Robot Motion Command Simplification and Scaling

Kuu-Young Young and Shi-Huei Liu

***Abstract*—It has been observed that human limb motions are not very accurate, leading to the hypothesis that the human motor control system may have simplified motion commands at the expense of motion accuracy. Inspired by this hypothesis, we propose learning schemes that trade motion accuracy for motion command simplification. When the original complex motion commands capable of tracking motion accurately are reduced to simple forms, the simplified motion commands can then be stored and manipulated by using learning mechanisms with simple structures and scanty memory resources, and they can be executed quickly and smoothly. This paper also proposes learning schemes that can perform motion command scaling, so that simplified motion commands can be provided for a number of similar motions of different movement distances and velocities without recalculating system dynamics. Simulations based on human motions are reported that demonstrate the effectiveness of the proposed learning schemes in implementing this accuracy–simplification tradeoff.**

***Index Terms*—Accuracy–simplification tradeoff, command simplification and scaling, robot learning control.**

I. Introduction

When bringing robots from factories into the human world and letting them perform tasks involving human–robot interactions, it is quite natural to introduce human motor control strategies into robot control. The human motor control system has demonstrated several appealing features and has stimulated research into human limb motions and control strategies [4], [5], [7], [9], [18], [22]. Among the features of human motor control, effectiveness in accommodating a wide variety of motions is of great research interest; by contrast, robot learning schemes usually have difficulties in facing too large a learning space for many practical applications [17], [20], [23], [26]. Because human beings are not very accurate in their movements, one hypothesis posits that the human motor control system may have the learning space simplified at the expense of accuracy in movement [24]. Meanwhile, when dealing with a group of similar motions, the human motor control system may also have executed motion command scaling, by which motion commands can be generated by scaling commands derived for some motions selected from a group rather than by repeatedly performing the learning process for each individual motion.

Even though there is still some doubt as to whether the human motor control system actually performs this tradeoff, the hypothesis is supported by evidence in human motor control studies such as the speed–accuracy tradeoff in human movements, the simplification hypothesis that the mechanism supporting a skill becomes progressively simpler with practice, etc. [3], [18]. In daily life, the process of how humans learn to generate signatures demonstrates the tradeoff between motion accuracy and command simplification. Signature generation involves fast handwriting with little demand for motion accuracy, yet handwriting is a skilled human action acquired via practice [16]. Because humans learn how to sign their names after they learn how to write, in the second learning process, they somehow learn to trade motion accuracy for motion speed and command simplicity, given that signatures are simplified forms of original handwriting. Likewise, an orchestra conductor using the same gestures with different sizes and speeds to control the tempo is a good example of motion command scaling execution in the human motor control system.

Motivated by this accuracy–simplification hypothesis, this study concentrates on how the tradeoff can be realized by using learning mechanisms and what benefits it may bring for robot learning control. This paper proposes learning schemes for motion command simplification based on the equilibrium-point hypothesis, discussed in Section II [5], [18]. By applying the proposed learning schemes, adjusted according to the degree of accuracy given up, motion governing can be transitioned from accurate motion tracking toward point-to-point motion regulation. Correspondingly, the original complex motion commands capable of tracking motion accurately are simplified. With motion commands in simple forms, learning controllers can then be designed that do not consume excessive memory resources [24]. In addition, simplified motion commands also lead to faster and simpler command execution and smooth motion control with fewer oscillations. We also propose learning schemes to trade motion accuracy for simplified motion commands that generate motions similar to the original motion, but different in movement distance and velocity. In other words, by performing motion command simplification and scaling simultaneously, the proposed learning schemes are able to achieve motion command scaling without recalculating system dynamics [8]. The rest of this paper is organized as follows. In Section II, biological backgrounds related to the human motor control system and equilibrium-point hypothesis are described. Section III proposes a learning scheme for accurate motion tracking. The proposed learning schemes for motion command simplification and scaling are introduced in Section IV; discussion about how the structure of the learning controller can be simplified according to command simplification is there as well. In Section V, simulation results and analyses based on using robot manipulators to generate human signatures and imitate orchestra conducting are reported. Conclusions are given in Section VI.

Manuscript received August 19, 1999; revised November 19, 2000 and January 26, 2002. This work was supported in part by the National Science Council, Taiwan, R.O.C., under Grant NSC 87-2213-E-009-145. This paper was recommended by Associate Editor W. A. Gruver.

