MOVEMENT PRIMITIVES AND PRINCIPAL COMPONENT ANALYSIS

Frank C. Park and Kyoosang Jo
School of Mechanical and Aerospace Engineering
Seoul National University
fcp@plaza.snu.ac.kr

Abstract Robot trajectory generation based on the optimization of physical criteria is by now a well-established means of motion programming, but the computational burden still remains a formidable challenge for real-time implementation. Motivated by results from the biological motor control literature, we propose a method for constructing movement primitives based on principal component analysis (PCA) of a set of reference motions. The movement primitives provide an efficient and task-specific set of basis functions for the representation of general motions; by using the basis functions constructed from a set of reference motions associated with a given task, sub-optimal motions that closely resemble the reference motions can be generated rapidly. Experimental results obtained from motion capture data of human arm motions are offered to support the merits of our PCA-based approach.

Keywords: Movement primitive, trajectory generation, motion optimization, principal component analysis.

1. Introduction

The difficulty in generating movements for robots and other complex articulated systems can, we believe, ultimately be traced to the high dimensionality of the problem. The kinematic structure usually has redundant degrees of freedom, a desired task can usually be achieved by infinitely many end-effector trajectories, and both the admissible joint and task configuration spaces exhibit some nonlinearity and irregularity as a result of physical and mechanical constraints.

Yet in spite of these difficulties, humans and animals are able to generate movements instantaneously and with seemingly little effort. To endow robots with similar movement generation capabilities, therefore, it is natural to look to animal motor control for clues. While a widely accepted comprehensive theory of how animals generate movements has yet to emerge, two phenomena seem to be consistently observed in the biological movement literature: (i) the way in which the degrees of freedom appear to be grouped for planning and control purposes, and (ii)

evidence suggesting the existence of movement primitives, and how they are concatenated to form more complex motions.

There is also strong evidence that some type of optimization is taking place, usually with respect to some physical criterion. In many cases the motions appear to become smoother and richer with practice, as more degrees of freedom are being utilized to enhance performance; this is readily evident, for example, when watching someone learning how to swing a golf club. As movement learning occurs we also see a transition from closed-loop to open-loop control. In the golf swing example, while initially there is a heavy reliance on visual feedback (to ensure proper posture, etc.), with practice one relies less and less on such external sensory feedback as the body "learns" the swing.

Taking these lessons and applying them to robots, we are faced with the following practical issues: (i) how do we select the movement primitives that constitute the vocabulary of motions? (ii) how do we optimize the movements in real-time? This paper offers a framework in which to simultaneously address these two issues. First, movement primitives are defined according to the task, and stored not as a single trajectory or curve, but as a finite family of basis functions. Given a specific task description, the objective is to find the optimal linear combination of basis functions to generate the required movement.

The distinguishing feature of our approach is that the basis functions are determined from principal component analysis (PCA). This has two advantages: because the optimized motions are from a restricted class, they are similar to the benchmark motions used to create the basis functions. This can be useful, for example, when we desire to create human-like (and not necessarily physically optimal) motions rapidly for robots and animated characters. Secondly, by restricting the number of basis functions associated with any given task, the optimal motions can be obtained efficiently, possibly even in real-time.

The relevant literature on biological motor control and movement primitives is extensive (see, *e.g.*, the seminal work of Bernstein, 1967), as are recent approaches to robot movement generation and control inspired by the biological paradigm, *e.g.*, Muss-Ivaldi, 1997, Schaal et al., 2003, Fod et al., 2002. The interpretation of movement primitives as a set of basis functions specific to a given task, combined with the application of principal component analysis to generate these primitives, has, to our knowledge, not been addresssed in the literature.

