Proceedings of the 1993 Winter Simulation Conference
G.W. Evans, M. Mollaghasemi, E.C. Russell, W.E. Biles (eds.)

PROOF ANIMATION: BETTER ANIMATION FOR YOUR SIMULATION

Nancy J. Earle
James O. Henriksen

Wolverine Software Corporation
4115 Annandale Road
Annandale, Virginia 22003

ABSTRACT

The Proof Animation™ family of animation software is designed to meet the animation needs for a vast array of applications. This product family runs on readily available, inexpensive PC hardware. Some of the overall features are general purpose architecture, ASCII file-driven, vector-based geometry with the ability to zoom in or out while maintaining crisp clear animations, post-processing for maximum performance, CAD-type drawing tools for ease of creating animations, and a unique presentation mode used for displaying the results of the study. Proof Animation is not tied to one specific simulation language. Because of its open architecture, Proof Animation can serve as the animation tool for models written in a wide variety of simulation and programming languages. Proof Animation's user interface is menu-based and easily navigated using either a mouse or keyboard. Its superior performance assures smooth, realistic motion whether the animations depict simple systems or complex systems with many moving parts.

1 INTRODUCTION

Animation has become an integral part in simulation projects from beginning to end. There are many aspects to consider when choosing animation software for a project:

Is the animation software capable of easily representing the system? Some software is geared to a specific application, such as manufacturing or health care. Application specific animation software is useful if only one type of system will ever be studied.

Will the animation software work easily with my simulation software? Since the onset of a project is *not* the ideal time to learn new modeling software, it is important that the animation software work with the existing simulation package.

Can the software handle the size or complexity of the system? The animation software must have the capacity to handle a full-scale, industrial project. Size limitations of animation applications should not be an issue.

Is the software affordable? A tool to help determine feasibility of a project should not exceed the project budget.

During the development of the model/animation, the user must be concerned with the animation software's ease of use. How easy is it to draw the animation layout? Can the layout be import from a CAD drawing? Does the animation software allow the passing of information to the simulation model, therefore lessening the modeling data input? How easy is it to define and modify the characteristics of moving objects? Can the animation be easily used to determine whether the model correctly represents the system?

As the project draws to a close, issues of portability and effectiveness of presenting results become increasingly important. An animation can effectively show the results of a simulation study to a general audience with varying technical backgrounds, but getting the animation to the audience is the key issue. Can the animation easily be taken to meeting room or to a customer's site, or does it require special hardware that they may not have? Can a professional looking presentation be assembled consisting of a collection of slides and animation clips to summarize the results of the simulation/animation project? Software should be chosen that gives a positive response to these types of questions.

The Proof Animation family of software products can meet the challenges posed in the above questions. In the following sections we describe the family of products, discuss their underlying design philosophy, describe their organization, give an overview of how they are used, and describe how features can be applied to specific applications.

2 THE PROOF ANIMATION FAMILY

The following products comprise the Proof Animation family:

- PA — The basic animator. Requires a 286 or better CPU, a math coprocessor, and an EGA/VGA video card. DOS 640K memory limitations apply.
- SPA — The student version of Proof Animation. Included with the *Using Proof Animation* text. Size and playing time limitations are imposed; otherwise identical to PA.
- PP — Proof Professional. Requires a 386 or better CPU. Uses 32-bit DOS-extender technology to break the 640K barrier. 1024 X 768 high-resolution version, PP10, is included at no additional cost.
- PPRUN — Run-time Proof Professional. Same hardware requirements as PP. Runs developed animations or presentations. No drawing capabilities. Provides a lower cost way to run different scenarios with a fixed layout file prepared using PP or PA.
- PADEMO — The Demo version of PA. Can only be used to run animations prepared under a licensed copy of PA containing a Demo-Maker add-on. Copies of PADEMO can be reproduced and distributed free of charge.
- PPDEMO — The Demo version of PP. Same features as PADEMO, but can handle much larger animations.
- DXF2PA — An add-on utility for converting industry-standard .DXF CAD files into Proof Animation layout (.LAY) files.
- PA2DXF — An add-on utility for converting Proof Animation layout files into .DXF files.

