44

A Control-Based Architecture for Animal Behavior

Michael Ramsey

44.1 Introduction

Many games include creatures or animals that exhibit the illusion of life while interacting with the game player and with the world around them. This illusion breaks when a creature does something that seems out of character or unnatural—and these types of breaks in illusion are unfortunately, all too common. Thus, we need to provide the ability for our characters to exhibit *believable behavior* that is purposeful, while being robust enough to appear fully life-like.

For this article, we define *behavior* as the actions an agent performs. *Purposeful behavior* describes actions performed in the context of some objective (for example, going to a food bowl in order to eat, or climbing a tree to sleep in its branches). There are numerous well-known architectures that can generate sets of behaviors and chain them together to be purposeful (for example *behavior trees* (BTs), *finite-state machines* (FSMs), *hierarchical task network planners* (HTNs), and so forth), and many of these are discussed elsewhere in this book. However, if our behavior is going to be *believable* then it needs to be more than just the output from a BT or other AI architecture. It requires a system which can deliver the appropriate interactions regardless of the ever-changing situation in-game.

If purposeful behavior is going to be believable, then it must produce consistent results regardless of varying environmental conditions. This is something that real-world creatures typically handle without much thought, but for an AI character it can be quite hard. Think about the sequence of motions associated with something as simple as walking across a room and hitting a light switch, for example. They vary greatly depending on your starting position and stance, the exact position of the light switch, the types and locations of obstacles between the two, the type of surface being traversed, etc.

Adaptive, purposeful behavior was perhaps best characterized in 1890 by the American psychologist William James [James 90]:

> Romeo wants Juliet as the filings want the magnet; and if no obstacles intervene he moves towards her by as straight a line as they. But Romeo and Juliet, if a wall be built between them, do not remain idiotically pressing their faces against its opposite sides like the magnet and the filings with the card. Romeo soon finds a circuitous way, by scaling the wall or otherwise, of touching Juliet's lips directly.

William James' description of Romeo's behavior provides an intuition of how purposeful behavior could inform our work on virtual characters. Just as Romeo with Juliet, our behavioral modeling solution needs to take into account and handle the variability of the world, and adapt accordingly—so when it is time to kiss, we can make that happen whether it requires crossing a dance floor, climbing a balcony, or simply bending over to pickup a dagger.

World of Zoo is an example of a game that could easily suffer from issues with the believability of its behavior in just this manner. This game is an animal simulator that contains various types of zoo animals in a variety of environments. The player can alter the layout of the zoo, changing the environment around the animals. In the worst case, this can result in massive topological alterations to the animals' game world that directly interferes with an animal's current behavior! This is actually not at all uncommon, since the players rarely plan their changes around the animals' actions, and in some cases will even actively try to "mess with them." As a result, animals need to be able to react believably to shifts in their environment that cannot be predicted when their behavior is being crafted during development, or even when the behavior is being selected by their high-level AI.

The remainder of this chapter provides an introduction to aspects of controller theory that can be used to implement a behavioral system for life-like animals.

44.2 A Control System

Even the simplest observation of the physical world demonstrates that environments do not remain static. No fixed, predetermined behavior will allow an animal to cope with the world, so behavior must somehow take into account an ever-changing and unpredictable world. Clearly then, our animal AI must continually modify its behavior to adjust for disturbances that would cause a predetermined behavior to look incorrect—but how and where do we find the appropriate modifications to make?

One solution can be found in the work of electrical and mechanical engineers in the early 1930s in a discipline known as *control theory*. Where a behavior-centric approach sees the world as being under the control of the animal through its behavior, control theory

instead sees the animal as being at the mercy of the world, but able to vary its behavior to compensate for changes in the world through the use of *control systems*. Examples of these systems are found in a variety of electronic devices, such as the thermostat in your home which regulates the heating or cooling system, or even automated antiaircraft guns on naval vessels.

To help us come to grips with what a control system is, we begin with a high-level model of a control system. Fundamentally, there are four things that these systems do:

1. Signals are received.
2. Signals are analyzed.
3. Instructions are made to act on the analyzed signal.
4. These instructions are used to do something in the world or in another control system.