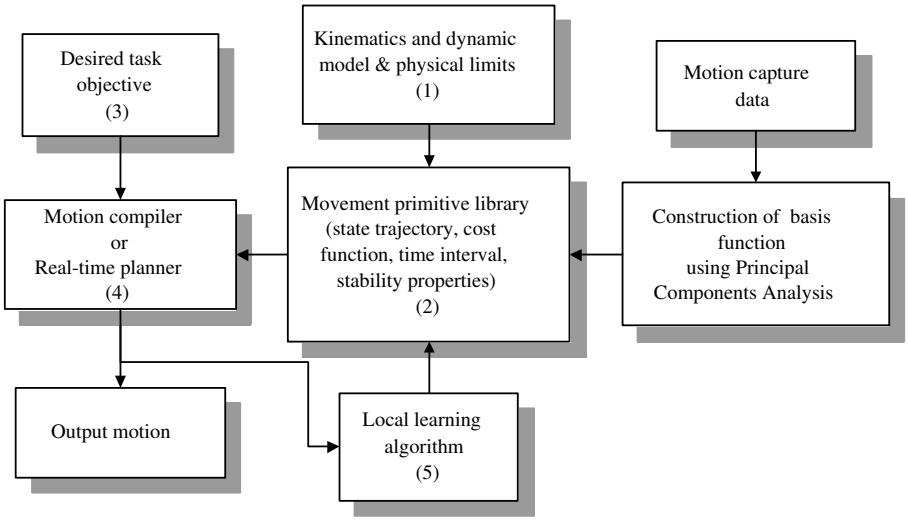

Figure 1. Movement Generation Framework

2. Movement Generation Framework

Our proposed high-level movement coordination and learning framework, in particular how we intend to characterize the problem and organize the data, is shown in Fig. 1. The kinematic and dynamic features of the robot, and all the data needed to define a reasonably good dynamic model, are first defined (Block 1). With the desired class of goal behaviors in mind, we then create a rich set of movement primitives (Block 2) that will form the basis for the motion compiler. Each movement primitive consists of a finite set of basis functions, and will ultimately be paired with an associated control law and its stability properties. A desired task goal (Block 3) is then input to the planner (Block 4).

The planner chooses the motion sequence that best achieves the desired task. This is done at two levels. At a lower level, given the set of basis functions corresponding to a given movement primitive, local optimization is performed to determine the optimal set of weights; the objective function to be used in the optimization is given by the planner. At a higher level, in the event that the task calls for a more complex sequence of movements, the planner first selects and concatenates a set of movement primitives. The concatenation may be done in real-time if optimal behavior is not required. If repeated (on-line) trials are possible, or if planning may be conducted off-line, local learning is used (Block 5) to improve the resulting motion. The learning block refines the entire

movement sequence by scaling and warping the selected movement primitives in order to best achieve the desired goal. The refined movement itself may be be used as a new movement primitive if desired.

The motion compiler will convert the high level goal to an objective function to be used in a nonlinear constrained parameter optimization program; in effect, it is the motion compiler that performs the local optimization to determine the optimal set of weights for each movement primitive.

The focus of this paper is on the selection of basis functions corresponding to a movement primitive, using principal component analysis of motion capture data, and on the local optimization performed within the motion compiler. By relying on a set of differential geometric formulations for multibody dynamics Park et al., 1995, Kim et al., 1999, we are able to determine analytic, or exact, gradients of the objective function and the constraints.

3. Principal Component Analysis

Principal Component Analysis (PCA) is a statistical methodology based on singular value decompositions for identifying patterns in, and determining compressed approximate representations of, a set of given data (see, *e.g.*, Jackson, 1993). For our particular movement primitive application, we focus exclusively on three-dimensional arm motions. We first collect motion capture data for a wide array of arm motions corresponding to some task (*e.g.*, swinging a tennis racket, lifting a heavy weight); This usually entails attaching markers to reference points on the arm, and solving the inverse kinematics equations to obtain a set of joint angle trajectories.

To illustrate the general PCA procedure, consider a p-dimensional vector $\vec{X} = (X_1, X_2, \cdots, X_p)^T$, and suppose we have vector time series data for this vector $\vec{X}$. Let the sample mean and covariance matrix associated with this vector time series be respectively given by μ and Σ, respectively. Represent the eigenvalue-normalized eigenvector pairs of the covariance matrix by

$$(\lambda_1, \vec{e}_1), (\lambda_2, \vec{e}_2), \cdots, (\lambda_p, \vec{e}_p)$$

$$\begin{aligned} \text{where} \quad & \vec{e}_i^T \vec{e}_i = 1, \quad \vec{e}_i^T \vec{e}_k = 0, \quad i \neq k \\ & \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p > 0 \end{aligned} \tag{1}$$

