Primitives and Behavior-Based Architectures for Interactive Entertainment

Odest Chadwicke Jenkins and Maja J Matarić
Computer Science Department
University of Southern California
941 West 37th Place, Mailcode 0781
Los Angeles, CA 90089-0781
cjenkins|mataric@cs.usc.edu

Abstract

Behavior-based architechtures use a basis set of behaviors as a primitive-level action vocabulary for an autonomous agent, such as a character in an interactive environment. This vocabulary can be extended to perform more complex actions by sequencing and superimposing primitive behaviors. Perceptual-motor primitives are an extension of the behavior-based architecture in which the behaviors encode perceptual characteristics for observing the behavior when executed by another agent, in addition to control mechanisms for executing the behavior. These architectures are applicable to character control for interactive environments. Behavior-based control has been a subject of research in multi-robot control and has potential for controlling groups of characters as well as creating autonomous characters. Control of articulated characters based on perceptual-motor primitives is a recent area of research and is applicable to extending the range of actions for user-controlled and computer-controlled characters and incorporating new human-centered input mechanisms into gameplay.

Introduction

The control of characters, in particular articulated characters, for interactive virtual environments can be difficult and tedious to implement. As persuasively argued in several articles in Game Developer Magazine (Hecker 1996; 2000), using physics-based, rigid-body dynamics in such environments adds substantially more difficulty to this problem. This fact can be evidenced by the amount of effort spent by both entertainment companies and the research community on controlling characters. One approach to generating believable character control is to endow the character with plausible behaviors for interaction with a complex environment, in order to generate appropriately realistic autonomous behavior. We advocate the use of behavior-based architectures to endow such characters with autonomy. Specifically, we present a concept-level discussion of inspirations and properties of behavior-based systems, then describe an architecture for character control we have developed based on perceptual-motor primitives, and finally describe potential applications of this approach to video games and interactive entertainment.

Behavior-Based Architectures

The research resulting from several lines of our work advocates the use of behavior-based control for driving characters for interactive virtual entertainment environments. Behavior-based control involves the design of control systems that consist of a collection of behaviors (Arkin 1998), real-time processes that take inputs from sensors (such as vision, sonar, infra-red), or other behaviors, and send output commands to effectors (wheels, motors, arms), or to other behaviors in the system (Matarić 1997a). The controller, then, is a distributed network of such communicating, concurrently executing behaviors resulting in excellent real-time and scaling properties. The interaction of the behaviors through the environment results in the desired overall system performance.

The inspiration for behavior-based control comes from biology, specifically from neuroscience evidence, which suggests that natural systems may be similarly organized (Bizzi, Mussa-Ivaldi, & Giszter 1991; Matarić 1995). We have focused on applying the principles of behavior organization to high-dimensional behavior-based systems, such as humanoids and groups of interacting robots. In both problem domains, we have used basis behaviors, or primitives, in order to structure and simplify the control problem, as well as enable adaptation and learning (Matarić 1997a).

Several methods for principled behavior design and coordination have been proposed (Arkin 1998). (Matarić & Marjanović 1993) and (Matarić 1995) introduced the concept of basis behaviors, a small set of necessary and sufficient behaviors that could be composed through sequencing or superposition, as a means of handling controller complexity and simplifying robot programming. Basis behaviors are the primitives that serve as a substrate for control, representation, and learning in behavior-based systems. This paradigm was first demonstrated on groups of mobile robots. A basis set consisting of avoidance, following, homing, aggregation, and dispersion was used to demonstrate higher-level group behaviors including flocking, foraging/collection, and herding (Matarić 1995). We also demonstrated how, given such a basis behavior set, a learning algorithm could be applied to improve behavior selection over time (Matarić 1997b).

