Chapter 1: Introduction

1.1 Motivation

There are two primary approaches to creating action-selection architectures for autonomous agents. The classical planning approach assumes that an agent has an internal representation of actions, goals, and the state of the world, and this representation is used to plan a sequence of actions to achieve these goals. The plan is handled by an execution process that more or less blindly performs the specified actions. Classical planners can produce near-optimal sequences of actions to satisfy multiple goals in complex environments. However, they tend to fail in dynamic environments or when the state of the world is not knowable [Brooks, 1990; Agre & Chapman, 1987]. For this reason, they are best suited for bounded environments, such as chess. Situated-agent architectures, on the other hand, emphasize direct coupling of perception to action, decentralization, and dynamic interaction with the environment [Maes, 1990]. Situated agents are designed to perform "good enough" actions in an unpredictable environment with incomplete knowledge. However, most situated architectures can not satisfy multiple goals nor perform complex sequences of actions to achieve a goal. They tend to be best suited for lower-level problems in robotics, where the complexity and unknowability of the real world prevents planning a sequence of actions based on an internal model [Brooks, 1991].

There are many environments that fall into neither of these categories but share features of each. In interactive story systems, for example, autonomous agents and human players both play characters in a narrative, and the story progresses partially as a result of character actions [Kelso, 1992]. Agents operating in such an environment must have abilities of both reactive robots and chess-players. Like robots, virtual characters must act in real-time in a rapidly changing and unpredictable world. However, to be believable and interesting a character must be able to perform complex sequences of actions and must at least appear to be motivated by internal goals.

1.2 Characters, not agents

Most autonomous agent systems are designed to satisfy one or more goals in a given environment. How these goals are accomplished is usually secondary to getting the job done. Virtual characters, on the other hand, have more than goals and plans: they have biases, fears, styles, hidden motivations, and a host of other traits that make up an individual personality. These traits affect how a character acts — and it is through these actions that an audience gains insight into a character's personality.

For example, there is one scene in the Walt Disney movie, *Cinderella*, in which the two mice Jacques and Gus are trying to collect kernels of corn without being caught by the cat. Jacques, the wiser, experienced mouse, runs out, grabs a few kernels and runs back to his hole. Gus, the new-comer to the mouse-hole, has not yet learned to be fearful of the cat, and is also much fatter and greedier than Jacques. Gus runs out of the hole and starts collecting every kernel in sight. As he staggers back under his heavy load, he sees one more kernel and tries to pick it up as well, upsetting his already precarious pile and dropping his entire load. Even as the cat was pouncing towards him, Gus furiously tried again to pick up every last kernel. The two mice had the same primary goal, to collect the corn, but because of their personalities they had different ways of achieving this goal.

Gus does not act in ways that optimally achieves his primary goals, and these foibles are what make him a likable and interesting character. However, it is not enough to simply make an error-prone agent and call it a character. If this were the case, the job would be easy! Rather, it is the complex melding of multiple, often conflicting motivations that make a personality. Jacques acted as he did because of the interaction between his hunger, his fear of cats, his desire to not take risks, his desire to make the cat looks stupid, and his desire to help people. Gus has a much higher relative desire to get every last kernel of corn, and this difference in desires shows through his actions.

Because a character's personality shines through his actions, convincing, believable characters for interactive story systems must be based on a good action selection architecture. This architecture must be able to integrate multiple motivations in choosing a character's actions. The architecture must also be able to choose between many different methods to achieve the same goal, because it is the choices between these different methods that best display a character's personality. These and other desirable traits in an action-selection algorithm for interactive story systems are discussed below.

1.3 The domain addressed by this thesis

Even among action-selection architectures specifically designed for characters in interactive story-systems, there are many different approaches. Some architectures try to make characters with animal-like behavior and thus look to ethology for inspiration [Terzopoulos, 1995; Blumberg, 1994]. Other architectures, such as earlier versions of the OZ project's HAP system [Loyall, 1991], are designed for creating more human-like characters that can handle complicated sequences of actions in environments where the effects of actions are atomic and the geometry of the world is discrete. This type of environment is commonly found in text-based interactive stories, where the single action *walk-north* will move to a completely new room or location. These and related architectures are described in Chapter 2.

