Proceedings, IEEE International Conference on Robotics and Automation (ICRA-2002)
volume 1, pages 863-868, Washington DC, May 11-15, 2002

Parametric Primitives for Motor Representation and Control

R. Amit Maja J. Matarić
Computer Science Department
University of Southern California
Los Angeles, CA 90089-0781, USA
{amitr|mataric}@cs.usc.edu

Abstract

The use of motor primitives for the generation of complex movements is a relatively new and interesting idea for dimensionality reduction in robot control. We propose a framework in which adaptive primitives learn and represent synergetic arm movements. A simple and fixed set of postural and oscillatory primitives form the substrate through which all control is elicited. Higher level adaptive primitives interact and control the primitive substrate in order to handle complex movement sequences. We implemented this model on a simulated 20 DOF humanoid character with dynamics. We present results of the experiments involving the presentation and learning of synergetic arm movements.

Keywords: Primitives, motor control, learning, humanoids, self organizing maps

1 Introduction

One of the main challenges in robotics is to devise algorithms and methodologies for control of complex robots. The complexity of the robot's body is typically characterized by the number of its degrees of freedom. With the recent surge in both research and commercial interest in humanoids, robotics has had to face one of its greatest challenges in terms of control complexity [18]. The prohibitively large number of humanoid control parameters has led researchers to explore means of curtailing the motor control space to make this problem more tractable. Similar problems also arise in research involving the control of hyper-redundant systems [5].

One means of reducing the dimensionality of the motor control problem, motivated by neuroscience evidence, is through the use of *motor primitives* [14, 15, 17, 19] – a set of movement programs that form a vocabulary for the generation of a variety of complex movements. Gizster et al [9] demonstrated the presence of spinal force fields in the frog and rat that, when appropriately combined through supra-spinal inputs, result in the entire repertoire of observed movement. The presence of central pattern generators (CPGs) [6] in mammals also provides evidence for a basic set of motor programs that produce a variety of movements. These studies have been at the spinal level, while higher mammals tend to use cortical areas for movement generation. Preliminary studies [19] of human data provide some evidence toward an encoding of primitives.

Our past work has outlined the general primitives-based model for humanoid control and learning [14]. In this paper, we propose an implemented framework that fits within that model and enables learning, recognition, and execution of movements by a humanoid character. We use the notion of postural and oscillatory primitives as the lowest level of the control system. We then show how higher-level primitives can be learned by making use of this substrate. We experimentally validate these concepts on a complex humanoid simulation with dynamics. The experiments are devised in the context of imitation [2, 3, 16], wherein movements of a demonstrator are made available to a humanoid character which then learns to represent and execute them. It is important to note that our goal is not to reproduce the precise demonstrated trajectory (something that can be done through other, less general means), but rather to use imitation as a means of learning new motor skills.

The rest of the paper is organized as follows. Section 2 presents the overall framework giving the organization of our architecture and its components. Section 3 discusses how the components interact with one another to bring about the desired behavior. Section 4 details the mechanism of learning. Section 5 presents the experiments performed to validate the presented approach, and Section 6 provides the results of the experiments. Section 7 discusses related work, and places this research in its context. Section 8 concludes the paper.

2 Framework

The basic idea of primitives is that they serve as a compactly-represented vocabulary of motor programs that can be combined to produce a wide variety of motor behaviors. Our framework is hierarchically structured so that a set of basic primitives forms the substrate, and higher-level primitives are constructed from that substrate. The basic primitives are assumed to be "innate", i.e., built-in motor programs which are relatively simple, while higher-level primitives are learned and adaptive, so they can, incrementally through the hierarchical structure, grow in complexity. A block diagram illustrating the components of the framework is shown in Figure 1. As can be seen, actuators form the lowest level in the motor control hierarchy and are controlled by the basic primitives, which in turn are controlled by adaptive primitives. We describe each layer of the framework in turn.