3 THE BASIC DESIGN

3.1 General-Purpose

Proof Animation was designed to be general-purpose in many ways. First, it is independent of any particular simulation or programming language. While Proof Animation was built to work easily with Wolverine's GPSS/H simulation language, we are pleased to have provided animation software to users who develop models in other popular languages. The only requirement for compatibility, is that the language used be capable of writing ASCII text files. Most animation software from other vendors is tightly coupled to their simulation software. The claimed advantage of this approach is that because the animator has direct, automatic access to the events which occur in a simulation, development of the animation is simplified. This may be true for small, simple animations in which there is a one-to-one relationship between simulation events and animation events, for example, a box moving from point A to point B. However, in more complex simulations, the modeler may actually have to alter the approach used in modeling the simulation to achieve the desired *look* for the animation.

The second way in which Proof Animation is general purpose is in the design of its command set. Commands such as CREATE, DESTROY, PLACE, MOVE, SET SPEED, and SET COLOR are easily learned and easily used. They provide exactly the kind of flexibility necessary to easily be integrated with the simulation model logic.

Finally, Proof Animation is not tied to a specific application. There are features that make it an ideal choice for animation of systems like computer networks, health care, transportation, manufacturing, etc.

Purveyors of tightly coupled simulation/animation software claim that their approach is the *only* way to add animation to a simulation; it is not. A mix-and-match strategy for acquisition of software allows you to select optimal functionality and prices. Sole sources of any goods or services tend to be expensive.

3.2 PC Platform, DOS Application

The Proof Animation family was designed to use widely available, inexpensive hardware, in order to maximize portability. PA, the basic version of Proof Animation, requires only a 286 or better CPU, a math coprocessor, and an EGA/VGA-compatible video card. Configurations of this type are very widely available, and it is good bet that this hardware will be at a customer's site. High-end 486 and higher based PCs are ideal platforms for running Proof Professional (PP), and are

available for reasonable prices. The choice of DOS as a host operating system was made because of its enormous installed base and its simplicity of operation (no multi-tasking). The Proof Animation family can also be launched as a full-screen application under both Windows 3.1 and OS/2 Release 2.

3.3 ASCII File-Driven

Proof Animation requires two ASCII input files to run an animation: a layout (.LAY) file and a trace (.ATF) file. The layout file describes the geometric details of the background over which objects move, provides geometric definitions and properties for such moving objects, and provides logical paths along which these objects move.

Ordinarily, layout files are produced at least in part by using Proof Animation's CAD drawing tools; however, the layout file command set specifications are published. This makes it possible to write programs to generate layout files. For example, some users have written front ends for their simulation models which allow different system design parameters to be specified for each run. Based on these parameters, different geometric configurations are written and incorporated into a layout file. The new layout is visible when Proof Animation is invoked.

The trace file contains a time-ordered sequence of commands such as CREATE, DESTROY, PLACE, MOVE, and many more. It supplies Proof Animation the information on when, where, and what to create, destroy, place, move, etc. Trace files are free-format, and the syntax of the commands is designed for ease of generation. Any language which can produce formatted ASCII output can easily write a trace file.

3.4 Vector-Based Geometry

In the Proof Animation family, all layout geometry is stored using vector form. Vector-based descriptions are automatically mapped into the screen's pixels to build the image. One of the advantages of this approach is that layouts can be much larger than a single screen. With the ability to zoom in and out and pan, larger layouts are easily navigated to show the overall layout or zoom in to whatever level of detail is necessary. Vector based geometry also provides the ability to have moving objects *rotate* around corners instead of the sliding effect to which other animation packages are limited.