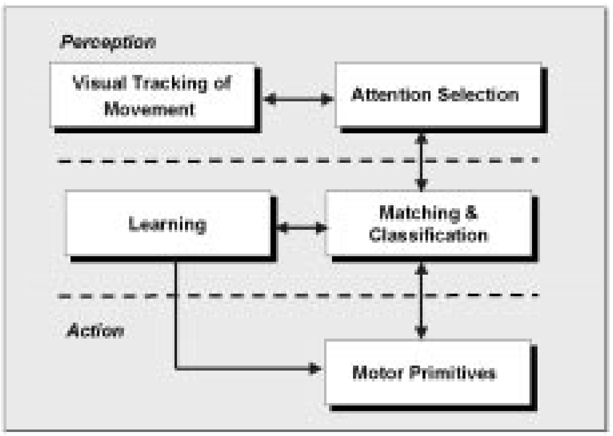

Figure 1: *Our imitation model.*

Perceptual-Motor Primitives

As the complexity of a character or a robot increases, the need for a modular approach for control also increases in order to keep control viable and efficient. Humanoid agents are highly complex; a human arm has 7 degrees of freedom, the hand 23, and the control of an actuated human spine is beyond current consideration. Yet humans realize complex dynamic behaviors in real time and learn various motor skills throughout life, often through imitation. This leads us to look for inspiration from biology and neuroscience. Evidence from neuroscience points to two neural structures of key relevance: spinal fields and mirror neurons. Spinal fields, discovered in frogs and rats, code for complete primitive movements (or behaviors), such as reaching and wiping (Bizzi, Mussa-Ivaldi, & Giszter 1991) and are additive when multiple fields are stimulated, thus resulting in a meaningful combination of movement. Because the spine codes a finite number of such fields, they represent a basis set of primitives, and were precisely the inspiration for our work on basis behaviors, described above. Mirror neurons, found in monkeys and humans, appear to directly connect the visual and motor control systems by mapping the observation of behaviors, such as grasping, to motor structures that execute them (Rizzolatti *et al.* 1996). We combine these two lines of evidence into a more sophisticated notion of behaviors, known as perceptual-motor primitives. This type of behavior allows a complex character, such as a humanoid, to recognize, reproduce, and learn motor skills. The primitives are used as the basis set for generating movements, but also as a "vocabulary" for classifying observed movements into executable movements (Jenkins, Matarić, & Weber 2000; Matarić 2000).

We have incorporated our concept of perceptual-motor primitives into an architecture for motor control and skill learning through imitation. The architecture consists of three layers: Perception, Encoding, and Action, and five components: Tracking, Attention, Classification, Learning, and Execution (Jenkins, Matarić, & Weber 2000). The Perception layer has two components that serve to acquire observation data (Tracking) and situate this data according to its saliency (Attention). The Encoding layer generates a set of perceptual-motor primitives from a set of observation data (Learning) and encodes incoming observation data into the set of primitives (Classification). For a character, a perceptual-motor primitive is a set of motor controllers and a set of associated perceptual descriptions that correspond to visual descriptions of the controllers' execution. Primitives are then combined through concurrent or sequential activation to create arbitrarily complex character behavior.

In the ideal case, the Learning component in the architecture provides a blueprint for generating the motor controllers by the Execution component in the Action layer and the perceptual descriptions are used for matching in the Classification component. It is possible, however, that this ideal case may not be feasible or desired for certain characters due to several influences. Among these are bottom-up limitations, based on the kinematic and dynamic properties of the character, and top-down constraints, based on the type of movements the character will be expected to perform, and the structure of the systems used for tracking and perception. Thus, a suitable set of perceptual-motor primitives to use in our architecture can be hand-selected or derived. In order to serve as a general and parsimonious basis set, the primitives encode groups or classes of stereotypical motion, invariant to exact position, rate of motion, size, and perspective. The primitives represent a set of "generic" building blocks of motion that can be implemented as parametric motor controllers.

For the case of controlling end-effector motion, this approach stands in sharp contrast to explicit planning, which computes trajectories at run-time whenever they are needed. While fully general, on-demand trajectory generation is computationally expensive and potentially slow. In our approach, stereotypical trajectories are built-in as well as learned, then looked up and parameterized for the specific task at hand. The notion of primitives takes advantage of the fact that it is simpler to learn and reuse an approximation of the inverse kinematics for specific areas in the workspace or a specific trajectory, then it is to compute them anew each time.

We have experimented with three means of obtaining a set of primitives for humanoid agents: 1) manually designing and encoding the set by hand (Matarić, Zordan, & Williamson 1999), 2) manually extracting the set from human movement data (Jenkins, Matarić, & Weber 2000), and 3) automatically extracting the set from human movement data (Fod, Matarić, & Jenkins 2000). In that work, we have focused on the control of the upper body of a humanoid character named Adonis. Adonis is a physics-based dynamic humanoid simulation with 20 DOF, 7 per arm, and uses equations of motion generated from the commercial package SD/Fast. For our tracking information, we have used a variety of source including a 2D vision system for Cartesian feature extraction we developed (Weber 2000), FastTrak magnetic motion capture, and the Sarcos SenSuit wearable exoskeleton. This has allowed us to experiment with a variety of input data types, from visually-acquired features, to computed joint angles and/or end-point position.