A great example of this kind of control system in practice is the way the cruise control of a vehicle maintains a steady driving speed in the absence of input from the driver. When the cruise control is engaged with a particular desired speed, this speed becomes the control system's *reference signal* (which is also the system's desired goal). If the vehicle's speed varies from the goal, then the control system will increase or decrease the amount of fuel that is delivered to the engine, thus adjusting the speed until it once again matches the goal. Because this system is designed to minimize the variance between the actual value and the goal value, it is referred to as a negative feedback system. We will focus on negative feedback systems throughout the remainder of the chapter.

44.3 Perceptual Control Systems—Negative Feedback

As shown in Figure 44.1, negative feedback can be visualized as a marble in a curved cup. The bottom of the cup represents our reference signal—that is, the value that we want to retain. Much like gravity, the control system constantly pulls the actual value down into the cup, until it matches the goal value that is at the lowest point. When external forces push the actual value away from our desired value (imagine moving the cup from side to side or knocking the marble upward with a finger), the negative feedback brings it back down again.

A more detailed view of how a negative feedback system can be implemented is shown in Figure 44.2. The *input function* converts some variable aspect of the world into a

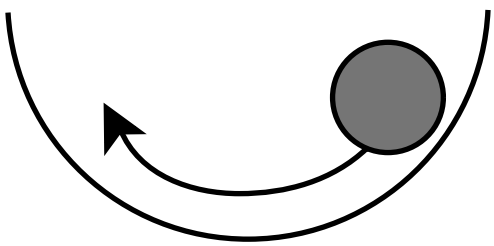

Figure 44.1

Negative feedback in action.

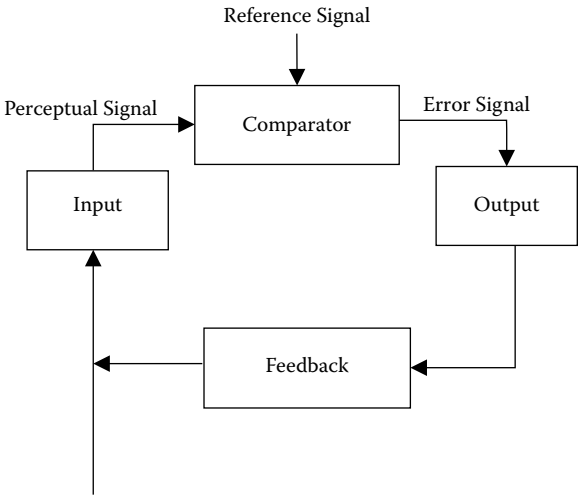

Figure 44.2

A sample negative feedback control system.

perceptual signal. The perceptual signal is then compared with a *reference signal* within the *comparator function*. The difference between these two, called the *error signal*, is then converted within the *output function* into a behavior. A *gain factor* can also be applied to the output. The gain adjusts the magnitude of our output behavior—so in the case of a cruise control, for example, it would change the rate at which we speed up or slow down.

The behavior, as altered by the gain, then acts on the world, changing it in the desired manner. The variable that the system is controlling is not only being influenced by the output from the system, however—it is still being impacted by the ever-changing state of the world. Thus, we continuously reexamine the state of the world, feed that into our control system, and then modify our behavior based on the output of the control system in an attempt to keep the variable as close to the reference signal as possible.

Returning to the cruise control example, we see that a cruise control system has a number of components that allow for the exhibition of purposeful behavior—behavior that is perceived to compensate for disturbances, even when those disturbances are *unknown*. The control system has no understanding of whether the vehicle is proceeding up an incline or descending, if it is hauling a camper, if there is wind resistance, or if the engine is not fully functional. The only aspects of the world that this control system has knowledge of are the vehicle's speed and the rate of fuel delivery to the engine. In this case, the *vehicle's speed* is the *controlled variable*; however, a control system does not directly control its *value*, instead it controls *how that value changes*. The cruise control system senses the speed of the vehicle and the only control it has is by adjusting the flow of fuel. To maintain the goal (set as the reference signal) the control system must continue sensing the vehicle's speed as disturbances attempt to vary the control system's input. Any disturbances are then compensated for and the exhibited behavior changes as the conditions change around the vehicle.

Although we have discussed control systems in the context of automating a process for the user, such as the driver setting the cruise control's desired speed, it's important to highlight that the reference level does not have to be defined as the game is played; the initial value can be specified by a designer during development and then, if necessary, updated as the game progresses.