In the event that Σ is positive definite, the eigenvectors serve as a set of basis vectors; the relative magnitude of the eigenvalues implies how dominant the corresponding eigenvectors are in representing the time series

data. These eigenvectors represent the principal components asssociated with the given data, and have the following properties;

- i^{th} principal component:

$$Y_i = \vec{e}_i^T \vec{X} = e_{1i}X_1 + e_{2i}X_2 + \cdots + e_{pi}X_p, \quad i = 1, 2, \cdots, p \tag{2}$$

- variance and covariance of i^{th} principal component:

$$\begin{array}{l} Var(Y_i) = \lambda_i, \quad i = 1, 2, \cdots, p \\ Cov(Y_i, Y_k) = 0, \quad i \neq k \end{array} \tag{3}$$

- variance equality relationship:

$$\sum_{i=1}^{p} Var(X_i) = \sum_{i=1}^{p} Var(Y_i) = \lambda_1 + \lambda_2 + \cdots + \lambda_p \tag{4}$$

- normalized variance equality relationship:

$$\frac{Var(Y_k)}{\sum_{i=1}^{p} Var(X_i)} = \frac{\lambda_k}{\lambda_1 + \lambda_2 + \cdots + \lambda_p}, \quad k = 1, 2, \cdots, p \tag{5}$$

By selecting the principal components with the highest weights, we can use these comoponents as basis functions with which the construct more general trajectories. Of course, this set of basis functions does not constitute a basis in the strict linear algebraic sense, since it can only represent a finite dimensional subset of all possible trajectories. However, the hope is that by capturing the most essential features of a given set of captured motions through its principal components, one can come up with reasonable approximations to this class of motions with great efficiency and speed.

4. Case Study

In this section we explore an example of generating arm raising motions based on PCA of human arm motions obtained from motion capture data. We model the kinematic structure of the human arm using 3 links (upper arm, lower arm, chest) and 2 joints (shoulder, elbow). The shoulder joint is modeled as a three dof spherical joint (obtained by serially arranging three revolute joints in orthogonal fashion), while the elbow joint is a single dof revolute joint. The inertial parameters that we use for the human arm model are shown shown in Table 1.

We first obtain motion capture data for various typical arm motions, such as lifting, pointing, pulling, pushing, rotating, etc.. We then calculate the joint angle trajectories for each motion from the inverse kinematics equations. For our experiment we obtain a total of 52 joint angle

Table 1. The mass and the moment of inertia of the human arm model

Link	*Mass(kg)*	*Moment of inertia(kgm^2)*
Chest	29.27	(0.73, 0.63, 0.32)
Upper arm	2.97	(0.025, 0.025, 0.005)
Lower arm	1.21	(0.005, 0.0054, 0.0012)

trajectory samples. Following the procedure outlined in the previous section, we then obtain the principal components corresponding to our motion data. We discover that the first two principal components explain more than 90% of the data; Table 2 shows the weight of each variance component according to each joint, while Figure 2 shows the graph of each joint's principal components, represented as a joint trajectory in time. For our simulation example we determine the minimum-effort

Table 2. Variance weight according to each joint

Joint	*Shoulder1(%)*	*Shoulder2(%)*	*Shoulder3(%)*	*Elbow(%)*
PC1	92.49	82.37	85.63	74.36
PC2	6.39	12.65	11.90	23.50
PC3	0.82	3.69	1.90	1.65
PC4	0.13	0.81	0.29	0.28
Sum	99.83	99.52	99.72	99.79

control motions for specified beginning and end points in the joint space Martin and Bobrow, 1995, Kim et al., 1999. The trajectories are represented as a linear combination of the basis functions obtained via PCA, approximated by fourth-order polynomials. The cost function is defined as:

$$J = \frac{1}{2}\int_0^{t_f} \|\tau(t)\|^2 dt \tag{6}$$

where τ denotes the torque vector and t_f the final time. The dynamics computations were performed using the geometric formulation described in Park et al., 1995, Kim et al., 1999; The optimizations were performed in Matlab (ver. 6.5) on a 1.8 GHz Pentium 4, making use of the analytic gradient information provided by our geometric dynamics formulation.

