Configurable Semi-Autonimic Animated Animal Characters in Interactive Virtual Reality Applications
Zhuming Ai† Mark A. Livingston†
Naval Research Laboratory

ABSTRACT
In many virtual reality (VR) simulation and training applications, it is desirable and even critical to have computer controlled animal characters that can behave with a high degree of realism. The realism can be measured in two aspects. One is behavioral realism, or how real the characters act and respond to the commands and environments. The other aspect is visual realism.

Many efforts have been directed at animating human characters as well as other animal characters in applications such as interactive computer games. However, more research is still needed for realistic animation of animal characters due to the vast variety of animal species and different application purposes. In some of the environments for which security personnel are being trained, animals, such as dogs, are part of the training programs. So we need to include realistic behavior and visual representations of animal characters in our training applications.

This study is focused on animal behavior and animation for VR applications. A simple implementation of real-time animal animation method that is configurable is proposed to make it easy for user interaction. Synthetic animal characters are included in the system so that their behavior can be programmed and controlled digitally. A two stage state machine is used in the system. The top level state machine controls the animal behavior. The secondary state machine controls animation. Animation blending and procedure control are used to make the animation smooth.

Index Terms:
I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Virtual reality
I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation

1 INTRODUCTION
In many virtual reality (VR) simulation and training applications, it is desirable and even critical to have computer controlled characters that can behave with a high degree of realism. In some of the environments for which security personnel are being trained, animals may be part of the training programs. So we need to include realistic behavior and visual representations of animal characters in our training applications. One such example is that dogs are used extensively in security and law enforcement environments, and there is a need to include realistic dogs in VR training and simulation applications.

Many efforts have been directed at animating human characters in applications such as interactive computer games and motion pictures.[14, 15] We have also seen realistically animated four-legged characters in movies. Realistic simulation of animals in video games can be displayed in real-time. Quadruped animation has made much progress in recent years.[17] There are many efforts on animal animation, especially research on articulated animal movement.[21]

For our purposes, we are more interested in domestic animals that are well-trained and behave in a more predictable manner. Our efforts are concentrated on how the animals are directed by users, but meanwhile react to the environment based on their training as well as instinct. Efforts have been made in our implementation to make such behaviors easily configurable.

The realism can be measured in two aspects. One is behavioral realism, or how real the characters act and respond to the commands and environments. The other aspect is visual realism.

1.1 Behavior Control using AI
Much research has been done on artificial intelligence (AI) for computer games[9], robotics[22], and VR simulation and training[6]. Behavioral models have been used to generate computer animation. Most of the works are focused on a behavioral model for a specific animal in a given environment. Tu and Terzopoulos’s work on artificial fish is one example.[19] Blumberg and Galyean proposed an approach that allows an external entity to direct an autonomous creature at multiple levels.[7] Tomlinson and Blumberg studied synthetic social behavior of wild wolves using a combination of hard-coded behaviors and learned behaviors.[18]

Models have been built to mimic animal behavior in crowd simulation.[23] One example for military applications is the US Army’s OneSAF Testbed Baseline Semi-Automated Forces (OTBSAF) system[5]. The system has computer-generated forces that can interact with customized training or simulation applications through a connection to a local instance of the Run-Time Infrastructure (RTI) through a gateway. We have developed augmented reality training applications that can communicate with it. The users are reflected in real time in OTBSAF as friendly forces, and the computer-generated forces respond appropriately. These responses are sent to the training system to control the visualizations of the computer-generated forces.[8]

1.2 Animal Animation
With motion capture being widely used in animation, human (and humanoid) character animation has reached high fidelity in motion pictures. It is much more difficult to capture animal motions using motion capture techniques, both because of the difficulties on the equipment side as well as on the cooperation of the animals.

Although animal animations have made much progress, more research is still needed for realistic animation of animal characters due to the vast variety of animal species and different application purposes.

This study is focused on animal behavior and animation for VR applications. Synthetic animal characters are included in the simulation and training system so that their behavior can be programmed and controlled digitally. One example of such an application would be a system that can be used to train how to behave in a manner compatible with cultural norms.