44.4 Hierarchy of Control Systems

In 1989, William Powers proposed a complex hierarchy with 11 ordered levels of control systems as the foundation for human and animal perception [Powers 89]. The details of this approach are more complex than appropriate for our animal AI, but the general concept is quite useful. The key idea (see Figure 44.3) is that the higher order control systems adjust the lower-order systems' reference signals, and the input that is sensed by the lower-order systems is also made available to the high-order systems. Perceptions in the higher level components of the system are typically composed of combinations of lower-order systems, which are then controlled by the way the high-level system alters their reference level. In this way, we don't create a hierarchy of command; instead we have

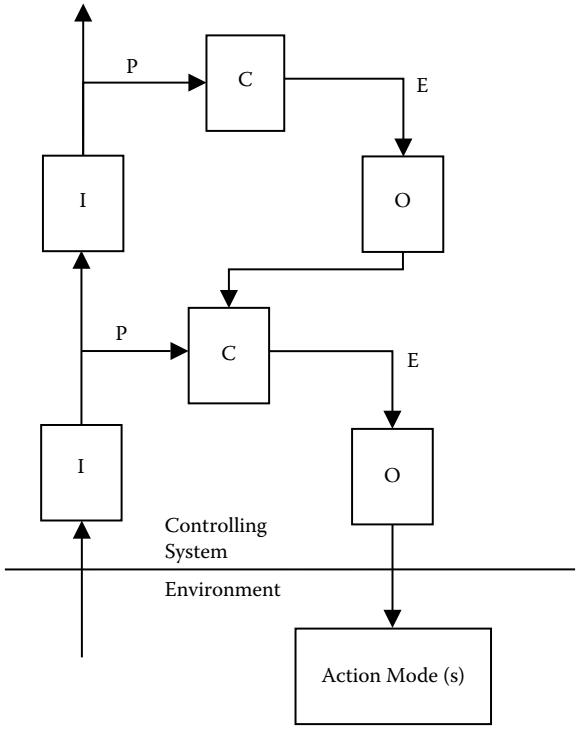

Figure 44.3

A sample hierarchical control system illustrating the input being propagated up to the higher order control system, while the output from the higher order system feeds back as the reference signal. P represents the incoming perceptual signals to a comparator, and E is the applied error amount.

high-level systems adjusting the reference values of the low-level systems, and then those low-level systems directly outputting the appropriate behavior.

Returning once again to the cruise control example, some newer cars have "intelligent cruise controls" that will automatically slow down when following another vehicle. This can be achieved by adding a higher order layer to the system. The new layer takes the input from the existing component (the current speed), but it also considers how closely the car that it's controlling is following the vehicle in front of it. The output for this new component is the reference value for the lower level system—that is, it adjusts the desired speed of the cruise control automatically so as to slow down and avoid a collision.

44.5 Control Systems for Animal Behavior

While many animals appear to exhibit nothing more than a simple stimulus and response mechanic, research into animal psychology has discovered that much of animal behavior is far more complex and intricate than the execution of an if/then/else test. One of the first studies of defensive behavior in an animal's natural environment was conducted by Heini Hediger [Hediger 55]. Hediger provided the initial description of a *flight zone*. The key idea is that an animal will not simply flee at the sight of a predator, but will wait until the predator has approached within a specific distance, at which time the threatened animal will move away in order to recreate the desired space. In Hediger's work we see that defense is not simply a reflexive response to a stimulus, but is instead a constant process of spatial assessment and movement. These constantly executing and assessing systems are, in essence, control systems.

44.6 Customizing a Negative Feedback Controller—Game Extensions

The cruise control system outlined above focused on managing a single floating-point control variable (the speed of the vehicle) using one or at most two controllers. Animals need to be able to do more than travel a linear path at a fixed speed, so they will require more complex inputs, and a greater number of controllers. As a result, we need to provide a custom negative feedback controller implementation that operates on coarse data structures/objects, [Ramsey 09, Ramsey 10]. Examples of controllers appropriate to animal control include the above mentioned flight zone controller, a personal space controller, a biological urge controller (e.g., food, water, or sleep [Toda 82]), and an environmental spatial orientation controller [Ramsey 11].