Instead of vector form, many animation packages use a pixel-oriented approach for drawing. The advantage of this approach is purely aesthetic. With the ability to manipulate individual pixels, one can produce detailed, arty images. However, this level of detail is time consuming to draw and can often be lost because of the scale at which the animations are viewed. Some other disadvantages of pixel-orientation are: (1) pixel-oriented images cannot be rotated; (2) layouts are often confined to single-screen images. Some animators offer multi-screen operation; however, the individual screens are disjoint and independent, unlike Proof Animation's single, continuous canvas; (3) Zooming in on pixel images magnifies the "jaggies" inherent in all such images. When a zoom in is performed in Proof Animation, the vector-based image maintains its crisp and clear appearance. Lines continue to look like lines. In pixel- based animation packages, the "jaggies" effect makes a line look like a stairway, if a zoom in is performed.

3.5 Post-Processor Operation

Proof Animation is a post-processing animation package. That means it runs *after* the simulation has executed. Both the layout and trace files must exist before invoking Proof Animation. They cannot be written and read concurrently. The post-processing approach offers two great advantages. First, PC hardware resources are not shared between the simulation and the animation. This leaves the entire CPU for running the animation. Second, it provides the ability to jump back and forth in time during the animation playback and the ability to speed up and slow down viewing speed. These features make it easy to identify and investigate unusual system behavior.

3.6 Importing .DXF and .PCX Files

Sometimes a .DXF formatted CAD drawing already exists for a system which is to be animated. In this case, the effort of redrawing an entire layout can be avoided. Credibility with end-users of the animation is enhanced because they are accustomed to viewing the CAD drawing of the system. Proof Animation's optional add-on DXF2PA and PA2DXF utilities provide the capability of converting industry-standard .DXF CAD files into Proof Animation layout files, *and vice versa.*

In addition to being able to import and export CAD files, Proof Animation can read and write bit-mapped screen images. It is very straightforward to save Proof Animation screen images in industry-standard .PCX files and incorporate them into presentations as *slides*. Third party packages for producing very high-quality charts, graphs, and slides can also be used. There are many such packages available, and virtually all of them can export images as industry-standard .PCX files.

3.7 Designed for Maximum Interoperability

Proof Animation was designed for maximizing *interoperability* with other software. Because of prices, compatibility, and features, mix-and-match software has become a reality. Clearly, no single vendor can hope to be the best possible source of software to fulfill *all* the requirements of an animation: simulation, animation, CAD, and presentation graphics. Proof Animation's use of an open, published, flexible, file-driven architecture, and its ability to read and write industry-standard file formats enable users to easily implement a mix-and-match approach. Proof Animation can be used with CAD, graphics, and a wide choice of simulation software.

3.8 Performance

A great deal of emphasis has been placed on Proof Animation's performance while running an animation. High performance enables Proof Animation to achieve very smooth motion by updating or refreshing the screen 60-70 times per second. Other software can often sustain refresh rates of only 5-10 updates per second. The ultimate purpose of an animation is to achieve a realistic depiction of the system that is being studied. This allows the audience to easily gain confidence in the results of the simulation study. Objects which move smoothly across the screen are more realistic than those that do not. Everyone knows that forklifts are supposed to roll across the factory floor, not jump.

4 THE PROOF ANIMATION MODES:

Proof Animation is organized into seven menu-driven *modes*. Each mode is a collection of closely related functions. Switching among these functions is very easy. Usually, a single mouse click is all that is required. Switching among modes is also easily done, but it implies major changes of context. For example, running an animation and drawing a layout are vastly different activities. Each mode has one or two main menu bars at the top of the screen. Clicking on main menu items invokes the options for the lower level tools.

Proof Animation's modes are summarized as follows:

Run Mode is the mode in which animations are viewed. It provides menu tools for starting and stopping an animation, changing views, controlling viewing speed, jumping ahead and back in time, and more.

Debug Mode is a variant of run mode. It provides tools for stepping through an animation by individual events or time commands and examining the resulting movement. Information pertaining to an individual object can be obtained by clicking on the object with the mouse.

Draw Mode is used for creating the layout background. Tools are provided for drawing static elements such as lines, arcs, text, fills, etc. Dynamic elements such as messages, bars, and layout objects are defined in Draw Mode.

Class Mode is used for defining object classes. An object class serves as a template for creating both the dynamic objects that move around a layout and layout objects that generally remain stationary. The template determines an object's size and shape and other initial properties such as default speed and color. An animation will probably contain multiple object classes. For example, an animation of a shipping dock might contain classes that represent trucks, pallets, or staff.

