

TEXAS A&M UNIVERSITY LIBRARY

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

OBJECT ORIENTED FRAMEWORK FOR
CHARACTER ANIMATION
AND DESIGN

A Senior Honors Thesis

By

ERIN DEVOY

Submitted to the Office of Honors Programs
& Academic Scholarships
Texas A&M University
in partial fulfillment of the requirements of the

UNIVERSITY UNDERGRADUATE
RESEARCH FELLOW

April 2002

Group: Computer Science

OBJECT ORIENTED FRAMEWORK FOR
CHARACTER ANIMATION
AND DESIGN

A Senior Honors Thesis

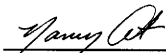
By

ERIN DEVOY

Submitted to the Office of Honors Programs
& Academic Scholarships
Texas A&M University
in partial fulfillment of the requirements of the

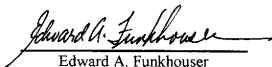
UNIVERSITY UNDERGRADUATE
RESEARCH FELLOW

Approved as to Style and content by:



Nancy Amato

(Fellows Advisor)



Edward A. Funkhouser

(Executive Director)

April 2002

Group: Computer Science

ABSTRACT

Object Oriented
Character Animation and
Design. (April 2002)

Erin Devoy
Department of Computer Science
Texas A&M University

Fellows Advisor: Dr. Nancy Amato
Department of Computer Science

Three-dimensional computer character animation is becoming more and more prevalent in the entertainment industry. As the demand grows, so does the need for tools that allow animators to create animations quickly. In this thesis, we present a framework for animating with the goals of minimizing the time to train new animators, the time to develop new characters, and the time to make a finished animation. We propose a system that is analogous to object oriented paradigms in software development – namely reuse and encapsulation. We then demonstrate the feasibility of the model by describing the prototype developed during the course of this research.

ACKNOWLEDGEMENTS

I would like to extend my gratitude to Daniel Caruso and his advisor Dr. John Keyser for collaboration on code to load and display three dimensional models.

I am also grateful to Jennifer Walter who served as interim advisor when Dr. Amato was away from the university on business.

Finally, I would like to thank the PARASOL Lab's Robotics Research Group for sharing their insights, knowledge, and work. They have been an invaluable source of inspiration and ideas.

TABLE OF CONTENTS

ABSTRACT	iii
ACKNOWLEDGEMENTS	iv
TABLE OF CONTENTS	v
LIST OF FIGURES	vi
INTRODUCTION	1
PROBLEM	3
PROPOSED SOLUTION	5
IMPLEMENTATION OF PROTOTYPE	7
Overview	7
Underlying Technology.....	9
Adaptive Animation.....	11
SUMMARY AND CONCLUSIONS	15
REFERENCES	16
VITA	17

LIST OF FIGURES

FIGURE	Page
1 Model Parameters in User Interface.....	7
2 User Interface Controls for Animation.....	8
3 User Interface Controls for Display.....	10
4 Basic Forces in Flight.....	12

INTRODUCTION

The usage and uses of three-dimensional animated characters are growing every day. Animated characters permeate the entertainment sector, from the characters in computer games to starring in their own feature films. As this market continues to grow, the demand for production drives the need for new tools to address the weaknesses in current animation processes.

Today, many animators are forced to build a new model from scratch for each new character. After this structure is in place, they have to provide a framework for animating the character. Finally, they create their animation, slaving over each movement made by a character, painstakingly positioning it key-frame by key-frame.

To alleviate the time and effort involved, quite a bit of work has been done on automating animation. Many researchers focus their efforts on humanoid characters. In [3] researchers from Stanford University and the Toshiba Corporation use a motion planning approach to allow a human actor to complete complex tasks involving grasping and manipulating objects. Jordan

This thesis follows the style of the IEEE Transactions on Robotics and Automation.

and Hodgins combined motion capture and dynamic simulation to animate humanoid characters in [6]. Hodgins also collaborated on work with Wooten in [2] to create humans with complex athletic behaviors.

Other researchers have tackled different types of characters.

Terzopoulos, Tu, and Grzeszczuk created fish that have physically based movements and demonstrate complex behaviors [5]. Terzopolos, Faloutsos, and van de Panne collaborated on a system to automatically generate control structures and animations for classical anthropomorphic subjects (e.g. teapots) [1].