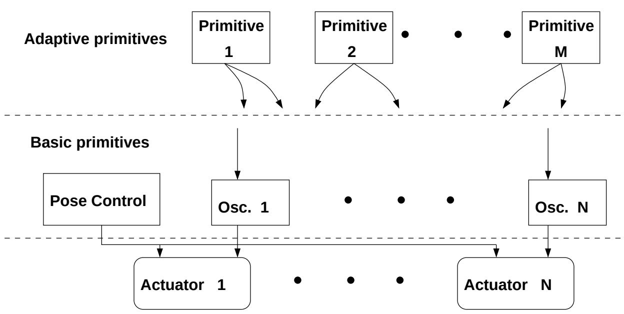

Figure 1: The framework of the control architecture

2.1 Basic primitives

In our framework, the basic primitives form the underlying substrate for both representation and control. They are the interface to the joint actuators. A certain level of competence is assumed in this layer, as these primitives are not adaptive but are well tuned to perform a specific type of control. In the implementation we present here, the basic primitives consist of two types: postural and oscillatory.

Postural primitives are built-in motor programs that orient joints so as to achieve a desired pose. These are similar to the discrete primitives of Schaal et al. [17]. The main purpose of these primitives is to provide an appropriate constant offset to the joint while the oscillatory primitive, which we discuss next, takes care of the oscillation. Postural primitives thus have a single parameter, the joint angle, for every joint in the robot. They can thus be succinctly represented as a function, F, to generate the torque vector, τ, corresponding to a set of joint angles, θ:

$$\tau = F(\theta) \tag{1}$$

We implement oscillatory primitives as harmonic oscillators associated with each joint. The amplitude, a, and frequency, f, of the oscillators are parameters of the primitives. Each such oscillatory primitive, when active, applies a time-varying torque, τ, to the joint actuator it controls. An oscillatory primitive is thus a function, g, such that:

$$\tau = g(a \sin(2\pi f t)) \tag{2}$$

In addition to being able to control the joints, both types of primitives are also capable of estimating the parameters from an observed movement demonstration. In our case, the incoming information to the system consists of the instantaneous joint angles of the demonstrator. This is possible through a variety of motion-capture mechanisms, ranging from external vision-based systems, to marker-based systems, to exoskeletons. We are particularly interested in learning from a single or a small set of tele-operated demonstrations, where joint information would be readily available.

2.2 Adaptive primitives

Adaptive primitives, at the next level of the hierarchy, do not have direct control over the joint actuators, but instead exert control through the basic primitives. The adaptive layer interacts with the basic primitives directly, and is partitioned so as to modularize control. Specifically, primitives are partitioned such that different groups are involved in the control of different joint combinations, thereby also reducing control complexity for each primitive. Ideally, these joint pairings are based on natural, frequent pairings of DOFs. In this particular implementation, they were chosen by the designer.

Adaptive primitives encode the parameters of the basic primitives. Each partition, which is a set of primitives for the same joint pairings, is a Self Organizing Map (SOM) [7, 11, 12]. SOMs are ordered two-dimensional arrays, called maps, of vector elements that encode features of the input space. Features are topographically organized over the surface of the map, i.e., adjacent elements encode similar features. The encoding in SOMs has two notable properties: 1) vector quantization and 2) dimensionality reduction using non-linear manifolds. The first property implies that the error in reconstruction of the input data is minimized. A corollary is that the number of elements representing regions of the input space is directly proportional to their probability density in the input data. The second property results in lower dimensional features of the input space that retain the topological properties. These properties make SOMs a useful tool for adaptive primitives. The primitives become robust to noisy inputs and represent movements based on their frequency of occurrence.

Each adaptive primitive gets its input in the form of the parameters of oscillation of the joints it has control over. For example, a primitive may have control over two DOFs of the right shoulder. In such a case, the primitive would encode frequency and amplitude of those joints for a certain movement, such as drawing a "figure 8."

3 Layer Interaction

Basic primitives begin estimating the parameters of a movement while a demonstration is in progress. Each of the oscillatory primitives begins to ascertain the amplitude and frequency of the movement pertaining to the joint it is associated with. Similarly, postural primitives begin to estimate the offset of the joints they are linked with. Basic primitives become active once they have an estimate of the parameters.

Postural primitives, in our implementation, use their offset estimate to appropriately orient the joints. They are not controlled by the adaptive primitives.

