Proceedings of the 2001 IEEE/RSJ
International Conference on Intelligent Robots and Systems
Maui, Hawaii, USA, Oct. 29 - Nov. 03, 2001

Trajectory Formation for Imitation with Nonlinear Dynamical Systems

Auke Jan Ijspeert[1,2], Jun Nakanishi[1] and Stefan Schaal[1,2]

[1]Computational Learning and Motor Control Laboratory
University of Southern California, Los Angeles, CA 90089-2520, USA
[2]Kawato Dynamic Brain project, ATR, Kyoto 619-0288, Japan
Email: ijspeert@usc.edu, nakanisi@usc.edu, sschaal@usc.edu

Abstract

This article explores a new approach to learning by imitation and trajectory formation by representing movements as mixtures of nonlinear differential equations with well-defined attractor dynamics. An observed movement is approximated by finding a best fit of the mixture model to its data by a recursive least squares regression technique. In contrast to non-autonomous movement representations like splines, the resultant movement plan remains an autonomous set of nonlinear differential equations that forms a control policy which is robust to strong external perturbations and that can be modified by additional perceptual variables. This movement policy remains the same for a given target, regardless of the initial conditions, and can easily be re-used for new targets.

We evaluate the trajectory formation system (TFS) in the context of a humanoid robot simulation that is part of the Virtual Trainer (VT) project, which aims at supervising rehabilitation exercises in stroke-patients. A typical rehabilitation exercise was collected with a Sarcos Sensuit, a device to record joint angular movement from human subjects, and approximated and reproduced with our imitation techniques. Our results demonstrate that multi-joint human movements can be encoded successfully, and that this system allows robust modifications of the movement policy through external variables.

1 Introduction

An important issue in humanoid robotics, and in learning by imitation in particular, is the question of how to encode desired trajectories to be performed by the robot. Various approaches have been suggested in the literature, ranging from memorizing the entire trajectory at the sampling rate of the control servo [1], using spline-based methods [2], using optimization criteria [3], or employing lookup tables and neural networks that represent global control policies [4]. In general, there seems to be consensus that among the most important desirable properties of movement encoding are: 1) the ease of representing and learning a goal trajectory, 2) compactness of the representation, 3) robustness against perturbations and changes in a dynamic environment, 4) ease of re-use for related tasks and easy modification for new tasks, and 5) ease of categorization for movement recognition. When examining standard approaches of movement planning and representation against this check list, it becomes obvious that no approach exists that accomplishes all these goals. For instance, memorized trajectories are easy to learn, but are hard to re-use for new tasks and not robust towards significant changes of the environment. Spline-based approaches have a more compact representation, but otherwise share most of the properties of memorized trajectories. Optimization approaches are computationally expensive and cannot re-plan rapidly when the environment changes, and neural network based control policies are very hard to learn for even moderately dimensional systems.

In this article, we explore an alternative to these approaches which can fulfill all the desired characteristics above. The idea is to encode desired trajectories, or more precisely complete control policies, in terms of a mixture of pattern generators built from simple nonlinear autonomous dynamical systems. In contrast to other approaches in the literature, the dynamical systems encode *desired* trajectories, not motor commands, such that an additional movement execution stage by means of a standard controller (e.g., inverse dynamics controller in our work) is needed. This strategy, however, does not prevent us from modifying the trajectory plans on-line through external variables, as will be demonstrated later.

Our current work is restricted to discrete movements, i.e., movement with a unique point attractor. Complex movement can be represented by a sequence of such mixture model movement primitives. The transition from one movement segment to another could be state-triggered or time-indexed, depending on the task to be accomplished – this is similar to via-point approaches suggested by others [2]. Given that the component dynamical systems are globally stable, our suggested weighted superposition will be globally stable, too.