K.-Y. Young and S.-H. Liu are with the Department of Electrical and Control Engineering, National Chiao-Tung University, Hsinchu 300, Taiwan, R.O.C. (e-mail: kyoung@cc.nctu.edu.tw).

Publisher Item Identifier S 1083-4419(02)04376-5.

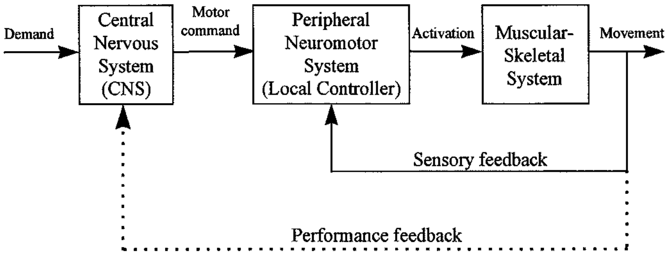


Fig. 1. Simplified block diagram of the human motor control system.

II. Human Motor Control System and Equilibrium-Point Hypothesis

Fig. 1 shows a simplified block diagram of the human motor control system that governs limb motion. In Fig. 1, we can see that human motion is governed by a hierarchical structure [9], [10], [18]. In response to various demands, the central nervous system (CNS) makes motion plans. Appropriate motor commands are then generated and sent to the peripheral neuromotor system, which may then modify the motor commands according to sensory feedback. The peripheral neuromotor system behaves as a local controller that adapts to different motions, loads, and environments, in addition to accepting commands from the CNS. Finally, the modified commands are sent to the muscular–skeletal system for motion execution. With this hierarchical structure, the difficulty of performing complex motions can be shared by the CNS at the higher level and the local controller at the lower.

Among those hypotheses for human motion control, the equilibrium-point hypothesis suggests that the CNS specifies equilibrium points between agonist and antagonist muscle groups that correctly position limbs in relation to the target by indicating new sets of length–tension curves for the muscle groups [4], [9], [18]. In other words, motions are treated as transitions between postures. The CNS needs only to select new levels for the motor commands. The subsequent result, mediated by autogenetic reflexes and the mechanical properties of the muscles, should be a smooth transition from one posture to another. The simple control-signal format makes the equilibrium-point hypothesis very attractive for robot motion control, although there are still some debates about this hypothesis. However, since only one equilibrium point is selected, a control strategy based on this hypothesis would not let one even vary the motion speed between two given postures. To exploit the simplicity of the equilibrium-point hypothesis and let it deal with different velocities and loads in reaching various positions, motor commands may consist of a number of equilibrium points. Thus, slow motions can be produced by progressive shifts of equilibrium points. Motions can be speeded up by assigning an initial shift that is larger than necessary, followed by a return to a proper static level [9]. In light of both physiological and engineering considerations, the number of equilibrium points in the motor command should be kept fairly small [7], [12], [22].

The proposed learning schemes for motion command simplification and scaling were developed according to the equilibrium-point hypothesis. By applying the proposed learning schemes, the original complex motion commands for motion governance can be simplified into series of equilibrium points in the form of square pulses of various heights and widths [12], [22], [24]. With the controlled parameters in the motion command being the heights and widths of the square pulses, the learning space for dealing with variations exhibited by different motions is greatly reduced. In the authors' previous paper [24], we demonstrated how to implement a fuzzy system with a reasonable number of rules to store and manage the heights and widths of the square pulses of the simplified equilbrium point trajectories, derived for governing a group of robot motions. This motion command simplification is, however, achieved by sacrificing motion accuracy, because continuous control signals suitable for accurate tracking are approximated by signals consisting of square pulses. Note that the controlled parameter in the equilibrium-point hypothesis is muscle compliance instead of the equilibrium point used in the proposed schemes. Our purpose is not to propose a new biological hypothesis, but rather to develop control strategies for robot motion control inspired by the human motor control system.

III. Learning Scheme for Accurate Motion Tracking

Before the proposed learning schemes for motion command simplification and scaling can be used to derive simplified motion commands for robot motions, motion commands capable of accurate motion tracking must be provided. Therefore, we propose a learning scheme for accurate motion tracking based on using robot manipulators, as shown in Fig. 2 [25]. This scheme basically emulates the simplified structure of the human motor control system shown in Fig. 1: the fuzzy neural network (FNN) acting as the CNS of the human motor control system and the local controller as the peripheral neuromotor system. The scheme is used to derive the accurate robot motion trajectory and the corresponding continuous complex motion commands for tracking the human motions studied in this paper. The derived motion trajectory and commands are then used for the study on the tradeoff between motion accuracy and command simplification.