The estimated parameters from the oscillatory primitives are made available to the adaptive primitives that control them. The controlling adaptive primitives then provide the appropriate parameters of the oscillation, based on what they represent, and in turn the joints are actuated accordingly.

4 Learning of primitives

As stated, the adaptive layer is a collection of SOMs. Each SOM gets its input from a set of oscillatory primitives which is under its control. All units on a map (SOM) obtain the same input. The input to the units, in our model, is a vector that is composed of normalized frequencies and amplitudes obtained from the basic primitives. For example, if a particular map is in control of two oscillatory primitives with their respective frequency and amplitude parameters being $< f_1, a_1 >$ and $< f_2, a_2 >$, then the input vector, v, to each unit on the map is:

$$v =< || < f_1, f_2 > ||, || < a_1, a_2 > || > \tag{3}$$

As is the case with SOMs, the units are all initialized to small random vectors. Then, on the presentation of the input vector, a winner unit, w, is found from the units, u, belonging to the map, U.

$$w = u : \min_u \{|v - u|\} \forall u \in U \tag{4}$$

After the winner is found, it and its neighboring units, represented as w_n, within a radius, r, on the map are updated as:

$$w_n^{t+1} = w_n^t + \gamma(v - w_n^t) \forall w_n \in N(w, r), 0 < \gamma < 1 \tag{5}$$

where $N(w, r)$ is the neighborhood set that contains the units within distance r on the map from winner, w; γ is the learning rate.

When a winner is found, the vector it encodes is used to instantiate the parameters for execution of the oscillation. Specifically, assume that the estimated parameters of the oscillations of two oscillatory primitives controlled by the adaptive primitive are f_{e1}, a_{e1}, f_{e2}, and a_{e2}. Let the vector represented by the winner be $< f_{w1}, f_{w2}, a_{w1}, a_{w2} >$. Then, the actual instantiation of the joint oscillation parameters (f_1, a_1, f_2, a_2) for actuation is given by:

$$< f_1, f_2 >= (< f_{e1}, f_{e2} > \cdot < f_{w1}, f_{w2} >) < f_{w1}, f_{w2} > \tag{6}$$

$$< a_1, a_2 >= (< a_{e1}, a_{e2} > \cdot < a_{w1}, a_{w2} >) < a_{w1}, a_{w2} > \tag{7}$$

The parameter vector for execution of a movement is thus the projection of the parameter vector from the observation in the direction of the vector encoded by the winning unit.

Our representation makes use of normalized frequencies and amplitudes (Equation 3) to encode paired oscillations. This makes the encoding independent of the actual values of these parameters and instead captures the *structure* of the movement. For example, consider the movement involving drawing of a “figure 8”. This involves two DOFs of the shoulder, where one has the frequency of oscillation twice that of the other. In our representation, such structure is effectively captured. A unit on the map encoding a “figure 8” in effect represents this type of movement for any combination of frequencies and amplitudes. In our implementation we have restricted ourselves to frequencies and amplitudes of oscillations alone, and have not taken phase into account.

The representation we used provides robustness against noise and small variations or perturbations in movements. The fact that winning units are found (Equation 4) and used as a template for movement recognition and execution results in this robustness.

5 Experimental Validation

We have implemented the above framework and tested it within a physics-based simulation of a humanoid character. We provide a brief description of the simulator, the specific implementation of the model, and finally the test movements used in the experiments.

5.1 The physics-based simulator testbed

We developed a physics-based simulation using the Vortex real-time advanced physics libraries from Critical Mass Labs [13]. The humanoid has actuated joints from the waist up, totaling 20 active DOF. The lower body is not actuated, and the character is firmly attached to the ground at the feet. For all actuated joints in the humanoid simulation, we made use of the so-called RPRO joint in the Vortex library, which behaves like a PD servo loop, and allows specification of desired joint orientations. There are collision models for all objects in the environment, and gravity is also present. Graphical rendering is done with the SGI OpenInventor libraries. A snapshot of the simulator environment is shown in Figure 2.