We apply this trajectory formation system (TFS) to a task involving the imitation of human movements by a humanoid simulation. This experiment is part of a project in rehabilitation robotics —the Virtual Trainer project — which aims at using humanoid rigid body simulations and humanoid robots for supervising rehabilitation exercises in stroke-patients. This article presents how trajectories of typical rehabilitation exercises recorded from human subjects can be reproduced by the simulator using the trajectory formation system, and that this kind of encoding

0-7803-6612-3/01/$10.00©2001 IEEE

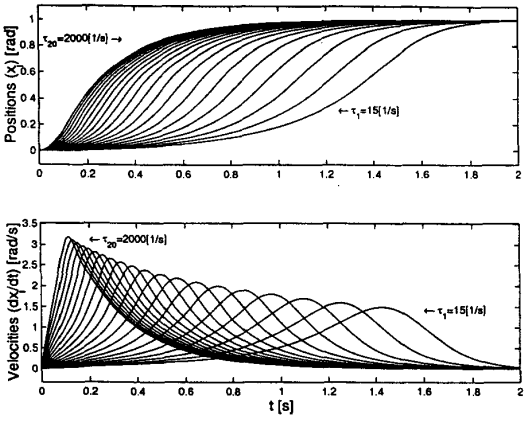

Figure 1: Set of primitives for one DOF. Top: trajectories, Bottom: velocity profiles. The following parameters are used throughout the paper: $N = 20$, $\gamma = \varepsilon = 0.005$, $\beta = 4.0[1/s]$, $\mu = 3.0[1/s]$, and $\tau_i = [15, 19.4, 25.1, \ldots, 2000][1/s]$. G was here set to 1.0 and no perturbation was fed back into the system, i.e. $\theta_R = \theta$.

fulfills all the desired characteristics enumerated above.

In the next sections, we first present our trajectory formation system and its extensions to imitation in Section 2. Then we briefly describe the Virtual Trainer project and the requirements it places on the trajectory formation system (Section 3.1), our humanoid simulation (Section 3.2), and the method for recording human movements (Section 3.3). Results of multi-joint fitting are finally presented in Section 3.4. Our approach is discussed as well as compared with alternative methods in Section 4.

2 Dynamical systems for trajectory formation

In earlier work, we have developed nonlinear dynamical systems for discrete and rhythmic movements in a humanoid robot[5, 6]. Trajectories for each degree of freedom were created by superimposing two dynamical systems, one for discrete (point-attracting) movement, the other for rhythmic (limit-cycle) movement. The system could produce robust trajectory planning (e.g. perturbations did not prevent from reaching specific targets, and did not disrupt smooth and stable movement execution) by numerically integrating the differential equations of the dynamical systems, and was successfully used to control juggling and drumming tasks. The applicability of this system, however, was limited to simple reaching movement as only monotonic movement could be realized with symmetric bell-shaped velocity profiles. For instance, creating a tennis fore-hand swing would not be feasible with this encoding. In this paper, we generalize movement planning with dynamical systems by developing a trajectory formation system based on mixtures of nonlinear dynamical systems, which allow accurate reproduction of almost arbitrary human movements.

Our approach presented in this paper is inspired by the idea of basis function and mixture models in statistical learning [7]. Instead of representing a function with one complicated representation, it is often easier to accomplish the same goal with a representation of superimposed simple component systems. Spline-based trajectory encodings are essentially such mixture systems, except that they are non-autonomous and do not form a stable control policy. Thus, instead of splines, we would like to use autonomous dynamical system as components with well-defined point attractors. The weighted superposition of these component system, or movement primitives, can form very complex attractor landscapes on the way to the equilibrium point. Complex movements that have intermediate starts and stops need to be segmented at these "via points" such that every segment is fitted separately by a mixture of movement primitives.