Path Mode is used for defining fixed paths. A path is a perfect application for *guided*, directional object movement such as conveyors. The geometry or route of a path is easily defined by clicking on existing lines and arcs of a layout. Tools are also provided for defining path speeds, circularity, and accumulation status. Accumulating paths provide automatic queuing for objects which pile up at the end of the path.

Presentation Mode is used for running scripted presentations. Scripts can include static slides and snippets of animation, separated by special effect segues such as screen fades and dissolves.

Setup Mode is used for examining and altering infrequently changed configuration data. For example, the color palette can be customized or the mouse speed increased or decreased.

5 USING PROOF ANIMATION: AN OVERVIEW

5.1 Drawing a Layout

The first step in developing an animation is to draw a layout. Given a .DXF formatted CAD drawing of the system that will be animated, you can begin by importing the drawing into a Proof Animation layout file. This is done using the DXF2PA add-on utility. The entire CAD drawing or selected layers can be imported. If you do not have a CAD drawing or just like

to draw using a computer, the drawing tools provided in Draw Mode are easy to learn and use. Although it is mouse-oriented, Draw Mode also provides for keyboard input, so if you need to draw a line of a length 12.5 at exactly a 36 degree angle, you can enter these specifications *numerically*, and the geometry will appear on the screen. To help in drawing scaled, accurate layouts, a visible grid is turned on automatically when you enter Draw Mode. Another drawing aid is the Snap-to-Grid. This option is also *on* as the default setting. Snap-to-Grid limits the drawing of layout element from grid point to grid point, thus eliminating the chance of small gaps between the endpoints of "connected" lines.

5.2 Defining Object Classes

Once the background of the animation is drawn, the second step in developing an animation is usually to define one or more object classes. This is done in Class Mode. Objects and object classes are among the most important constructs in Proof Animation. An object class provides the geometric description, or shape of the individual objects that move throughout the animation. The class definition also includes the initial properties such as physical clearances, color, and an optional speed of the individual objects. Each animation will usually have a collection of object classes. For example, a traffic model might include object classes for automobiles, trucks, buses, campers, and motorcycles.

Although Proof Animation does not purport to implement a true "object-oriented" framework, it is meaningful to call each object an instance of a particular object class. Think of an object class as the template from which the individual objects are made. Expanding on the traffic model mentioned above, one could have cars that are different colors and moving different directions. Each of these cars is an individual object, based on the single geometric description of an automobile. There can be an arbitrary number of Automobile *objects* in the system at once, but there need be only *one* Automobile object class.

Motion and color-changing commands from the Proof Animation trace file operate on objects. Most layouts are drawn directly on the screen using Draw Mode. These background components cannot move or change color. If such changes are required, the appropriate components must be defined in an object class. Objects from that class can then be created and prepositioned in Draw Mode. The objects that are created and placed on the screen as part of the layout are called *layout objects*. They provide the capability to place objects into the layout while you are drawing the background. This enables scaling or positioning the object while having the background components visible as reference points. While the animation is running, these layout objects can be manipulated using trace file commands. For example, if an idle machine is shown as green and a busy machine as red, the machine could be defined as an object class. Objects from that class could be created and placed as part of the layout file, and color could be changed while the animation is executing based on the system status.

5.3 Defining Paths

The next step in developing an animation is to use Path Mode to define one or more paths - if guided motion is a part of your system. Proof Animation provides two kinds of motion: absolute and guided. Absolute motion, specified by the MOVE trace file command, causes an object to be moved between two arbitrary points A and B. Guided motion always occurs along a fixed route, called a path. The geometry of a path is defined by clicking on previously existing lines and/or arcs. The lines and arcs are first drawn using Draw Mode or imported from CAD. The path segments are then defined *on top* of them. A single line or arc can be part of one or more paths. Once defined, paths are saved as part of the layout file.

