENCES

Hash, M., 2004. “Preloading and Precedence Lists For Smooth Playback Of 3D Character Animation On The Internet”, Quality of Service in Heterogeneous Wired/Wireless Networks IEEE Qshine.

DeFanti, T., Stevens R., 1999. The Grid: Blueprint for a New Computing Infrastructure, edited by Ian Foster and Carl Kesselman, Morgan Kaufmann Publishers, Inc. 131-155.