In Fig. 2, input desired motions, such as human signature trajectories, are first transformed into Cartesian trajectories $(P_c(t), V_c(t))$ in the robot workspace via a trajectory mapping process. The Cartesian trajectory $(P_c(t), V_c(t))$ is further mapped into a joint trajectory $(q_c(t), \dot{q}_c(t))$ via an inverse-kinematic transformation. According to the joint trajectory $(q_c(t), \dot{q}_c(t))$, the FNN generates motion commands $\mathrm{EP}_c(t)$ for trajectory tracking. Note that motion commands generated by the FNN consist of equilibrium points in continuous forms. In turn, the local controller modulates the motion commands via sensory feedback and uses the resultant torque $\tau(t)$ to move the robot manipulator. According to some biological evidence, the CNS may provide only the desired equilibrium points for motion control [18]. Therefore, to simplify the design of the scheme, only the desired equilibrium points and no desired velocities are specified in the motion commands. A simple position control law with linear damping is then used for the local controller [21]:

$$\tau = K_p(\mathrm{EP}_c - q_t) - K_d\dot{q}_t \tag{1}$$

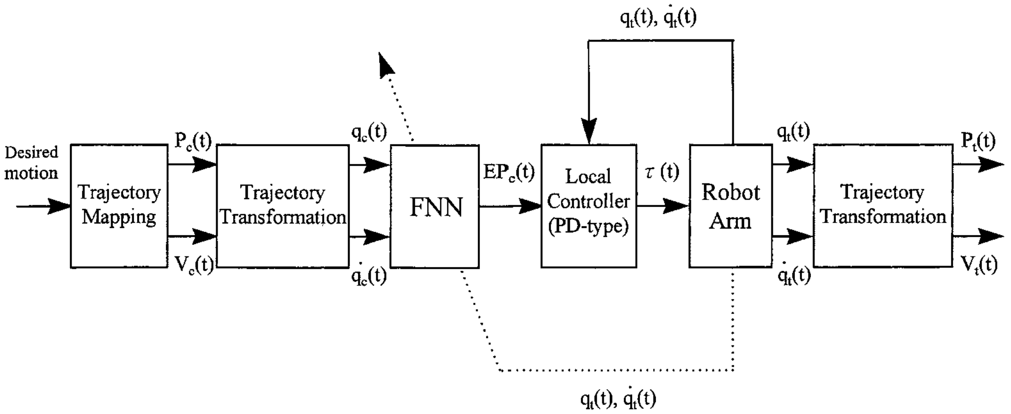


Fig. 2. Learning scheme for accurate motion tracking.

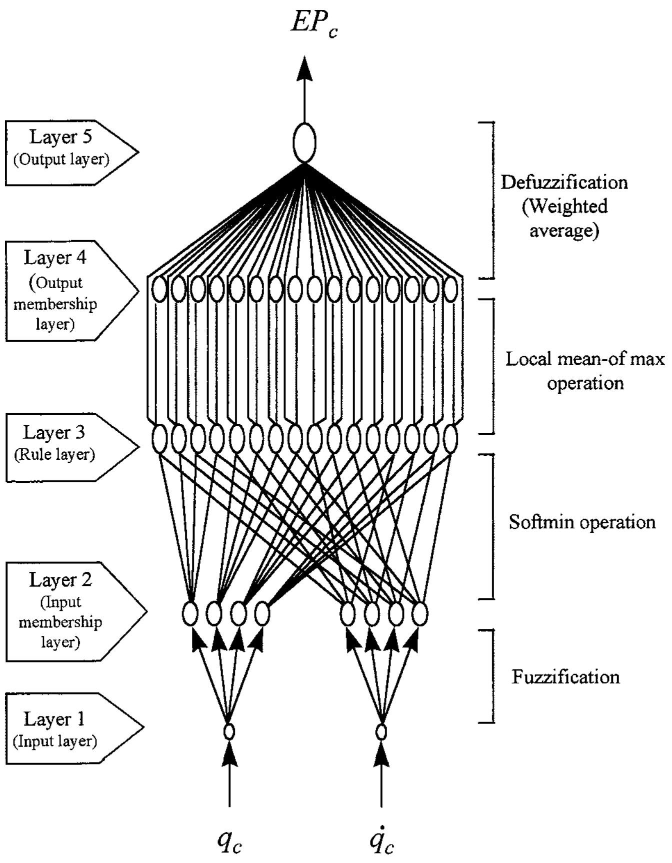

EP_c
Layer 5 (Output layer)
Layer 4 (Output membership layer)
Layer 3 (Rule layer)
Layer 2 (Input membership layer)
Layer 1 (Input layer)
Defuzzification (Weighted average)
Local mean-of max operation
Softmin operation
Fuzzification
q_c
$\dot{q}_c$