As a first approach, we created the following family of N primitives for encoding the trajectory of one DOF:

$$\dot{v}_i = \tau_i \cdot \left(\gamma + \frac{x_i^2}{G^2 + \varepsilon}\right) \cdot (-v_i + \beta(G - x_i) + \mu(\theta_R - \theta)) \quad (1)$$

$$\dot{x}_i = v_i \quad (2)$$

$$\theta = \sum_i^N w_i x_i \quad (3)$$

$$\dot{\theta} = \sum_i^N w_i v_i \qquad \text{with} \qquad \sum_i^N w_i = 1 \quad (4)$$

where θ and $\dot{\theta}$ are the desired angular position and velocity to be performed by the DOF, and G is the desired displacement (or goal position). θ_R is the actual joint angle realized by the DOF. The state variables x_i and v_i are the positions and velocities of each individual primitive i. Primitives are determined by four strictly positive parameters —two time constants τ_i and β, and a parameter γ which determines the width of the velocity bell shape— one positive gain parameter μ, and a small constant ε. The contribution of each primitive to the movement is determined by the weights w_i.

By construction, the position x_i of each primitive converges to the goal with a bell-shaped velocity profile (Figure 1). Note that, as the weights w_i are not restricted to be positive, the $\theta(t)$ trajectories do not need to be monotonic. Perturbations to the system which prevent the real angle θ_R to be equal to the desired angle θ (e.g. due to external forces applied to the DOF) are fed back into the system by the $\mu(\theta_R - \theta)$ term in equation 2, effectively slowing down or terminating the planning dynamics of the mixture model.

In our experiments, the trajectory formation system (TFS) will be based on a set of $N = 20$ primitives per DOF in which the parameters β, γ and ε are fixed and identical for all primitives, while τ_i varies according to $\tau_i = \tau_{min}(\tau_{max}/\tau_{min})^{\frac{i-1}{N-1}}$ to ensure more or less equal spacing between the maxima of the velocities (Figure 1) .

Despite its simplicity, the mixture of primitives possesses a variety of appealing properties:

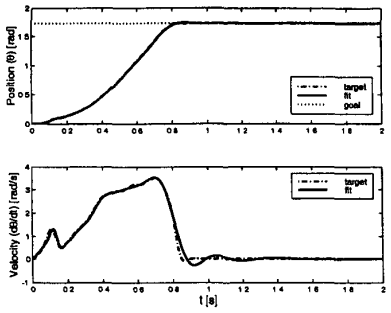

Figure 2: Example of fitting a desired trajectory using recursive least squares regression. Top: position trajectory, Bottom: velocity trajectory.

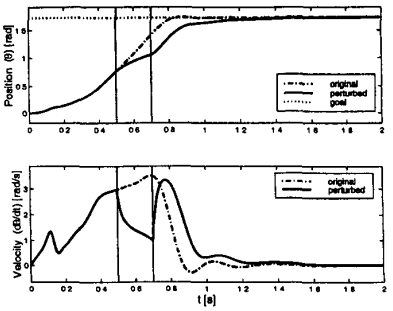

Figure 3: Recovery from a perturbation of the position (θ_R is forced to be 0.0 from t=0.5s to t=0.7s, while it is set to θ the rest of the time). Top: position trajectory, Bottom: velocity trajectory.

1. the velocity profiles of each individual primitive is smooth and has a strong resemblance to the bell-shaped profiles observed in human movements – this will be beneficial for smooth trajectory generation,
2. the mixture of primitives can be fitted to a desired trajectory using an incremental least squares regression on line assuming that the goal is known (see below),
3. individual primitives, as well as linear combinations of them, are stable systems which all converge to a unique point attractor corresponding to the goal when only transient perturbations are applied to them (see next section),
4. the complete system remains an autonomous dynamical system, i.e., there is no explicitly time dependency,
5. a particular movement can be characterized by the weight vector of the primitives—this has the potential to allow movement recognition based on these parameters.

2.1 Stability of the mixture model

This section presents a stability analysis of the proposed dynamical primitives for the case $\theta_R = \theta$.