Figure 2: A snapshot from the simulator

5.2 Experiments

For the experiments a total of six DOFs were considered, i.e., the three DOFs in each shoulder. There were thus six postural and six oscillatory primitives. The layer of adaptive primitives consisted of four SOMs, two for each hand, each with 5×5 units. Units in each of the SOMs had access to two DOF. The learning rate, γ, was set to a constant value of 0.01 and the neighborhood radius, r, was fixed at 2.

The experiments performed for validating the model involved demonstration of repeated arm movements. These included horizontal and vertical “figure 8”s , waving, and wading. We provide here the mathematical descriptions of the instantaneous joint angles for these movements.

Notation:

S_{AA}	Shoulder adduction-abduction angle
S_{FE}	Shoulder flexion-extension angle
S_{HR}	Shoulder humeral rotation angle
f	Frequency of oscillation
a, a_1, a_2	Amplitude of oscillation

- Vertical figure 8’s

$$S_{AA} = \frac{\pi}{2} + a\sin(2\pi f t)$$
$$S_{FE} = a\sin(2\pi \frac{f}{2} t)$$

- Horizontal figure 8’s

$$S_{AA} = \frac{\pi}{2} + a\sin(2\pi \frac{f}{2} t)$$
$$S_{FE} = a\sin(2\pi f t)$$

- Wading

$$S_{AA} = \frac{\pi}{2}$$
$$S_{FE} = a_1\sin(2\pi f t)$$
$$S_{HR} = a_2\sin(2\pi f t)$$

- Waving

$$S_{AA} = a_1\sin(2\pi f t)$$
$$S_{HR} = \frac{\pi}{2}$$

The data set contained 80 movements. These were uniformly taken from each of the four movements mentioned above. The choice of whether the movement was performed using the right hand, left hand, or both hands was also chosen at random while picking the data set. The frequency and amplitude parameters of the oscillations were varied to cover the entire range of values for each.

6 Results

The presentation of the movements resulted in learning of the adaptive primitives. The units are initially random, so the executed imitations movements are not the same as what was demonstrated. In the course of the learning process, the primitives begin to specialize and eventually begin to represent the observed movements, though in a more abstract manner. Figure 3 shows the trajectory of the hand of the demonstrator for successive tracing of “figure 8”. Figure 4 shows the trajectory of the hand of the imitator through the entire development. As can be seen, initially the imitation is rather random, corresponding to the early stages of the learning process, and then it converges to “figure 8” traces, after a primitive is formed.

Figure 5 gives the plot of the representation of frequencies in the two maps handling the left shoulder joints. In both maps, the data points only lie on a band that is part of a circle of unit radius. This is a result of the normalization process. Looking at the distribution of the points in the first plot of Figure 5, we see that the concentrations are high in the regions where one frequency is twice the other. This shows that the units in those regions learned to represent both the vertical and horizontal “figure 8”s. Among the rest of the points, we see those that are close to the area where one of the frequencies is very low and

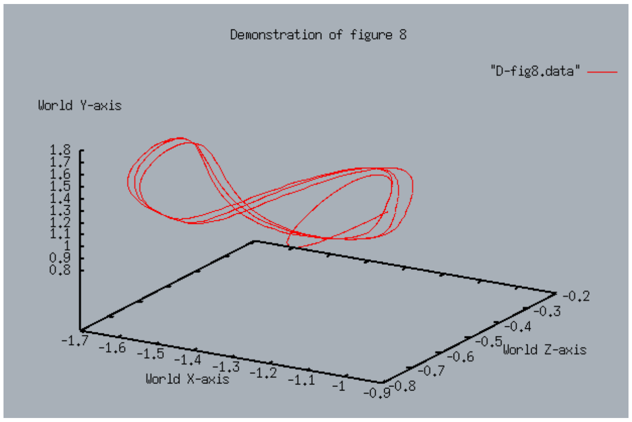

Figure 3: Trajectory of the demonstrator's hand for successive tracings of figure 8, starting from a rest position