Our discussion of the problem will begin with a more detailed analysis of the problem, followed by an outline of a system that removes or reduces the impediments of the current process. Then, we describe the prototype that was constructed to demonstrate the feasibility of the proposed system. Finally, we close with a discussion of the results and what conclusions can be drawn from them.

PROBLEM

There are several aspects that limit productivity on an animation project, which we wish to eliminate:

- Steep software learning curve: Most commercial modeling and animation software has a steep learning curve. Some programs take months of training and years of practical use to master. By making the system easier to use, we can increase the number of competent animators and decrease the cost of training.
- Little or no geometry reuse: Most three-dimensional models are not built with reusability in mind. Even if an animator wishes to build a reusable model, there is little software support to make this feasible. At this point, many production houses turn to proprietary plug-ins.
- Tedious animation: Animation is currently a tradeoff between automation and control. At one extreme is key framing, where the animator creates a timed sequence of poses, relying on the computer to interpolate between them. This technique is tedious, but allows for complete control over the final animation. At the other extreme is physically based simulation. Using the forces key to the animation and control algorithms, the characters are animated automatically. Unfortunately, these algorithms are difficult to

define and the systems are difficult to control. Ideally, we would like to create a system that can give us the best of both worlds – automation with intuitive control.

- Inflexibility of design: With both physically based and key framed techniques, any significant change in the character will make previous animations useless due to changes in the physical properties of the character.

We hope to create a solution that will address all of these issues, resulting in a faster and easier way to model animations. In the following section, we will outline the system we propose to confront these issues.

PROPOSED SOLUTION

To combat these problems, we propose an approach analogous to the Object Oriented software development paradigms. At the core of our system is a database of characters with geometry animation structure in place. These characters are designed to have properties that allow them to be customized to a specific application. For example, a bird model could have adjustable wing and mass proportions to change its appearance and style of flight from that of an albatross to a hummingbird. Each member of the database also has several possible animations that are designed to adapt to changes in the character's properties. To make sure that the animator has complete control over the system, the animations should be designed to adapt to user input.

This system parallels Object Oriented Design in two ways. Data (geometry) and function (animation) are encapsulated into a single logical unit. Each type of object is extensible (Alteration of Parameters) to allow for customization and reuse.

There are several benefits to this system. Because a beginner works from an existing model, they can be immediately productive. Both new and experienced users can reap the benefits of geometry reuse. Less time modeling

is a direct gain in productivity. Because each character in the database has automated animations associated with it, animation time is reduced as well.

Not only are animation and modeling cycles reduced, because we use the template character and all children of the template character will be able to follow similar animations, it is possible to do all but the final tweaking of the animation in parallel with the modeling process. The adaptability of the animation also allows the character's geometrical design to be more fluid throughout the animation process.

The success of this approach hinges on our ability to adequately describe motion and geometry so that it is flexible and adaptive.

IMPLEMENTATION OF PROTOTYPE

Overview

To demonstrate the feasibility of such a system, we implemented a prototype with the basic functions listed above. Because an exhaustive database of characters was an unreasonable goal for the time period and man-hours available for this project, a single character, a bird, was created. This bird has two adjustable properties: mass and wingspan (see Fig. 1 for more information).

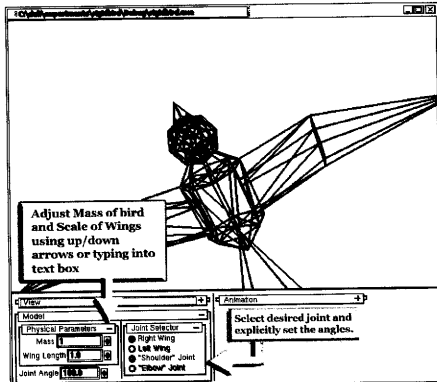


Figure 1: Model Parameters in User Interface

The basic animation loop consists of a series of key frames. When the mass is altered, the speed of the animation is also altered. The change in speed is driven by the need to oppose the stronger gravitational force. Altering the wingspan of the bird will generate more lift per wing stroke, and thus the period of the wing cycle will increase. The animator can refine the rough animation by specifying additional key frames. They can chose to either have the animation change applied to all cycles, or only the current