To benchmark the gains in computational efficiency afforded by the PCA approach, we also perform the motion optimization using B-splines with 6 control points (as opposed to our basis trajectories obtained from PCA). Our simulation results indicate that, with respect to time cost,

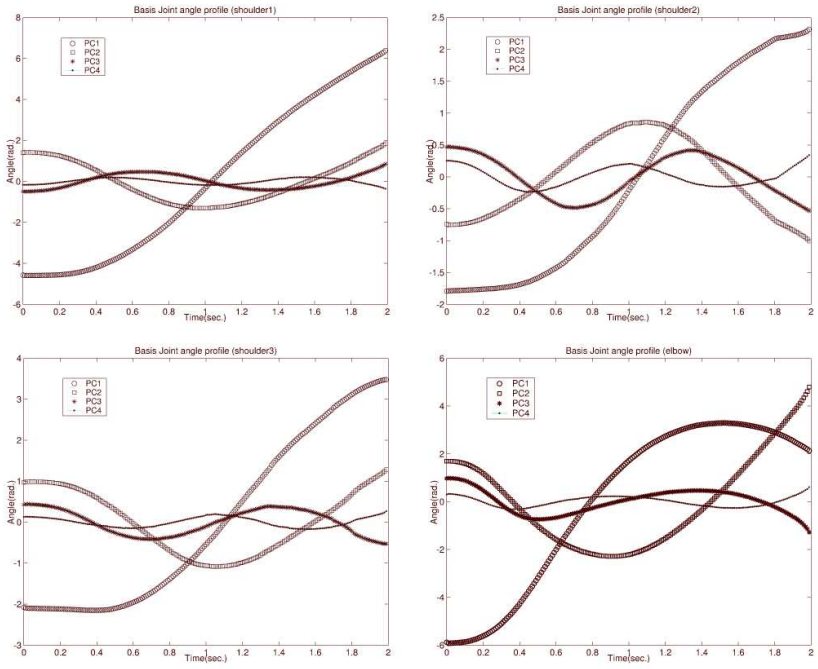

Figure 2. Graph of Each joint' PC

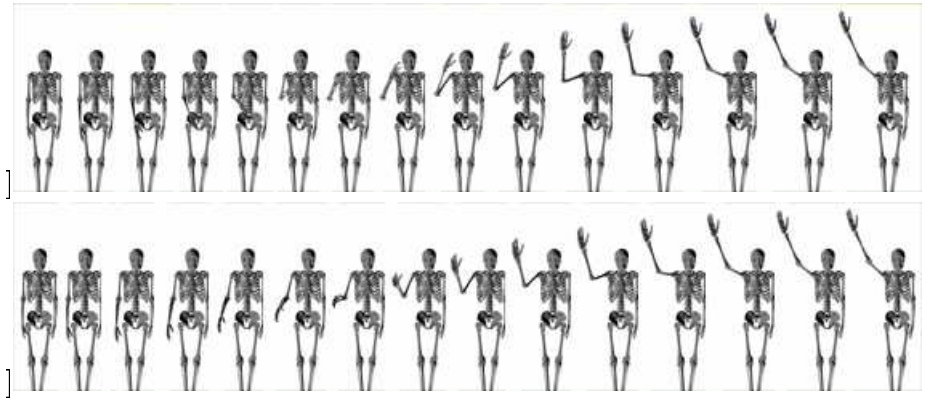

Figure 3. Optimal 3D arm-raising motion: front fiew (top: using B-splines, bottom: using PCA)

the PCA-based approach is more efficient than the B-spline approach (Table 3). Not surprisingly, the B-spline approach leads to smaller values of the cost function, indicating that the PCA based approach leads to suboptimal solutions.

This is not necessarily a disadvantage, and in human animation and computer graphics applications, can in fact be an important advantage. Figures 3 and 4 offer a side-by-side comparison of the optimal motions obtained using PCA and B-splines. In the B-spline case, the arm is first rotated away from the body (front view), and a retraction of the arm behind the body can be seen (side view). This is quite unnatural compared to human arm motions. On the other hand the motion obtained using PCA can be seen to be resemble much more closely to human arm motions; more natural motions can be generated with much less effort, an important benefit from the perspective of realistic human motion animation.