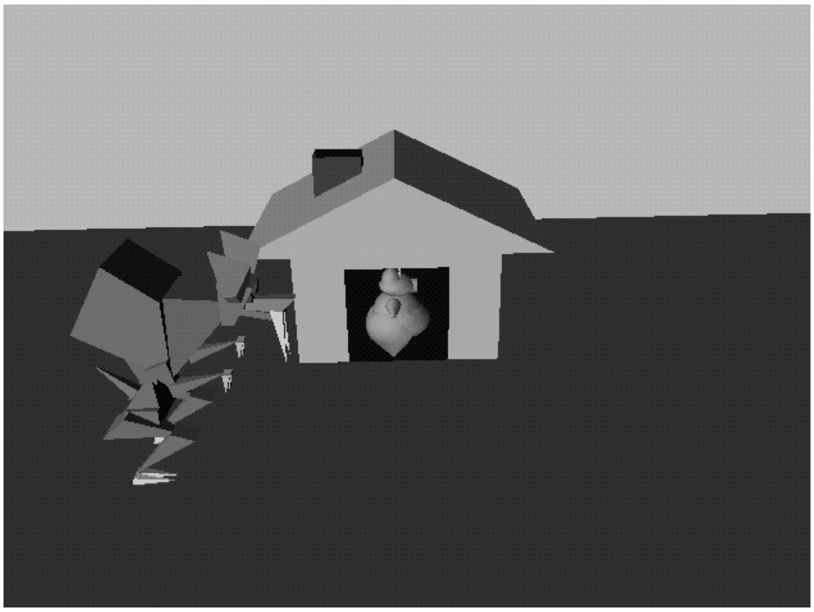

Figure 1: The Big Bad Wolf, a PHISH-Nets Character

The algorithm described in this thesis, called PHISH-Nets (Planning Heuristically In Situated Hybrid Networks), is designed to create characters that interact with human players in a graphical environment. These characters would not necessarily be human (language, for example, is completely ignored by the algorithm). However, neither would they be completely animalistic in how they act to satisfy their goals. The type of character envisioned by this algorithm is closest to Wile E. Coyote from the old Warner Brothers *Coyote / Roadrunner* cartoons, who, although he does not speak, can still make very complicated plans to catch the Roadrunner.

These characters should have a large repertoire of actions, many of which could potentially, but perhaps unreliably, satisfy the same goal. They should also be able to have multiple, possibly conflicting goals. Finally, they should be able to produce complex sequences of actions to achieve these goals. All these traits make a character more interesting over time and give it greater flexibility to act in ways that are "in character."

Because human players are involved, a character must also be robust in an unpredictable world and handle failure gracefully.

While PHISH-Nets are designed with the application of interactive story systems in mind, they can be more generally applied to create characters that operate in environments with the following traits:

- The world is continuous. In such a world, character actions are usually fine-grained, where a single instantiation of *walk* might only take a single step forward.
- Because actions take time to accomplish, the environment might change before an action is completed, forcing the character to change his plan.
- The world is dynamic and unpredictable. Character actions might not have the effects intended, or they may fail entirely.
- Characters have multiple, possibly conflicting goals.
- Characters have multiple means to accomplish the same thing.
- Characters must often perform long sequences of actions in order to accomplish a goal. Sequences of five to ten actions should not pose a difficulty.
- Characters should be easily extendable. New actions, objects, and situations should easily be added to a character or the world without forcing an author to hand-code how each new element relates to the old.

There are two things this thesis is specifically not trying to address. First, PHISH-Nets is not proposed as an explanation for the internals of real animals or humans. Second, the graphical characters created should not be mistaken for simulations of physical robots which might be built. Although the algorithm may be used for physical robots if the domain matches the above traits, the graphical characters created are an end themselves rather than a simulation.

1.4 Roadmap to the thesis

Chapter two gives a brief introduction to the related work in the field of autonomous virtual characters. It includes a lengthy description of Maes' *How To Do The Right Thing* algorithm, which was the starting point for this work. It also briefly describes the behavior systems in *Hamsterdam*, *HAP*, and Minsky's *Society of Mind*, as well as Perlin's real-time graphical puppets and Terzopoulos' artificial fishes, which focus more on motion and realistic graphics than on behavior.

Chapter three describes in detail the PHISH-Nets algorithm. It first describes the high-level character-specification language, which is used by an author to encode a character's actions and goals. It then describes the structure of a PHISH-Net, the network that is compiled from this high-level specification. The action-selection loop is described along with an example. Finally, the rather extensive system for detecting and handling failure conditions is explained.

Chapter four describes test scenarios that have been run using the implementation of PHISH-Nets. These test-runs all take place in a 3D graphical interactive story based on *The Three Little Pigs* story. In them, the Big Bad Wolf tries to satisfy his hunger and urge for wanton destruction through a series of scenarios that demonstrate the algorithm's ability to handle multiple methods for solving a problem, multiple interacting goals, opportunistic behavior, and repeated failure of actions leading to eventually giving up on a goal entirely.

Chapter five discusses the creation of characters with personality, the extensibility of PHISH-Nets characters, and speed of the algorithm. It also compares the algorithm with its closest kin, Maes' Agent Network Architecture (ANA). The chapter closes with a discussion of the limitations currently inherent in the algorithm and future work yet to be performed.

Chapter six concludes with a brief summary of this thesis.