Fig. 3. Structure of the FNN.

where EP_c stands for the equilibrium point vector, q_t and $\dot{q}_t$ are the actual position and velocity vectors obtained via sensory feedback, and K_p and K_d are symmetric positive definite matrices for stability considerations [19].

The FNN for motion command generation is basically a fuzzy system implemented in the form of a neural network, as shown in Fig. 3 [1], [14]. The representation of a fuzzy system using a fuzzy neural network lets one take advantage of the learning ability of the neural network for automatic tuning of the parameters in the fuzzy system. The fuzzy reasoning parameters are thus expressed in the connection weights or node functions of the neural network. One major reason for adopting the FNN is that we also intended to investigate how the learning mechanism as a command generator can be simplified according to motion trajectory and command simplification. The simplification in a fuzzy system can be easily analyzed from its rule number and distribution [2], [13], [27]. In the proposed scheme, we chose an FNN with a structure similar to that in [1]; of course, other types of FNN can also be used. As Fig. 3 shows, the inputs to the FNN are the joint position and velocity trajectories of the input motions, $(q_c(t), \dot{q}_c(t))$, and the outputs are the equilibrium point trajectories $\mathrm{EP}_c(t)$.

It was assumed that stability and convergence of the FNN in learning to track continuous trajectories are guaranteed, and these issues are well dealt with in previous studies [1], [14]. Our previous results have demonstrated that the FNN is capable of governing continuous robot trajectories [26], and the results in this paper also show that the motions generated by the proposed scheme approximated the originals quite well. For the motions of signature generation and orchestra conducting studied in this paper, the FNN with a simple structure, described in Section V, is capable of motion governing. Thus, we did not apply sophisticated techniques to design the FNN. Nevertheless, for more complicated human motions, we suggest using FNNs with variable structures and applying the structure-parameter learning technique, for instance, that in [14], so that the FNNs can learn to govern human motions in a more flexible and effective way. Detailed discussions of the structure and learning process of this FNN are in Appendix A.

IV. Learning Schemes for Motion Command Simplification and Scaling

Fig. 4 shows the block diagram of the proposed learning scheme for motion command simplification and Fig. 5 that for motion command scaling. A progressive simplification process based on system performance is employed for motion command simplification, discussed in Section IV-A. Motion command scaling is used to generate motion commands for motions similar to the original motion, but different in movement distance and velocity, discussed in Section IV-B. The proposed learning schemes will simplify the continuous equilibrium point trajectories derived by the learning scheme for accurate motion tracking in Section III into trajectories consisting of a series of square pulses of various heights and widths. During this simplification process, the proposed learning schemes