Perhaps the most important customization is to provide a representation for "spatial semantics." In other words, we need to be able to represent concepts about locations to the control system's input. Examples of these concepts include the location(s) of nearby food sources, nearby obstacles, other entities (friendly or hostile), and so forth. The comparator can then work in much the same way—examining the input signal (perhaps the animal's current location), comparing that to a reference signal (perhaps a position adjacent to a nearby food source), and returning an appropriate action (such as going to the food source and eating it).

Another important modification to the controller model is the addition of a priority value. This is something that will be very dependent on the type of game you are making,

but generally speaking, if one control system has relatively low priority its output may be ignored in favor of the output from another control system with a higher priority. The details of this will become more apparent below.

44.7 Gosling Control System

At this point we have all of the building blocks required to create a realistic simulation of a young goose, or "gosling," highlighting some of the interesting ways it can believably interact with a small world.

One approach to tackling the issue of engineering a behavioral system is to attribute a wide variety of drives, needs, or urges to our virtual animal [Toda 82]. These motivations are modifiable by both the designer during development and the game at runtime, and this allows us to influence the current state of our higher order control system. In our example we'll assign two drives to our gosling that are relevant in our example world: security and curiosity.

Konrad Lorenz [Lorenz 81] discovered that a gosling will imprint on the first large animate object that it sees, which is typically its mother. This imprint serves as the initial priming of a control system that requires the gosling to maintain close contact with its mother throughout gosling-hood, by monitoring this distance and moving to compensate for any disturbances that are detected. In nature, these disturbances may be caused by the mother's actions, such as moving away, or by obstacles preventing the gosling from approaching the mother directly.

Of course, goslings do things other than simply stay near their mothers. They also seek out food, avoid predators, and explore the environment around themselves, among other things. Each of those things, taken in isolation, can be modeled fairly simply with a negative feedback control system. So we have one control system responsible for maintaining the distance from the mother; another which tries to move toward (and eat) nearby food; a third that is trying to avoid predators; and a fourth that is trying to seek out things that the gosling hasn't yet examined—trees, rocks, flowers, or areas that it hasn't been to.

Each of those systems takes the gosling's current position, and tries to adjust that position based on an input signal (the position of the nearest food, the position of nearby predators, and so forth). The trick, then, is to decide which of these four controllers should actually control the gosling at any given time—and that is where the priorities come in. Each controller is assigned a priority based on its current urgency. So the controller that seeks out the mother will be lower or higher priority, depending on how far away the mother is. The one that seeks out food will have a priority that depends on how hungry the gosling is. The one that seeks to flee from predators will be very high priority—but only when there is a predator within the flight zone. Finally, the controller that seeks to explore will be relatively low priority, but will always be active.

44.8 Conclusion

This chapter gives an introduction to the use of negative feedback controllers to produce animal behavior that takes into account the dynamic nature of a game world's environment. These control systems use the spatial semantics from the environment to inform its input and compare this with a designer specified reference signal. It is then able to

generate an error signal that is used to drive the animal's activity. It's not enough to stitch independent predetermined behaviors together; we need an underlying mechanism that through their interaction with the environment is able to exhibit corrective measures to the animal's desired (reference) state—which then can be viewed as a behavior.

References

[Hediger 55] H. Hediger. *Studies of the Psychology and Behavior of Animals.* New York: Criterion Books, 1955.

[James 90] W. James. *The Principles of Psychology.* New York: Dover Publications, 1890.

[Lorenz 81] K. Lorenz. *The Foundations of Ethology*. New York: Simon and Schuster, 1981.

[Powers 89], W. Powers, *Behavior: The Control of Perception.* New Canaan: Benchmark Publications, 1989.

[Ramsey 09] M. Ramsey. "A unified spatial representation for navigation systems." *Proceedings of the Fifth AAAI Artificial Intelligence and Interactive Digital Conference*, 2009.

[Ramsey 10] M. Ramsey. "A practical spatial architecture for animal and agent navigation." In *Game Programming Gems 8*, edited by Adam Lake. Boston: Charles River Media, 2010.

[Ramsey 11] M. Ramsey, "An egocentric motion management system." In *Game Engine Gems 2*, edited by Eric Lengyel. A K Peters, 2011.

[Toda 82] M. Toda, *Man, Robot and Society: Models and Speculations.* Dordrecht: Martinus Nijhoff Publishing, 1982.