Using paths is very simple, because Proof Animation does all the work. The most commonly used trace file path command is PLACE [object] ON [path]. Once an object is placed on a path, it will follow that path until it visually comes to rest at the end of the path or until it is PLACEd elsewhere or DESTROYed. All objects traveling on the same path can be stopped simultaneously and resume movement at a later time. Paths provide outstanding animation power in response to a single trace file command.

Accumulating paths provide even greater power for animating paths on which queuing can take place. On accumulating paths, Proof Animation reflects physical reality by *visually* queuing objects when bottlenecks occur. This often makes a simulation model of the system much simpler to construct, because such queuing need not always be explicitly represented in the model code. A surprising number of systems contain some accumulation. This property can be used to represent certain types of conveyors, cars at a red light, supermarket checkout lanes, and more. Paths play an especially important part in transportation and material handling animations.

5.4 Producing a Trace File

Proof Animation trace files consist of very readable commands. Trace files are time ordered. That means the specific animation events take place between Time

commands. Consider the following portion of a trace file:

```
TIME 10.25
CREATE WIDGET 1
PLACE 1 AT 20 30.5
```

It is very easy to understand what is going to happen here. At time 10.25, an object with an id number of 1 is created with geometry and properties inherited from class WIDGET. This object will be placed on the screen at x=20 and y=30.5. It is very easy to produce simple trace files with any ASCII editor.

Actual trace files will be much more complex and inefficient to create by hand. Letting a simulation model generate the trace file is usually the *only* viable approach. In order to produce a trace file, output statements must be inserted into a simulation model, to produce the appropriately formatted commands. The Proof Animation trace file command set has been designed to be easily generated. Any language which can do formatted writes to an ASCII file is capable of building a trace file.

5.5 Building a Presentation

As an optional final step, a presentation can be built using Proof Animation. Presentation Mode lets users create scripted sequences consisting of slides, full animations, and/or segments selected from full animations. These presentation elements can be linked together using fades, dissolves, and other special effects, to produce a polished presentation. This is done by writing a simple ASCII presentation script file (.PSF). Complete presentations can be viewed without ever exiting Proof Animation. This eliminates the awkwardness of switching back and forth between overhead transparencies and computer displays during a presentation.

Presentations can be developed so that slides and animations appear on the screen for a defined amount of time. The viewer does not have to interact with the computer for the presentation to continue. Presentations can also be developed to continue once a key or mouse button is pressed, giving the viewer or presenter varied time to comment on what is currently on the screen.

The presentation developer can choose to highlight areas of interest in both space or time within the animation by using different views or sound to draw the viewer's attention to particular aspects of the animation. Presentations can incorporate selectable menus defined by the presentation developer. These menus can be set up by topic, giving the viewer or presenter complete control and flexibility of what to show.

6 FEATURES FOR POPULAR APPLICATIONS

There are many types of applications for which Proof Animation is perfectly suited. The more popular applications include manufacturing, transportation, and health care systems. Specific features of both layout and trace files make animating difficult modeling logic in these systems simple.

6.1 Manufacturing

Manufacturing has long been a classic application for simulation and animation. Proof Animation, although not solely developed for manufacturing type applications, has the needed features to develop realistic manufacturing animations. A few of these features are described below.

The ability to define accumulating paths is a must for manufacturing systems. This can be done easily in the layout file while building paths in Path Mode. Once a path is defined as accumulating, objects will queue until they are removed from the path. This is especially useful when animating conveyors or guided vehicles. Along the same lines, there is a SET SPEED trace file command that sets the path speed to zero. Issuing this command can stop all object movement on a particular path. To resume movement, another SET SPEED command with a speed value greater than zero must be issued. This technique can be used to illustrate conveyors that stop due to a breakdown or other system events.

Color and shape of an object can be changed using the trace file commands SET COLOR and SET CLASS, respectively. The SET COLOR command can be used to differentiate between busy, idle, or broken down objects. This combined with layout objects can be used to represent the states of machines. SET CLASS changes an object's geometry and properties from one object class to another. This can be used to show an assembly operation where the *look* of the moving objects change at each point in the assembly process.