By setting $\dot{x}_i = \dot{v}_i = 0$ and solving the equations for x_i and v_i, we find that each component of the mixture model has a unique equilibrium point at $(x_i, v_i) = (G, 0)$ [1]. To see that this equilibrium point is globally asymptotically stable, we shift it to the origin by the change of variables $\bar{x}_i = x_i - G$.

Consider a Lyapunov function candidate

$$V(\bar{x}_i, \bar{v}_i) = \frac{1}{2}\bar{v}_i^2 + \frac{\beta\tau_i}{12(G^2+\varepsilon)} \cdot \left(3\bar{x}_i^2 + 8G\bar{x}_i + 6\gamma\varepsilon + 6G^2(\gamma+1)\right)\bar{x}_i^2 \qquad (5)$$

Note that (5) is indeed positive definite for $\gamma, \varepsilon > 0$ since

$$3\bar{x}_i^2 + 8G\bar{x}_i + 6\gamma\varepsilon + 6G^2(\gamma+1) = 3(\bar{x}_i + \frac{4}{3}G)^2 + \frac{2}{3}G^2 + 6\gamma\varepsilon + 6G^2\gamma > 0.$$

[1] The equilibrium is shifted to $\bar{x}_i = G + \mu(\theta_R - \bar{\theta})/\beta$ and $\bar{v}_i = 0$ if a constant perturbation is applied to the DOF.

Then, $\dot{V}(\bar{x}_i, \bar{v}_i)$ is given by

$$\dot{V}(\bar{x}_i, \bar{v}_i) = -\tau_i \left(\gamma + \frac{(\bar{x}_i + G)^2}{G^2 + \varepsilon}\right)\bar{v}_i^2 \leq 0. \qquad (6)$$

Since

$$\dot{V}(\bar{x}_i, \bar{v}_i) = 0 \;\Rightarrow\; \bar{v}_i = 0 \;\Rightarrow\; \bar{x}_i = 0, \qquad (7)$$

the origin is asymptotically stable by applying LaSalle's invariance theorem. Moreover, since $V(\bar{x}_i, \bar{v}_i)$ is radially unbounded, the origin is globally asymptotically stable. Thus, the equilibrium point of each primitive dynamics, $(x_i, v_i) = (G, 0)$, is globally asymptotically stable. Furthermore, the mixture of these dynamical systems has a globally asymptotically stable attractor point at $(G, 0)$ as long as the sum of all the weights is 1 for the case $\theta_R = \theta$.

2.2 Learning from demonstration

The mixture of primitives learns a desired trajectory from the demonstration by adjusting the set of weights, w_i. We use recursive least squares [8] to adjust the weights on line assuming that the goal G is known[2].

Given a training point $(\dot{\mathbf{x}}, \dot{\theta}_{des})$, $\mathbf{w}$ is updated by

$$\mathbf{w}^{t+1} = \mathbf{w}^t + \mathbf{P}^{t+1}\dot{\mathbf{x}}\mathbf{e}^T \qquad (8)$$

where

$$\mathbf{P}^{t+1} = \frac{1}{\lambda}\left(\mathbf{P}^t - \frac{\mathbf{P}^t\dot{\mathbf{x}}\dot{\mathbf{x}}^T\mathbf{P}^t}{\lambda + \dot{\mathbf{x}}^T\mathbf{P}^t\dot{\mathbf{x}}}\right), \mathbf{e} = \theta_{des} - \mathbf{w}^T\mathbf{x}$$

and $\mathbf{x} = [\, x_1, \cdots, x_N \,]^T$, $\mathbf{w} = [\, w_1, \cdots, w_N \,]^T$. We choose to use velocity data for learning since we empirically find that this leads to smoother trajectories.

Figure 2 shows an example of the result of learning of the elbow joint angle during a reaching movement demonstrated by a human. The fit is based on the 20 primitives shown in Figure 1. These primitives were sufficient to provide a good fit of the trajectory, with very little residual error.