† e-mail: zhuming.ai@nrl.navy.mil
† e-mail: mark.livingston@nrl.navy.mil
**Title:** Configurable Semi-Autonomic Animated Animal Characters in Interactive Virtual Reality Applications

**Authors:**

Naval Research Laboratory, 4555 Overlook Ave., SW, Washington, DC, 20375

**DISTRIBUTION/AVAILABILITY STATEMENT**

Approved for public release; distribution unlimited

**ABSTRACT**

In many virtual reality (VR) simulation and training applications it is desirable and even critical to have computer controlled animal characters that can behave with a high degree of realism. The realism can be measured in two aspects. One is behavioral realism or how real the characters act and respond to the commands and environments. The other aspect is visual realism. Many efforts have been directed at animating human characters as well as other animal characters in applications such as interactive computer games. However, more research is still needed for realistic animation of animal characters due to the vast variety of animal species and different application purposes. In some of the environments for which security personnel are being trained, animals, such as dogs, are part of the training programs. So we need to include realistic behavior and visual representations of animal characters in our training applications. This study is focused on animal behavior and animation for VR applications. A simple implementation of real-time animal animation method that is configurable is proposed to make it easy for user interaction. Synthetic animal characters are included in the system so that their behavior can be programmed and controlled digitally. A two stage state machine is used in the system. The top level state machine controls the animal behavior. The secondary state machine controls animation. Animation blending and procedure control are used to make the animation smooth.
There are many well-trained dogs that are widely used in search and rescue, guide, and security applications. We use a dog as an example in this paper to present the user-animal interaction in the VR application.

Among the common dog commands[3], the following selected commands are implemented in the application.

- Sit
- Down
- Stay
- Stand
- Watch
- Search
- Track
- Out
- Go Out
- Here
- Go Out
- Come
- Go Out

To simplify the implementation of the state machine, these commands are combined into three states as shown on the right side of Figure 1. “Sit”, “Down”, “Stay”, “Stand”, and “Watch” are combined as “Stop” state. “Come” and “Here” are combined as “Here” state. “Search”, “Track”, “Out”, and “Go Out” are combined as “Search” state.

When the user issues a command, the state machine will enter one of the user interaction states. In each of the three user command states (“Search”, “Here”, and “Stop”), a secondary state machine is created to control the animation (this will be discussed in Section 3). A subsequent user command will transfer the state machine to a different state. In each state, extra steps for smooth transitions are carried out, which will also be discussed in Section 3.

These commands may include requested directions. A Search command, for example, will generally include a directional cue that indicates the direction that the user desires for the dog to go. Similarly, the user may indicate a Here state with a specific location relative to the user (e.g. left side or right side) where the dog should sit. These secondary goals are included as a field in the state information. Again, the dog may or may not follow these goals precisely.

2.2 Animal Reaction to Environmental Events

There are many different environmental events that may interrupt what the animal is doing in response to user commands. For example, a certain smell can draw a dog’s attention, divert it to a different direction, and cause it to behave completely differently. The following environmental events have been identified for dogs and are being studied:

- The dog is too close to an obstacle, e.g. a building.
- The dog is disturbed by a moving person, object, or other “interesting” things.
- The dog smells something.
- The dog’s running direction may change due to the hill incline or cross wind.
- The dog’s intent to remain on a road even if the road curves.
- Other events.

Each of these events will send the state machine into a corresponding state. The current implementation has a limited number of states for environmental events. It can be expanded when more detailed animal behavior is needed.

There is a possibility that the animal will ignore the environmental events, or even the user’s commands if it is in a “bad” mood. In our application, it is assumed that the animal is accustomed to humans and will always follow the orders. So user commands will always transfer the state machine to user interaction states. However, an easy extension is to have a probabilistic selection of whether to follow or ignore a user command. Among all the environmental events that have been identified, the obstacle event (the first one in the list) is different than the rest. The animal can not possibility go through obstacles such as buildings. But it may choose to ignore the other environmental events. Based on the assumption that the animal’s attention is easily distracted, environmental events will also always transfer the state to one of the environmental event.
In each state in the top-level animal behavior state machine, there is a secondary state machine. This secondary state machine controls the low-level animation sequence of the character. For example, it controls the animal's transformation from standing to running, etc. The following states have been identified as animation states for a dog:

- Stand
- Sit
- Lie down
- Turn
- Run

In each state, there may exist a few parameters that are associated with the state. For example, in the “Turn” state, a vector parameter describes the direction it is turning to. This direction could either be indicated by the user, or determined in the upper level state machine based on the environmental events. In the “Run” state, a speed parameter describes how fast it moves, and further determines if the animal is walking, trotting, or running.