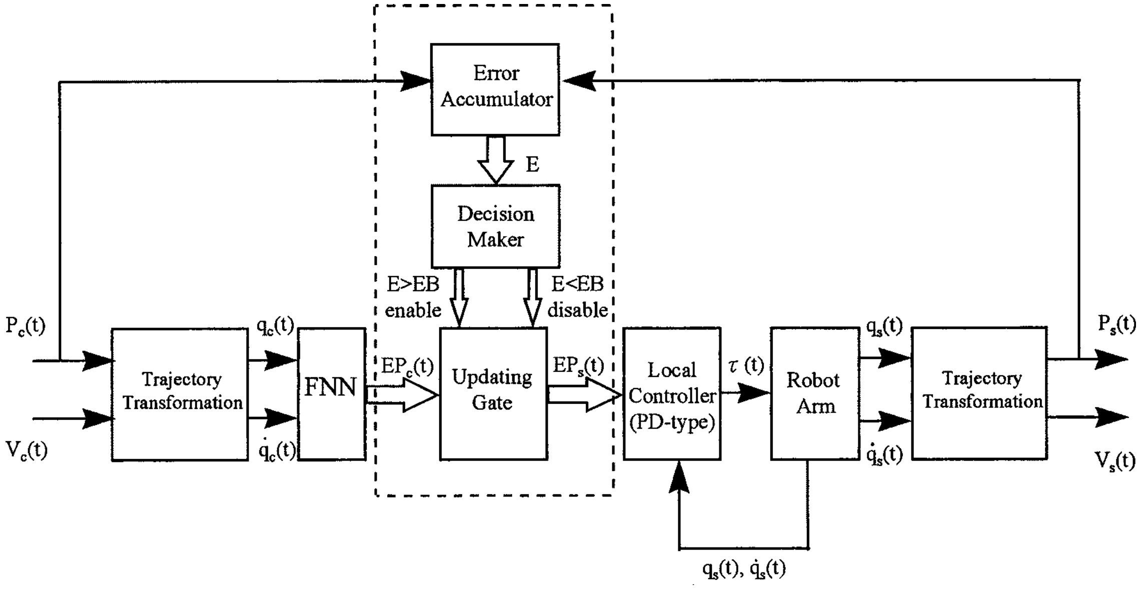
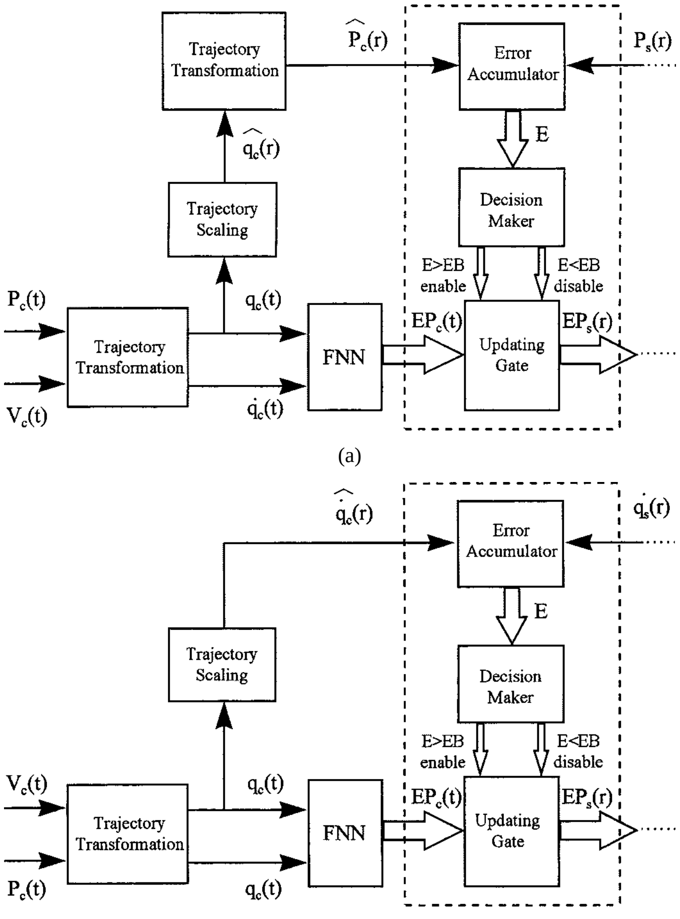


Fig. 4. Learning scheme for motion command simplification.



(a)

(b)

Fig. 5. Learning scheme for motion command scaling. (a) Size scaling. (b) Speed scaling.

gradually give up accuracy in approximating the motion trajectory, and the fundamental feature of the motion becomes more and more evident in the resulting trajectory. With motion commands in simple forms and motion trajectories in basic shapes, robot learning control can then be enhanced in both controller design and motion command execution. Finally, Section IV-C discusses how the FNN structure may be simplified according to motion trajectory and command simplification.