[2] λ is a forgetting factor which is normally set to 1.0 for stationary learning data (as is the case in this paper), but which might be set to a smaller value for non-stationary cases (e.g. when the goal G is not known in advance and has to be learned on line).

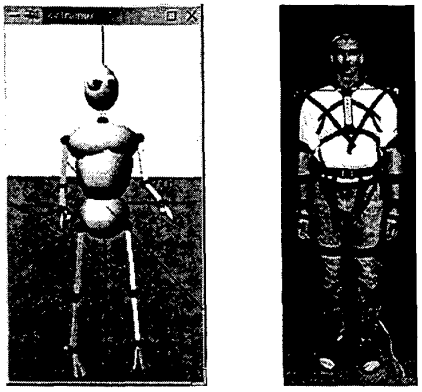

Figure 4: Left: Humanoid rigid body simulation. Right: Sensuit for recording joint-angle data.

The TFS with the fitted weights is robust towards large scale perturbations. For instance, when the position θ is forced out of its course by a transient perturbation, the trajectory planning is modified but eventually returns to the original (unperturbed) trajectory (Figure 3) and the desired point attractor. This feature is very different in comparison to non-autonomous trajectory planning, where planning would simply continue during the perturbation as if the perturbation did not exist.

3 Experimental evaluations

3.1 Humanoid robotics for rehabilitation

One of the motivations to develop the mixture of movement primitives comes from our Virtual Trainer (VT) project. The aim of this project is to supervise rehabilitation exercises in stroke-patients by (a) computerized learning the exercises from demonstrations by a professional therapist, (b) demonstrating the exercise to the patient with a humanoid simulation, (c) video-based monitoring the patient when performing the exercise, and (d) evaluating the patient's performance, and suggesting and demonstrating corrections with the simulation when necessary. Thus, the key idea is to create an interactive Virtual Trainer for movement exercising.

The essential ingredients of VT are a dynamic simulation of a humanoid robot, a trajectory formation system (TFS) with imitation abilities, and a motor system. The dynamic articulated body simulation is used to demonstrate the exercise to the patient, as well as to provide visual feedback by replicating the movements performed by the patient. The TFS is responsible for determining the kinematics of the desired trajectory, while the motor system is responsible for the correct production of the trajectory by the dynamic simulation.

As part of the VT project, the requirements for the TFS are the following. First, it must be able to fit the trajectories of the demonstrated movements with high precision. This implies fitting all degrees of freedom (DOFs) (i.e. not only end-points such as the hands), and all points of the trajectories (not only the final positions). Second, it must be robust against perturbations as some exercises require interaction with external objects, e.g., tables or tools. Finally, it must provide a good basis for comparing movements. One important aspect of VT will be to assess how well the patient performs an exercise, which therefore requires a metric for measuring differences between movements. Each of the demands can be fulfilled by the mixture of dynamic primitives. Obviously, all the requirement of the VT are equally important for humanoid robotics.

3.2 Humanoid simulation

We developed a rigid body dynamics simulation of a humanoid robot with 41 degrees of freedom (DOFs) for realizing the Virtual Trainer. The simulation is implemented using our Simulation Laboratory package [9]. The fact that the dynamics and not only the kinematics of the rigid body are simulated is an important component of VT. First, it ensures that movements are realistic and feasible. Second, it allows movements to be adapted to different body configurations by matching the dimensions, masses, and moments of inertia to those of the patient. Third it allows the torques in the joints to be monitored; limits can be set to avoid excessive torques, and can be gradually increased depending on progress. Finally, many rehabilitation exercises require interactions with objects (table, floor, ball,...). These can be realistically simulated in a dynamic simulation using standard contact models.