Obviously this list is not complete, but it is sufficient for our current application. For different applications and purposes, more states will be needed. The system is implemented so that new states can be easily added through a configuration file. Although there is still some coding needed in some cases, this structure makes the application easy to expand and improve.

To expand the state machine, the following information is needed. Firstly, if a new animation sequence is necessary, this information should be obtained through motion capture or other methods. Secondly, the state transition table should be modified through a configuration file. And finally, in some cases, some code needs to be added/modified to handle special cases such as multiple paths in the transition table. This will be explained in the next example.

These states are shared by all the animation state machines, but the state transition tables are different. As an example, the animation state machine for “Here,” “Search,” and “Stop” user interaction states in the behavior state machine is shown in Figure 2. The circles labeled “Sit,” “Stand,” “Turn,” and “Run” are states in the animation state machine (“Lie down” state is not shown in this figure). The arrowed lines are actions taken in the top level behavior state machine. For example, the arrowed lines labeled “Search” means that they are actions in the “Search” state of the behavior state machine.

These animation state machines not only transfer from one state to the next, as standard finite-state machines do, but they also have the mechanism to control which path to take in case there are multiple possibilities. For example, if the system received a “Search” command, it may start from a “Sit” state, go through “Stand,” “Turn,” and end at the “Run” state. The small circle in the “Run” state in Figure 2 denotes that the state machine stops here. However, if the user command is “Here,” the animation could start from the “Run” state, go to “Turn,” and back to the “Run” state. Then it could go to the “Stand” state (dash line in the “Run” state in Figure 2) and finally end at the “Sit” state. In the “Run” state, these two commands take different paths.

When the state machine enters a state, a pre-defined animation sequence is played. This animation sequence is a smoothed connection of several animation steps. Take the “Run” state as an example, the animation involves walk, trot, and run, depending on the speed the animal is traveling at the time. These animation steps are smoothly connected (which will be discussed in the following) to form a “Run.”

### 3.2 Modeling

We purchased a dog model and a set of animations. The model is modified and converted to the format that meets our needs.

An animated character contains four properties: skeleton, skin mesh, material (including texture mapping), and animation. For a given character, only the animation property changes during the simulation, all three other properties can remain the same.

By some simple modifications to the texture map or skin mesh (e.g. by changing the color of the texture map), we can create multiple animal characters to populate the VR environment with animal crowds. Figure 3 shows two dogs with different colors and slightly different skin meshes.

The available animation sequences are very limited; we extended the animation sequences by editing existing animal poses using 3D character animation software such as MotionBuilder and 3ds Max. Figure 4 shows the dog in the relaxed lying-down pose and the intense lying-down position after modification.

### 3.3 Animation Blending

It is more difficult to get motion-captured animation sequences for animals than for humans, because animals are not always cooperative. Efforts have been made to use human actors to perform animal actions. Animal animation developed with animation software can also be used. There is also research on animating from video[11, 13], or even from a single picture or a sparse set of pictures of animals[24]. Nonetheless, motion capture is labor-intensive
and expensive. With a limited number of animation sequences available, animation blending can be used to fill the gaps.

Returning to our example of a dog, a set of animation sequences has been determined that can cover the majority of the movement the dog may perform. These include:

- Run
- Trot
- Walk
- Walk happily
- Lay down
- Sniff
- Idle
- Sit

These sequences are connected with interpolations. Cal3D is a 3D character animation library based on skeletal animation.[1] It supports combining animations and actions. This development toolkit is used in our project to build smooth animal animations.

Take the “Run” state in the animation state machine as an example. It starts with an “Idle” animation and is then blended with “Walk.” When the speed goes up, it is further blended with “Trot,” and finally blending into “Run.”

It needs to be pointed out that animation blending works at a level beneath the animation state machine. It can either make the transition between the states smooth, or combine several animation sequences together into one action that otherwise may need a few states in the state machine to implement. The state machine and animation blending work together to accomplish smooth animation.

Figure 5 shows a sequence from a dog animation. The dog was sitting at the beginning, waiting for a command. When it received a “Search” command, it stood up, turned, and started to run with accelerating speed. This sequence is controlled by a series of state changes in the state machine. The changes between states are further smoothed by animation blending.