A. Motion Command Simplification

In the proposed learning scheme for motion command simplification, the tradeoff between motion accuracy and command simplification is achieved via a simplification process involving the error accumulator, the decision maker, and the updating gate, as shown in the blocks surrounded by the dotted lines in Fig. 4. The FNN used in this scheme is the same as that used in the learning scheme for accurate motion tracking in Section III with the learning process for accurate motion tracking completed. Therefore, before the simplification process is executed, the input joint trajectory $(q_c(t), \dot{q}_c(t))$ will elicit from the FNN continuous equilibrium point trajectories $\mathrm{EP}_c(t)$ able to track the motion accurately. In the simplification process, an error bound (EB) is first set in the decision maker. This error bound indicates the amount of accuracy to be traded for command simplification for a portion of the motion. When the cumulative error, i.e., the errors accumulated in tracking a portion of the motion, does not exceed the error bound, the updating gate is closed, letting the motion command remain at a fixed value during that period; otherwise, the updating gate is opened, and the motion command becomes the current value of $\mathrm{EP}_c(t)$. Consequently, the resulting simplified equilibrium point trajectories $\mathrm{EP}_s(t)$ will be in the form of a series of square pulses. This design causes the tradeoff to be performed in each local portion, leading to a more homogeneous tradeoff over the entire trajectory. The design can find its analogy in nonuniform sampling for image compression, in which fine sampling is applied to the neighborhood of sharp gray-level transitions in an image, and coarse sampling utilized in relatively smooth regions [6], [11], [15].

The number of square pulses in $\mathrm{EP}_s(t)$ after command simplification depends on the value of the error bound: when the error bound is large (small), $\mathrm{EP}_s(t)$ will have a small (large) number of square pulses. The Cartesian position error was used for the error bound because it relates to the Cartesian motion trajectory directly. The error bound was set to a small value at the beginning of the simplification process and increased gradually. Intuition suggests that the final value of the error bound can be determined according to the preset similarity criterion between the original and the simplified motion trajectories. However, the resemblance between these two is quite subjective and qualitative. In order to quantitatively describe the similarity between them, the concept of similarity bounding is proposed. The similarity bound E_s is defined as several times the total Cartesian error E_c between the input motion after trajectory mapping in the learning scheme for accurate motion tracking (see Fig. 2), $P_c(t)$, and the motion after learning, $P_t(t)$:

$$E_s = hE_c \tag{2}$$

where $h \geq 1$ is an empirical value, standing for the degree of similarity and referred to as a similarity index. A proper selection of h should make the original motion still recognizable from the simplified motion. The total Cartesian error E_c can be computed using

$$E_c = \sum_{i=1}^{n} [(x_c(i) - x_t(i))^2 + (y_c(i) - y_t(i))^2 + (z_c(i) - z_t(i))^2]^{1/2} \tag{3}$$

where (x_c, y_c, z_c) and (x_t, y_t, z_t) are the coordinates of the samples of $P_c(t)$ and $P_t(t)$, respectively, and n is the number of samples. Equation (3) can be applied to both two- and three-dimensional trajectories by ignoring or including the errors in the Z direction.

Based on the discussions above, the command simplification process in the proposed learning scheme will begin with a small initial error bound. The Cartesian position error will be evaluated at each sampling time. Only when the cumulative error exceeds the error bound, the updating gate is opened and the current value of $\mathrm{EP}_c(t)$ inputted as the new motion command. After motion command simplification for the entire motion has been completed, the total Cartesian error E between $P_c(t)$ and the simplified motion $P_s(t)$ is computed. When E is smaller than the similarity bound E_s, the error bound will be increased and the command simplification process resumes. The simplification process will proceed until E is greater than E_s. To summarize, the algorithm for motion command simplification is as follows.

1) Algorithm for Motion Command Simplification: Simplify continuous equilibrium point trajectories $\mathrm{EP}_c(t)$ into trajectories consisting of a series of square pulses $\mathrm{EP}_s(t)$ using prespecified degrees of similarity between originals and derived trajectories.

Step 1) Input $P_c(t)$ and $P_t(t)$.

Step 2) Compute the total Cartesian error E_c between $P_c(t)$ and $P_t(t)$. Determine the similarity bound E_s by selecting an empirical similarity index h.

Step 3) Initialize the error bound with a small value.

Step 4) Perform motion command simplification and evaluate the Cartesian position error between the current Cartesian position and the reference Cartesian position corresponding to $P_c(t)$ at each sampling time. When the cumulative error exceeds the error bound, open the updating gate and take the current value of $\mathrm{EP}_c(t)$ as the new motion command.

Step 5) Compute the total Cartesian error E between $P_c(t)$ and $P_s(t)$ after motion command simplification for the entire motion has been completed.

Step 6) Check whether E is smaller than E_s; if yes, increase error bound and go to **Step 4**; otherwise, the simplification process is completed and output the simplified equilibrium point trajectory $\mathrm{EP}_s(t)$ as a series of square pulses.