Statistics, such as utilization or queue length can be shown using a bar graph. The size, location, and color of the bar is defined in the layout file. The fill level of the bar is changed dynamically using the SET BAR trace file command. This feature can be used to show instantaneous statistics, a level indicator of a tank filled with liquid, a temperature, or other dynamic variables.

6.2 Transportation

Transportation applications can range from determining the traffic light cycle at simple intersection to determining runway utilizations at a major airport. Many of the features listed above can also be used in animations of transportation systems.

Objects can be assigned an individual speed with the SET SPEED trace file command. This command is useful when animating any type of traffic where individual units have different speeds. Examples of this behavior are cars on a highway, trains on the same track, or planes taxiing down a runway. When used in conjunction with accumulating paths, additional benefits of this feature appear. If an object with a faster speed is placed on an accumulating path behind an object with a slower speed *and* it catches up to the slower object, the faster object takes on the slower speed until it is no longer blocked by the slower object. It is difficult to think of any automobile traffic situation where this *does not* apply.

Accumulating paths have a Lag Time property that is defined in the layout file. Lag Time is the number of animation time units that an object will hesitate before beginning movement *after* the object directly in front of it has begun moving. This causes an *inchworming* affect similar to cars queued at a traffic light that has just turned green. This can be used to represent driver's reaction time to the movement of the car directly ahead.

6.3 Health Care

A growing number of hospitals and health care facilities are using simulation and animation to study operational strategies. Popular areas of simulation/animation studies are emergency room utilization, operating room scheduling, and staffing requirements.

Proof Animation can handle the high volume of motion involved in large hospital animations while retaining smooth motion. Trace file commands like SET CLASS can change an object from resembling a person to resembling person in a wheel chair or on a gurney. SET BAR can be used to display the utilization of operating rooms, doctors, and nurses.

If a large hospital or hospital floor is being animated, unique views can be set up. A view of only the emergency room, or only an operating room can be defined by zooming and panning to the appropriate area. These views are named and saved as part of the layout file. The SET VIEW trace file command can be used to dynamically change the view of the animation to highlight a certain part of the hospital.

7 SUMMARY

Animation is a powerful addition to any simulation effort. An animation benefits the modeler in verification, validation, presentation of results, and also helps with the overall system design process.

Simulation and animation technology is constantly improving. Wolverine Software Corporation continues to contribute to this improvement by providing an innovative family of animation products. Proof Animation is a general purpose animator that boasts many important features. They are the ability to create presentations, an open architecture for compatibility with a variety of simulation software, CAD drawing tools, smooth motion, and powerful features to produce detailed animations of any type of system.

REFERENCES

Henriksen, J.O. and N.J. Earle. 1992. Proof Animation: The General Purpose Animator. In *Proceedings of the 1992 Winter Simulation Conference*, eds. J. Swain and D. Goldsman, 366-370. Institute of Electrical and Electronics Engineers, Piscataway, New Jersey.

Brunner, D.T. 1992. *Using Proof Animation.* Annandale, Virginia: Wolverine Software Corporation.

Earle, N.J., D.T. Brunner and J.O. Henriksen. 1990. Proof: The General Purpose Animator. In *Proceedings of the 1990 Winter Simulation Conference*, eds. O. Balci, R.P. Sadowski, and R. Nance, 106-108. Institute of Electrical and Electronics Engineers, Piscataway, New Jersey.

AUTHOR BIOGRAPHIES

NANCY J. EARLE is a Senior Industrial Engineer at Wolverine Software Corporation. She received B.S. and M.S. degrees in Industrial Engineering from Purdue University, with a concentration in simulation. Her responsibilities include consulting, training, technical support, and product development support. Ms. Earle served as the Exhibits Chair for the 1992 Winter Simulation Conference.

JAMES O. HENRIKSEN is the President of Wolverine Software Corporation. He is a frequent contributor to the literature on simulation. Mr. Henriksen served as the Business Chairman of the 1981 Winter Simulation Conference and as the General Chairman of the 1986 Winter Simulation Conference. He has also served on the Board of Directors of the conference as the ACM/SIGSIM representative.