Table 3. Arm-raising motion: performance comparison

Type	*B-spline*	*PCA*
objective function value	5.19594	9.58805
optimization time(sec.)	148.9750	96.5090

5. Conclusions

In this paper we have proposed a means of constructing movement primitives based on principal component analysis (PCA) of a set of reference motions. Inspired by the biological movement control literature, which suggests that human and animal motions are organized into a vocabulary of primitive movements, with mechanisms available for local

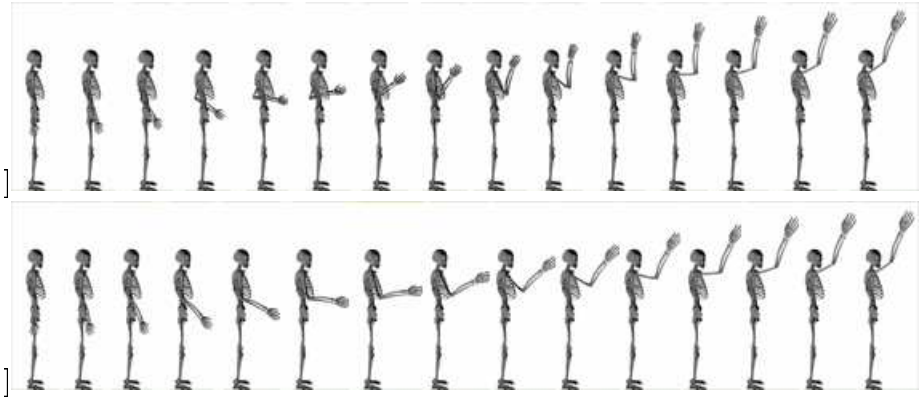

Figure 4. Optimal 3D arm-raising motion: side view (top: using B-splines, bottom: usingPCA)

learning and optimization, we adopt this framework for robot trajectory generation.

The movement primitives provide an efficient and task-specific set of basis functions for the representation of general motions; by using the basis functions constructed from a set of reference motions associated with a given task, sub-optimal motions that closely resemble the reference motions can be generated rapidly. Experimental results were performed using motion capture data of human arm motions, and compared to straightforward minimum torque optimization using standard B-splines. The results suggest that not only can the computational burden of movement generation be significantly reduced, but that much more natural-looking arm motions can be generated.

Our approach can in some sense be viewed as exploiting the classic computational tradeoff between memory and computing resources. By storing the movement primitives as basis functions, we significantly reduce the computational load involved in the optimization, while at the same time limiting the class of generated motions to be close to the set of reference trajectories used in the basis function selection. In future work we hope to examine a richer class of tasks, and extend the framework to more complex kinematic structures.

Acknowledgements

This research was supported by the National Research Laboratory for CAD/CAM at Seoul National University.

References

Bernstein, N.A. (1967), *The Coordination and Regulation of Movements.* London: Pergamon Press.

Fod, A., Mataric, M., Jenkins, O.C. (2002), "Automated derivation of primitives for movement classification," *Autonomous Robots*, vol. 12, no. 1, pp. 39–54.

Jackson, J.E. (1993), *A User's Guide to Principal Components.* New York: John Wiley and Sons.

Kim, J., Baek, J., Park, F.C. (1999), "Newton-type algorithms for robot motion optimization," *Proc. 1999 IEEE Int. Conf. Intelligent Robots and Systems (IROS)*, vol. 3, pp. 1842–1847.

Martin, B.J., Bobrow, J.E. (1995), Determination of minimum-effort motions for general open chains, *Pro. IEEE Int. Conf. Robotics and Automation*, pp. 1160–1165.

Murray, R.M., Li, Z., and Sastry, S.S. (1993), *A Mathematical Introduction to Robotic Manipulation.* Boca-Raton: CRC Press.

Mussa-Ivaldi, F.A. (1997), "Nonlinear force fields: a distributed system of control primitives for representing and learning movements," *Proc. IEEE Int. Symp. Computational Intelligence in Robotics and Automation*, pp. 84–90.

Park, F.C., Bobrow, J.E., Ploen, S.R. (1995), A Lie group formulation of robot dynamics, *Int. J. Robotics Research*, vol. 14, no. 6, pp. 609–618.

Schaal, S., Peters, J., Nakanishi, J., Ijspeert, A. (2003), Control, planning, learning, and imitation with dynamic movement primitives, in Workshop on Bilateral Paradigms on Humans and Humanoids, *IEEE International Conference on Intelligent Robots and Systems (IROS)*.