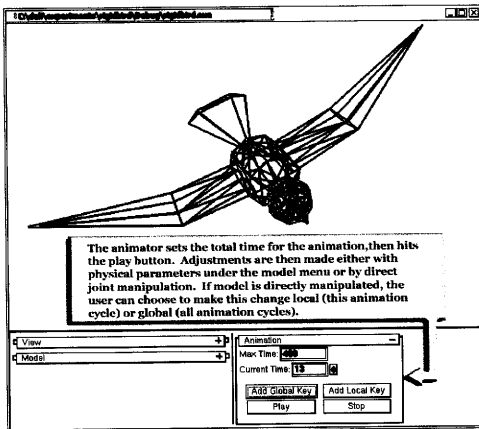


Figure 2: User interface controls for Animation

one.¹ The actual user interface controls are explained in Figure 2.

Underlying Technology

The interface for the prototype is built upon the OpenGL, GLUT, and GLUI libraries. These layers of abstraction allowed us to ignore system dependent windowing issues.

The bird model is a 3-D triangular mesh. The wings of the bird are currently the only animatable parts. The vertices of each wing are bound to one of three bones (4x4 transformation matrices).

The user interface includes settings to display the bones and the matrices bound to them. It also contains a checkbox that can be used to switch between wireframe and flat shaded modes. The wireframe mode is less visually appealing (wireframe is the mode that appears in Figures 1-3), but it is less

¹ Note: As of the release of this document, there are still programming issues with the animation system. If time permits, an updated document will be submitted when the program is in full working order. However, it is unlikely that any design changes will have to be implemented, so the methodology described in this document should be consistent with the final release of the program.

computationally intensive. This makes it useful for lower end systems. There is also a checkbox to switch in and out of vertex array mode. When in vertex array mode, the vertices are submitted to the OpenGL card as a single chunk of memory, offering significant performance improvements. However, we have found that on some graphics cards (notably the Oxygen GVX1), the vertex array implementation is unpredictable. On systems where this is the case, the user can opt for the slower, but more reliable per vertex submission method. See Figure 3 for layout and further information.

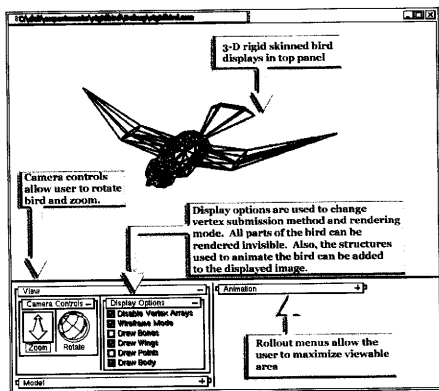


Figure 3: User Interface Controls for Display

Adaptive Animation

As discussed in the “Proposed Solution” section, the power of this method is based on the adaptability of the animation system. The animation module for this prototype is relatively simple. Because it is certainly not a given that a model will be physically accurate, using a physically accurate flight model is not a feasible approach. However, to make the animation “feel” natural, we need to mirror the physical world. To do this, we take the basic forces in flight and insure that our estimated forces increase in a similar manner. Thus, a linear force will remain linear, and one that increases on the square will follow suit.

The basic forces involved are shown in Figure 4: Basic Forces in Flight.

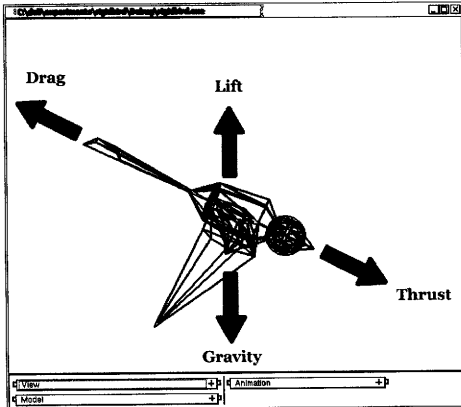


Figure 4: Basic Forces in Flight

The simplest force to calculate is Gravity, which follows the familiar $f = ma$ formula. The changes in the mass of the bird will affect this force. In order to maintain equilibrium, we must increase the opposing force, lift. The calculation of lift off of the wing of a bird can get quite complicated. Unlike the wing of an airplane, the wing of a bird is in constant motion. As an approximation, we will assume that the total lift of the cycle can be calculated similarly to lift off of a stationary wing:

$$L = C_l \times \frac{1}{2} \times \rho \times V^2 \times A$$

(Equation 1)

where : L = lift, C_l = Lift Coefficient, ρ = air density, V = air velocity, and A = wing area.