The dimensions and 41 DOFs of the robot approximately correspond to those of the human body (Figure 4). The main simplifications are that the robot is composed of only hinge-joints. Masses and moments of inertia of each link of the simulation are computed by assuming each link to be a cylinder and the mass to be uniformly distributed with a density corresponding to that of water. The body is fixed at the hip in the simulations for the experiments presented in this article (upper limb movements), i.e., balancing is not required. Control of the simulation is identical to that of our humanoid robot [10], i.e., using a compute torque controller at 420 Hz servo rate.

3.3 Recording of trajectories from human subject

We recorded trajectories of specific movements typically used during rehabilitation exercises with stroke patients. For this article, we recorded movements from a healthy person using a joint-angle recording system, the Sarcos Sensuit (Figure 4 right). The Sensuit allows directly recording the joint angles of 35 DOFS of the human body at 100Hz using hall effect sensors with 12 bit A/D conversion.

The movements that we recorded are part of the Wolf Motor test [11], which is an evaluation test of motor abilities commonly used with stroke patients. The test involves the measurement of times of execution of a set of well defined movements, which are performed by the patient with both the intact arm (which is used as control) and the disabled arm. This test is performed at different intervals during the patient's rehabilitation program such as to provide a quantitative basis of progress.

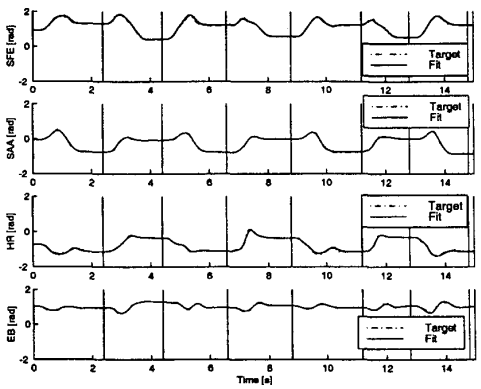

Figure 5: Fitting of an arm movement involving 4 DOFs: shoulder flexion-extension (SFE), shoulder adduction-abduction (SAA), humerus rotation (HR) and elbow rotation (EB). The recorded and fitted angles of the DOFs are shown (they are almost perfectly superposed). Vertical lines correspond to the time segmentation between zero-velocity points in the movement.

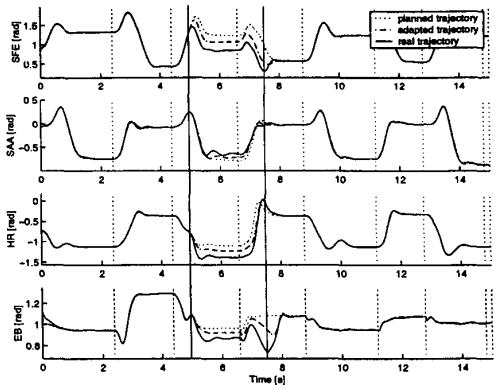

Figure 6: Reproducing the movement using the humanoid simulation. A perturbation is applied to the arm between time t=5.0s and t=7.5s through an external force on the wrist (80[N], downward). When the perturbation is applied, the real trajectories (continuous lines) are forced away from the original planned trajectory (dotted lines), which, in turn, automatically leads to the creation of an adapted desired trajectory (the dashed lines).

3.4 Fitting multi-joint movements

We used the proposed TFS to fit recordings of long multi-joint arm movements. Initially, the entire movement is segmented at zero velocity points. Each segment is then fitted using the mixture of primitives.

Figure 5 shows the recorded and fitted trajectories of a typical arm movement exercise of the Wolf Motor test. The movement consists of placing an arm from a rest position on the lap up onto a high box placed laterally to the seated subject. The exercise is here repeated three times. The TFS produces a close fit of the trajectory. Note that it can cope with the fact that most segments involve non monotonic trajectories between the beginning of the segment and the end of the segment.