The details of these approaches can be found in the refered papers, which are available at: *http://robotics.usc.edu/∼agents/imitation.html* and *http://robotics.usc.edu/∼agents/macarena.html*

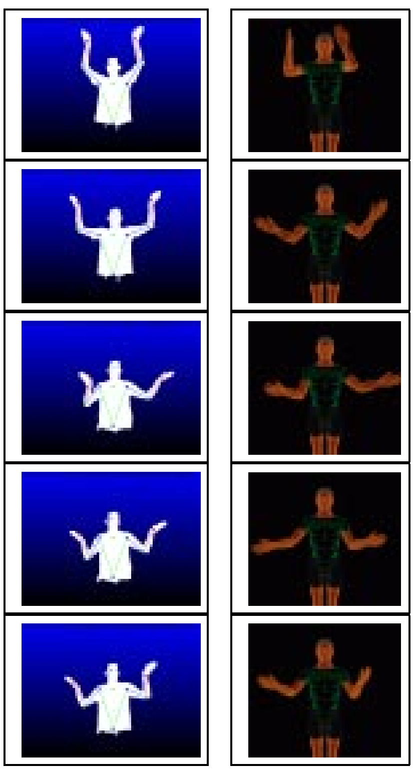

Figure 2: *Left: Snapshots of the output from tracking using (Weber 2000). Right: Snapshots of our humanoid simulation, Adonis, imitating the tracked motion.*

Applications to Interactive Entertainment

The Basic Idea

The application of perceptual-motor primitives is aimed at providing a vocabulary for motion to use with new user interfaces in games. The ideal case for such an interface would use vision sensing or motion capture to get a moving skeleton of an unmarkered user. Since each primitive represents a behavior, and each behavior has a meaning, the set of primitives can be used to represent perceived motion in terms of these meanings. For instance, certain motions correspond to a single button on a console controller. Thus, the user's rough performance of a certain action could execute the character's behavior in the game. For games like Tekken and Dance Dance Revolution (discussed below), this interaction could provide some aerobic exercise in the process. In addition, the user's motion is not classified in a mutually exclusive manner to the character's action, but rather encoded into the action repertoire of the character. Thus, an arbitrary motion by the user could be interpreted by the character to possibly create more interesting actions. The use of such a motion vocabulary is gaining popularity (Bregler 1997; Brand & Hertzmann 2000).

Motivation

Our area of concentration has been on the control of humanoid characters with rigid body dynamics. For a given character, a set of perceptual-motor primitives could be built from a set of motion data that the character would be expected to execute. Currently, animation for characters in video games is generated through motion capture or capable animators, and stored in motion libraries. Given commands from the user's controller, selected segments from the character's motion library are sequenced to perform the desired action. However, these segments are a fixed set of angles spanning a length of time and, thus, are not easily amenible to the creation of more complicated actions. This problem, refered to as motion editing, has been addressed by several approaches in the computer graphics community, including (Popovic & Witkin 1999; Gleicher 1999; Brand & Hertzmann 2000). While these methods yield desireable motion, each is based on optimizing some set of constraints, which is not well suited for on-line performance. This shortcoming is similar to that of trajectory planning.

In contrast, a set of perceptual-motor primitives serves as a more general model of a character's motion. In a sense, each primitive acts as a parameterized control policy. The parameters are used as indices to look up the appropriate control commands for executing a particluar motion. More complex motion can then be performed by taking control commands from multiple primitives executing them independently. This viewpoint leads to two applications of primitives user interfaces for video games.

Extensions to Traditional Interaction

The first of these applications is the use of primitive sets with systems using traditional methods, such as directional/button controllers. The flexiblity of using primitives will allow the user to create a wide variety of new actions by specifying parameters for the primitives that will sequence and combine known actions. These new actions can then be activated by a customized set of commands from the controller. In other words, the primitives are user-manipulated to supply a new vocabulary (or meaning) to the commands of the controller.

To further extend this idea, consider one of our favorite series of games, the Tekken series (Namco 2000). Tekken is a standard 3D fighting game with two opponents inflicting damage on each other using martial arts moves. However, the game is limited in that its main challenge is learning and executing the extensive list of key combinations to unleash the appropriate motion-captured move. By incorporating primitives, a new level of skill and strategy will result from the user's ability to craft new moves. Furthermore, the computer characters will have access to a motion vocabulary that will allow for more challenging artifical opponents, via off-line improvement by a genetic algorithm or other competition-based learning. Judging from the number of sites devoted to Tekken FAQ's and movelists on the Internet, there are probably great numbers of players that would welcome another level of intricacy and difficulty to such games with the ability to merge hand-tailored moves with their personal strategy. Furthermore, primitives also allow for character control in physically-based environments.