In some cases, an animation sequence could be interrupted by another animation before it is finished. This has been taken care of on both the state machine level and the animation engine level. The state machine will make sure a proper transitional animation sequence is inserted, and the animation engine will blend the animation to make the transition smooth.

### 3.4 Real-Time Control

Not all animal movements can be pre-recorded with motion capture techniques, simply because there are an unlimited number of possible movements. There are also aspects of the animation that cannot be pre-determined. For example, the animal may turn its head to a certain object that draws its attention. How much the head turns cannot be pre-determined and simulated with a limited pool of animation sequences. Although many movements can be approximated with a limited number of animation sequences, it is not desirable due to poor aesthetic quality. It is important to control some aspects of the animation through software procedures.

One common method used to control animation is inverse kinematics.[20] However, this method is not suitable for real-time VR applications. Guerrero et al. developed a blended inverse kinematics method to achieve real-time animation blending.[12] This method is used in our system to control the animal in real-time.
Figure 5: Animation sequence of the “Search” command. Top left: dog in the “Stop” state in the top-level state machine and in the “Sit” state in the animation state machine. Top center: transitioning to a “Search” state in the top-level machine requires a transition in the animation machine to a “Stand” state. Top right: the animation state machine has transitioned to a “Turn” state to orient the dog properly for the “Search” state in accordance with the user command to search in the direction away from the user’s position. Middle row: the dog has reached the “Run” state in the animation machine. Bottom row: the dog accelerates between the left and right images. Animation blending makes the acceleration smooth.

4 IMPLEMENTATION

The VR application software system is implemented around an open source game engine, Delta3D[2], which uses an OpenGL based open source toolkit, OpenSceneGraph[4], for rendering. An open source approach has been implemented to include animated characters into the system. It uses a 3D character animation library, Cal3D.[1]

Our VR application can reach a real-time frame rate without any problem. On a desktop system with a 2.4GHz Intel Core 2 CPU, 3GB of memory, and an Nvidia GeForce 8800 GTX graphics system, the frame rate is about 37 frames/second with a moderately complex environment model.

The system has a modular design. The AI module is separated from the rendering module. A network layer has been designed to exchange animal character status and user status information with external modules. This is useful when more sophisticated animal behavior modules and environmental models are developed separately.

State machines on both stages can be easily reconfigured through configuration files.

5 DISCUSSION

One of the issues that interests us is to integrate the animal animation into an existing simulation system. There is heavy investment and wide usage of some video game-like simulation and training environments that have many more features than our Delta3D based system. For example, Virtual Battlespace 2 (VBS2) is a battlefield simulation system that is used by the U.S. Marine Corps, Australian Defense Force, and many other military users.[16] For systems using standard interfaces, such as high level architecture (HLA)[10], it might be feasible to integrate a customized animated character into such a system. Otherwise, the possibility of such an integration largely depends on how open the existing system is.

For example, the dog model in the current version of VBS2 has a very small set of animation sequences, and the animation and behavior are not adequate for realistic training and simulation applications. Our VR system is designed in a modular fashion. This has the potential to be implemented as a “plug-in” or a module that can be integrated into existing simulation systems. The initial study shows that, although there are many obstacles, this approach is feasible. So far we have successfully imported the animated dog model into VBS2. With the help of the development tools such as VBS2 Fusion, it is possible to control the animal behavior with an external module. It is also possible to control the animation, may be blended with the animation engine, with external modules. We will further investigate such an issue in the near future.

Rendering large crowds of animal characters is another issue that needs more investigation. How many characters can a computer system handle given all the detailed animation and behavior that is involved? Although it is not the focus of our work at this time, it is
a subject that is worth further study.

6 Conclusions

In some of the environments for which security personnel are being trained, animals, such as dogs, are part of the training programs. Therefore, it is important to include realistic behavior and visual representations of animal characters in training and simulation applications. In this paper, we proposed a schema for developing animated animal characters in interactive VR applications. It is a configurable two-stage state machine based animal behavior simulation and animation. The top level state machine controls the animal behavior. The secondary state machine controls animation. Animation blending and procedure control are used to make the animation smooth.

This study is concentrated on well-trained animals such as dogs. These animals are under the direction of the user most of the time, but respond to the environment as well. The emphasis has been put on making the behavior easily reconfigurable. The two-state state machine structure works very well for this purpose.

References