The recorded movement is precisely reproduced by the humanoid simulation where the signals from the TFS are used to drive the compute torque controller of the simulation (Figure 6). Without external perturbation, there is a perfect match between the real trajectory and the desired trajectory. When a perturbation is applied which forces the arm away from its desired trajectory (e.g. an external force), the TFS temporarily modifies the desired trajectory to take the perturbation into account. A modified desired trajectory is thus created which eventually brings the real trajectory back to the planned trajectory. Note that, in this example, we feed the sequence of subgoals and primitive weights with the original timing into the TFS, which means that the duration of the whole movement is kept. This here leads to a switch of the weights of the primitives during the perturbation (at time t=6.6s), which does not disturb the trajectory creation because of the intrinsic stability of the system.

4 Discussion

We presented a trajectory formation system (TFS) based on dynamical system for reproducing human arm movements. Two key characteristics of the TFS are 1) that the trajectory is not indexed by time but rather develops out of integrating autonomous nonlinear differential equations, and 2) that the TFS does not encode one single specific desired trajectory but rather a whole attractor landscape, i.e., a control policy, with a unique point attractor, the goal state. It should be noted that our suggested approach to trajectory formation is in strong contrast with methods which explicitly index the trajectory by time such as spline fitting between via points. Explicit time indexing makes is hard to modify a trajectory in response to dynamically changing environments and strong perturbations that are beyond the abilities of a PD servo. Moreover, time-indexed trajectories are specific to initial conditions and re-using of the trajectory for new movement goals is complicated.

Our approach is inspired by the concepts of pattern generators [12] and force fields [13] found in biology. Related work includes [14, 15, 16, 17]. The TFS presented here is most closely related to the VITE model [14], but with the extension that we use multiple movement primitives to allow generating a larger class of movements, and that our dynamical systems do not require artificial reseting of the states of the dynamical systems after a movement. The concept of combining multiple primitives in a mixture model has parallels in statistical learning theory [7, 18].

Our TFS is an interesting candidate for satisfying the desirata enumerated in the introduction:

1. *Ease of representing and learning a goal trajectory.* The TFS can acquire desired trajectories by on-line recursive linear least squares regression techniques assuming that the goal is known. This property compares favorably in terms of learning speed and convergence properties towards other approaches of trajectory learning. Our only requirement is that trajectories end with zero-velocity, i.e.,

that they represent discrete movement. In our work, a set of 20 movement primitives was sufficient to represent trajectories lasting between 1.0 and 3.0 seconds. More primitives can be added for longer sequences, and/or for representing finer details of movement.

2. *Compactness.* The encoding in a mixture of dynamical systems is comparable in compactness with spline fitting using via points as movements are encoded by few parameters, namely the N weights w_i and the goals G for each trajectory.

3. *Robustness against perturbations and dynamically changing environments.* As illustrated in this article, the ability to smoothly recover from perturbations is a primary feature of the dynamical systems used in the TFS. Furthermore, the TFS gives a good basis for dealing with perturbations at a planning level instead of the execution level. For instance, recovering from a perturbation requires different actions depending on the purpose of the imitation task. If the purpose is to respect the subgoals of a movement despite the perturbation, the timing of the subgoals can be modified such that the next subgoal is fed into the system only when the current goal is reached within a satisfactory limit. On the other hand, if the purpose is to respect the timing at all costs, subgoals can be fed in at their original timings, and the TFS will naturally modify the desired trajectory during the perturbation, and skip desired subgoals (without building up a potentially huge torques in the robot's motors, as could happen in a PD controller).

4. *Ease of modification.* Desired trajectories can readily be modified by manipulating the values and timings of the subgoals, and by adding additional coupling terms. For instance, if the aim of a movement is to reach a particular object with a particular velocity profile (e.g. a tennis serve), the TFS can learn that particular velocity profile and reuse it in multiple occasions by adapting the goal to the current location of the object (if necessary, using inverse kinematics for finding the desired end positions of each DOF). Another possibility is to use potential field approaches to add attracting or repelling terms into the differential equations to navigate obstacles.