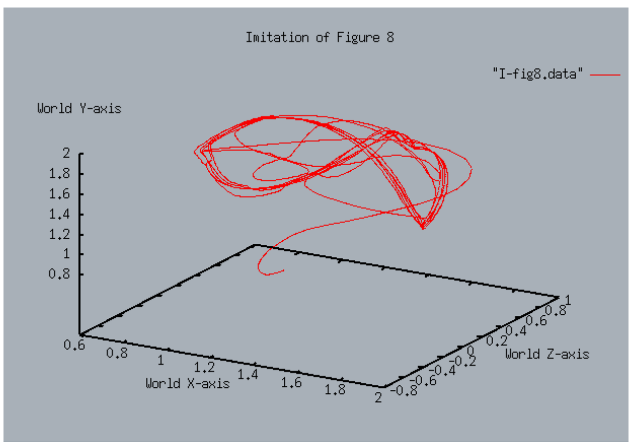

Figure 4: Trajectory of the imitator's hand throughout the learning process

the other very high. These correspond to the movements which involve one degree of freedom while not involving the other. In the second plot of Figure 5, the distributions are higher in regions where the two frequencies are close to each other. There are also points in the region where one of the frequencies is very high and the other very low. These represent the nature of the data set presented. The plots of the representations for the right shoulder are similar, and we do not show them here due to space constraints.

Amplitude representation plots are also not shown here due to space limitations. They have a more spread-out distribution due to the nature of the data used for learning. The previous interpretations are for the 2-D plots, separately for the frequencies and the amplitudes. However, the actual units represent 4-D information encoding both quantities in a single vector. Taken together, the primitives encode the movements that were presented in the demonstrations.

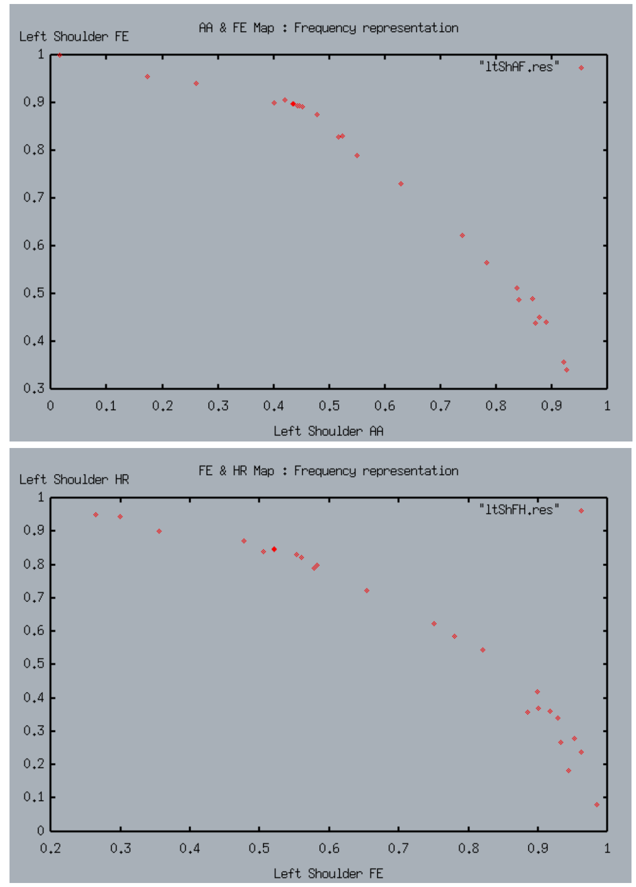

Figure 5: Frequency encodings of the primitives in the two maps of the left shoulder

7 Related work

Schaal et al [17] have demonstrated the use of oscillatory and discrete pattern generators, in combination, for various tasks like ball bouncing, drumming, "3-D" drawing patterns, etc. They used separate oscillators for each joint and a reference oscillator for coordination. Discrete movements were superimposed for positioning. We used a conceptually similar underlying substrate, but then built higher-level primitives that are capable of representing more complex movements.

Ijspeert et al [10] proposed using mixtures of nonlinear differential equations to represent movements for trajectory generation.

Bentivegna et al [1] have used the idea of primitives for motor learning. They applied the idea to learning to play air hockey and a marble maze in simulation and on a real robot.

Billard et al [2, 3, 4] used connectionist-based approaches to represent movements. They make use of a recurrent connectionist network that is able to learn oscillatory movements, and also discrete movements.