Preparing for New Types of Interaction

The second area of application for primitive sets is for use with new types of input/tracking mechanisms during game play, such as motion capture or vision tracking. Some day in the near future there will probably exist inexpensive motion capture/tracking systems that could be feasible as a user interface for gameplay. Such a method of interaction is a potentially powerful mechanism for gameplay and immersion. However, motion capture alone may not be sufficient because it is a literal interpretation of the user's pose and provides only a set of joint angles that yield an animation. This motion information is just data without meaning. By encoding incoming motion from the user into a primitive set, the motion is given meaning. When combined with the literal pose information, the primitive encoding provides information about the user's performance of certain actions, such as jumping or reaching in a certain direction, as well as the user's current configuration.

A current arcade/music game that exhibits the potential for using primitives is Dance Dance Revolution (Konami 2000). In this game, the user stands on a platform in front of a monitor. The platform has four square sensors with arrows indicating up, down, right, and left. In the game, the monitor scrolls a list of arrows indicating sensors where the user should step on at the current time. As the speed of the scrolling arrows increases, the user must start to dance in order to keep up the pace. With vision tracking or motion capture, the actions that can be expected of and performed by the user can be vastly increased. As stated before, those techniques alone cannot provide information about when and if a certain dance move was performed by a user. By encoding the observed motion into primitives, the motion is described in terms of a meaningful vocabulary, grounded in the generated primitive set. Based on the chosen or generated primitive set, the encoded motion now has meaning in terms of the primitive set and is amenable to the detection of certain movements.

Conclusion

Our approach and continuing research are aimed at user-character and multi-character interaction. We believe that the primitives-based control architecture we are developing is an intuitive means for both control and interaction. We welcome the opportunity to interact with the interactive game community.

References

Arkin, R. C. 1998. *Behavior-Based Robotics*. Cambridge, Massachusetts, USA: MIT Press.

Bizzi, E.; Mussa-Ivaldi, F.; and Giszter, S. 1991. Computations underlying the execution of movement: A biological perspective. *Science* 253:287–291.

Brand, M., and Hertzmann, A. 2000. Style machines. In *The Proc. of ACM SIGGRAPH 2000*, 183–192.

Bregler, C. 1997. Learning and recognizing human dynamics in video sequences. In *Proc. IEEE Conference on Computer Vision and Pattern Recoginition*.

Fod, A.; Matarić, M.; and Jenkins, O. 2000. Automated derivation of primitives for movement classification. In *Proc. of First IEEE-RAS International Conference on Humanoid Robots*.

Gleicher, M. 1999. Motion editing with spacetime constraints. In *SIGGRAPH 1997*, 139–148.

Hecker, C. 1996. Behind the screen: The next frontier. *Game Developer Magazine*.

Hecker, C. 2000. How to simulate a ponytail. *Game Developer Magazine*.

Jenkins, O. C.; Matarić, M.; and Weber, S. 2000. Primitive-based movement classification for humanoid imitation. In *Proc. of First IEEE-RAS International Conference on Humanoid Robots*.

Konami. 2000. Dance dance revolution description. WWW Site. http://www.konami-arcade.com/Music/Ddr/.

Matarić, M. J., and Marjanović, M. J. 1993. Synthesizing complex behaviors by composing simple primitives. In *Self Organization and Life: From Simple Rules to Global Complexity, European Conference on Artificial Life (ECAL-93)*, 698–707.

Matarić, M. J.; Zordan, V. B.; and Williamson, M. 1999. Making complex articulated agents dance: An analysis of control methods drawn from robotics, animation, and biology. *Autonomous Agents and Multi-Agent Systems* 2(1):23–44.

Matarić, M. J. 1995. Designing and understanding adaptive group behavior. *Adaptive Behavior* 4(1):50–81.

Matarić, M. J. 1997a. Behavior-based control: Examples from navigation, learning, and group behavior. *Journal of Experimental and Theoretical Artificial Intelligence* 9(2-3):323–336.

Matarić, M. J. 1997b. Reinforcement learning in the multi-robot domain. *Autonomous Robots* 4(1):73–83.

Matarić, M. J. 2000. Sensory-motor primitives as a basis for imitation: Linking perc eption to action and biology to robotics. In *Imitation in Animals and Artifacts*. The MIT Press.

Namco. 2000. Tekken tag tournament description. WWW Site. http://www.namco.com/ console /titles /tekken _tag _ tournament.html.

Popovic, Z., and Witkin, A. 1999. Physically based motion transformation. In *SIGGRAPH 1999*, 11–20.

Rizzolatti, G.; Gadiga, L.; Gallese, V.; and Fogassi, L. 1996. Premotor cortex and the recognition of motor actions. Cognitive Brain Research 3:131–141.

Weber, S. 2000. Simple human torso tracking from video. Technical Report IRIS-00-380, Institute for Robotics and Intelligent Systems, Universit y of Southern California.