5. *Ease of categorization.* An important question in learning movements by imitation is how to recognize similar movements and which metrics to use to measure differences between movements. In future work, we intend to investigate metrics for comparing movements in the N-dimensional space which contains the weights w_i encoding a trajectory. Because it encodes velocity profiles, that space has the potential to provide a better basis for comparing movements than cartesian space or joint-angle space.

Acknowledgements

This work was made possible by support from the US National Science Foundation (Awards 9710312 and 0082995), the ERATO Kawato Dynamic Brain Project funded by the Japanese Science and Technology Cooperation, and the ATR Human Information Processing Research Laboratories.

References

[1] C.H. An, C.G. Atkeson, and J.M. Hollerbach. *Model-based control of a robot manipulator.* MIT Press, 1988.

[2] H. Miyamoto, S. Schaal, F. Gandolfo, Y. Koike, R. Osu, E. Nakano, Y. Wada, and M. Kawato. A kendama learning robot based on bi-directional theory. *Neural Networks*, 9:1281–1302, 1996.

[3] M. Kawato. Trajectory formation in arm movements: minimization principles and procedures. In H.N. Zelaznik, editor, *Advances in Motor Learning and Control*, pages 225–259. Human Kinetics Publisher, Champaign Illinois, 1996.

[4] R. Sutton and A.G. Barto. *Reinforcement learning: an introduction.* MIT Press, 1998.

[5] S. Schaal and D. Sternad. Programmable pattern generators. In *International Conference on Computational Intelligence in Neuroscience (ICCIN'98).* Research Triangle Pack NC, 1998.

[6] S. Schaal, S. Kotosaka, and D. Sternad. Nonlinear dynamical systems as movement primitives. In *International Conference on Humanoid Robotics.* 2000. (CD-ROM).

[7] G. J. McLachlan and K. E. Basford. *Mixture Models.* Marcel Dekker, 1988.

[8] S. Schaal and C. G. Atkeson. Constructive incremental learning from only local information. *Neural Computation*, 10(8):2047–2084, 1998.

[9] S. Schaal. The SL simulation and real-time control software package. Technical report, Dept. of Computer Science, U. of Southern California, 2001. http://www-slab.usc.edu/publications/.

[10] C. G. Atkeson, J. Hale, M. Kawato, S. Kotosaka, F. Pollick, M. Riley, S. Schaal, S. Shibata, G. Tevatia, and A. Ude. Using humanoid robots to study human behaviour. *IEEE Intelligent Systems*, 15:46–56, 2000.

[11] S.L. Wolf, D.E. Lecraw, L.A. Barton, and B.B Jann. Forced use of hemiplegic upper extremities to reverse the effect of learned nonuse among chronic stroke and head-injured patients. *Exp. Neurol.*, 104:125–132, 1989.

[12] F. Delcomyn. Neural basis for rhythmic behaviour in animals. *Science*, 210:492–498, 1980.

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

[14] D. Bullock and S. Grossberg. VITE and FLETE: neural modules for trajectory formation and postural control. In W.A. Hersberger, editor, *Volitional control*, pages 253–297. Elsevier Science Publishers, 1989.

[15] D. Kleinfeld and H. Sompolinsky. Associative network models for central pattern generators. In C. Koch and I. Segev, editors, *Methods in neural modeling*, pages 195–246. MIT Press, 1989.

[16] F.A. Mussa-Ivaldi. Nonlinear force fields: a distributed system of control primitives for representing and learning movements. In *IEEE International Symposium on Computational Intelligence in Robotics and Automation*, pages 84–90. IEEE, Computer Society, Los Alamitos, 1997.

[17] P.Y. Li and R. Horowitz. Passive velocity field control of mechanical manipulators. *IEEE Transactions on Robotics and Automation*, 15(4):751–763, 1999.

[18] M.I. Jordan and R. Jacobs. Hierarchical mixtures of experts and the EM algorithm. *Neural Computation*, 6:181–214, 1994.