Fod et al [8] automatically derived primitives through an off-line process of segmentation and application of principal component analysis to motion-capture human arm movement data.

8 Conclusion

We presented a framework for representating movements using parametric primitives. The parametric nature allowed for an abstract representation of movements independent of speed and size. The hierarchical nature of the framework made it possible to learn novel complex movements.

So far, we have concentrated on repetitive movements, and their parametric representation. Future work will address the issues of representing discrete movements within the same framework. We will also address more complex movements involving higher DOF.

Acknowledgments

This work was supported by DARPA Grant DABT63-99-1-0015 under the Mobile Autonomous Robot Software (MARS) program.

References

[1] D. Bentivegna and C. G. Atkeson. Using primitives in learning from observation. In *First IEEE-RAS International Conference on Humanoid Robots (Humanoids 2000)*, MIT, Cambridge, MA, 2000.

[2] A. Billard. Learning motor skills by imitation: a biologically inspired robotic model. In *Cybernetics and Systems*, volume 32:1-2, pages 155–193, 2000.

[3] A. Billard and M. J Matarić. A biologically inspired robotic model for learning by imitation. In *Proceedings, The Fourth International Conference on Autonomous Agents (Agents 2000)*, pages 372–380, Barcelona, Spain, June 2000.

[4] A. Billard and M. J. Matarić. Learning human arm movements by imitation: Evaluation of biologically-inspired connectionist architecture. In *Robotics and Autonomous Systems*, volume 37:2-3, pages 145–160, November 2001.

[5] G. S. Chirikjian and J. W. Burdick. The kinematics of hyper-redundant robotic locomotion. In *IEEE Transactions on Robotics and Automation*, volume 11(6), pages 781–793, December 1995.

[6] J. Duysens and W. A. A. van de Crommert. Neural control of locomotion; part 1: The central pattern generator from cats to humans. gait and posture. *Gait and Posture*, 7(2):131–141, 1998.

[7] E. Erwin, K. Obermayer, and K. Shulten. Self-organizing maps : ordering, convergence properties and energy functions. In *Biological Cybernetics*, volume 65, pages 47–55, 1992.

[8] A. Fod, M. J. Matarić, and O. C. Jenkins. Automated derivation of primitives for movement classification. In *Autonomous Robots*, volume 12(1), pages 39–54, Jan 2002.

[9] S. F. Giszter, F. A. Mussa-Ivaldi, and E. Bizzi. Convergent force fields organized in the frog's spinal cord. *Journal of Neuroscience*, 13(2):467–491, 1993.

[10] A. J. Ijspeert, J. Nakanishi, and S. Schaal. Trajectory formation for imitation with nonlinear dynamical systems. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2001)*, pages 752–757, Maui, Hawaii, 2001.

[11] T. Kohonen. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43:59–69, 1982.

[12] T. Kohonen. The Self-Organizing Map. In *New Concepts in Computer Science: Proc. Symp. in Honour of Jean-Claude Simon*, pages 181–190, Paris, France, 1990.

[13] Critical Mass Labs. http://www.cm-labs.com.

[14] M. J. Matarić. Visuo-motor primitives as a basis for learning by imitation. In Kerstin Dautenhahn and Chrystopher Nehaniv, editors, *Imitation in Animals and Artifacts*. MIT Press, 2001.

[15] F. A. Mussa-Ivaldi, S. F Giszter, and E. Bizzi. Linear combination of primitives in vertebrate motor control. *Proc. Nat. Acad. Sci. USA*, 91:7534–7538, 1994.

[16] S. Schaal. Is imitation learning the route to humanoid robots? *Trends in Cognitive Sciences*, 3(6):233–242, 1999.

[17] S. Schaal and D. Sternad. Programmable pattern generators. In *3rd International Conference on Computational Intelligence in Neuroscience*, pages 48–51, Research Triangle Park, NC, 1998.

[18] M. L. Swinson and D. J. Bruemmer, editors. *IEEE Intelligent Systems*, volume 15(4), Jul 2000.

[19] K. A. Thoroughman and R. Shadmehr. Learning of action through combination of motor primitives. *Nature*, 407:742–747, 2000.