After the animation system is debugged, we will determine a reasonable constant, C_l , which will remain fixed after it is determined. Note that if the ability to further alter the shape of the body were added, C_l would vary with these deformations. We assume that air density is constant (1). Air velocity is stored and is dynamically updated based on forward thrust. The wing area changes when the wing scale is altered by the user.

To determine drag, we use the formula:

$$D_r = \frac{1}{2} \times D \times A \times V^2$$

(Equation 2)

where D is drag, A is frontal surface area, and V is velocity of the object. Since A is constant, V primarily determines drag.

Thrust is generated by the net forward lift generated in the wing cycle. Assuming that the wing is facing during 1/3 of the cycle, we can estimate the forward thrust to be 1/3 lift. This is a very sketchy approximation. Ideally, we hope to integrate a more physically based approximation of thrust.

Currently, equilibrium is only maintained in the up/down direction. This is done by increasing the number of wing beats, thus increasing the total lift (remember, the lift calculation above is for the average lift per cycle; To get the

total lift, we multiply by the number of cycles per second). I cannot stress enough that this is intended to be a useful approximation, not a strictly correct simulation. As such, inaccuracies will appear. Some fine tuning will be necessary, but this is a one time task and will not increase the overall time to complete a new animation after the initial setup is complete.

SUMMARY AND CONCLUSIONS

We have proposed and implemented a prototype for an alternate paradigm for character animation, loosely based on Object Oriented software development methodologies.

Although the prototype developed for this research is far from a production system, it demonstrates that a basic implementation of the proposed system is possible. To fully determine if this system is superior to current animation processes, future work is needed. Specifically, the system would need to be developed to the point where meaningful usability tests could be run. Implementing steering behavior and a more flexible model are hard prerequisites to such usability tests. The ability to form high level behaviors from lower level behavior is also vital to the system.

REFERENCES

- [1] Faloutsos, P., Van de Panne, M., Terzopoulos, D. "Dynamic Free-Form Deformations for Animation Synthesis." *IEEE Transactions on Visualization and Computer Graphics* vol. 3 No 3. July-September 1997.
- [2] Hodgins, J. K., Wooten, W. L., 1998. "Animating Human Athletes." *Robotics Research: The Eighth International Symposium* . Y. Shirai and S. Hirose, eds. Springer-Verlag, pp. 356-367.
- [3] Koga, Y., Kondo, K., Kuffner, J. and Latombe, J.C. "Planning Motions with Intentions." *Proceedings of SIGGRAPH'94*, pp. 395-408, 1995.
- [4] Metoyer, R. A., Hodgins, J. K., "Animating Athletic Motion Planning By Example." *Proceedings of Graphics Interface 2000*, Montreal, Quebec, Canada, May 15-17, pp. 61-68.
- [5] Terzopoulos, D., Tu, X., and Grzeszczuk, R. "Artificial fishes: Autonomous locomotion, perception, behavior, and learning in a simulated physical world." *Artificial Life*, 1, 4, December, 1994, 327-351.
- [6] Zordan, V. B., Hodgins, J. K. 1999. "Tracking and Modifying Upper-body Human Motion Data with Dynamic Simulation." *Computer Animation and Simulation '99, Eurographics Animation Workshop*, Sept. 1999, N. Magnenat-Thalmann and D. Thalmann, eds., Springer-Verlag, pp. 13-22.

Consulted Works

- Pennycuik, C. J. Bird Flight Performance : A Practical Calculation Manual. Oxford University Press, 1989

VITA

Erin Devoy was born in Baton Rouge, LA in 1980. She attended the Louisiana School for Math, Science and the Arts where she earned her high school diploma. After LSMSA, she moved on to Texas A&M University where she received the Teagle Foundation award, National Merit Scholarship, and President's endowed scholarship. She interned with Tivoli Systems Incorporated, a (previous) division of IBM. She expects to graduate in May 2002 with a BS in Computer Science. After graduation, she hopes to continue her education by earning an advanced degree in Computer Science. She will be interning this summer with the Extreme Blue Program at IBM's Almaden research facility.

Correspondence can be sent to the following address:

1241 Thoreau Dr., Baton Rouge, LA, 70808.

2172874✓