B. Motion Command Scaling

The proposed learning scheme for motion command scaling is used to generate motions that are different from the original motion in movement distance and velocity. Fig. 5 shows the block diagram for the proposed scheme, including the size scaling and speed scaling. This scheme is basically the same as that for motion command simplification in Section IV-A, except that, for motion command scaling, error evaluation executed by the error accumulator and decision maker is based on scaled reference trajectories. In other words, in the new error evaluation process, the trajectories generated by the scheme are compared with the reference trajectories after scaling, simplifying and scaling $\mathrm{EP}_c(t)$ simultaneously. The error bound may need to be increased (decreased) according to different size and speed requirements. In Fig. 5(a), to generate motions of different movement distances with the same movement velocity, the joint position trajectory $q_c(t)$ corresponding to the input position trajectory $P_c(t)$ is scaled to be $\hat{q}_c(r)$ corresponding to the scaled position trajectory $\hat{P}_c(r)$ by varying the sampling time, as follows:

$$r = c_1 t \tag{4}$$

$$\hat{q}_c(r) = (1 - c_1) \cdot q_{c0} + c_1 \cdot q_c(t) \tag{5}$$

where c_1 is a scaling constant and q_{c0} the initial joint position of $q_c(t)$. When $c_1 > 1$, it is amplification, and vice versa. In Fig. 5(b), to generate motions of the same movement distance with different velocities, the joint velocity trajectory $\dot{q}_c(t)$ corresponding to the input velocity trajectory $V_c(t)$ is scaled to be $\hat{\dot{q}}_c(r)$ by varying the sampling time, as follows:

$$r = c_2 t \tag{6}$$

$$\hat{\dot{q}}_c(r) = \frac{\dot{q}_c(t)}{c_2}. \tag{7}$$

When $c_2 < 1$, it is a speedup, and vice versa.

C. FNN Structure Simplification

When motion trajectories and commands are transformed into simple forms by applying the proposed learning schemes, it is of interest whether the corresponding FNN structures may

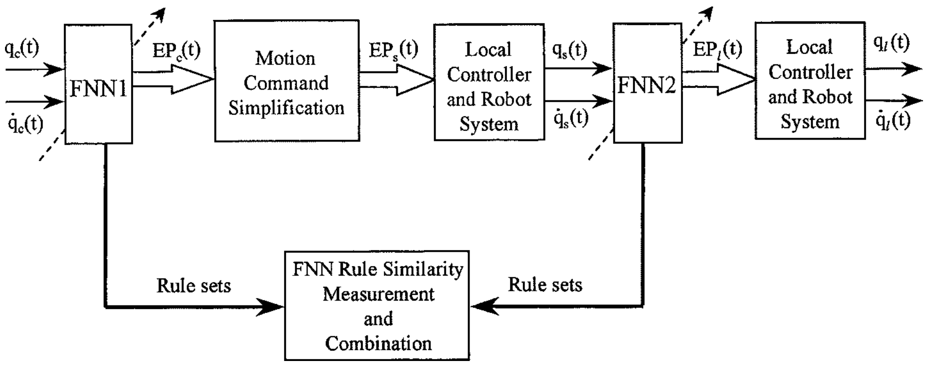

q_c(t)
q̇_c(t)
FNN1
EP_c(t)
Motion Command Simplification
EP_s(t)
Local Controller and Robot System
q_s(t)
q̇_s(t)
FNN2
EP_l(t)
Local Controller and Robot System
q_l(t)
q̇_l(t)
Rule sets
FNN Rule Similarity Measurement and Combination
Rule sets

Fig. 6. Block diagram for FNN rule similarity measurement and combination.

also be simplified. Thus, after the simplified motion trajectories were obtained, we trained another FNN to govern these simplified trajectories, and examined whether the rules in the FNNs would become similar. If so, the similar fuzzy rules can then be combined, leading to simplified FNN structures. Fig. 6 shows the block diagram for FNN rule similarity measurement and combination for the two FNNs governing the original and simplified motion trajectories. In Fig. 6, an FNN, FNN1, is first trained to govern an input motion $(q_c(t), \dot{q}_c(t))$ with motion commands $\mathrm{EP}_c(t)$. The motion commands $\mathrm{EP}_c(t)$ are simplified to be $\mathrm{EP}_s(t)$ by applying the algorithm for motion command simplification in Section IV-A, and $\mathrm{EP}_s(t)$ will generate a simplified motion $(q_s(t), \dot{q}_s(t))$. Another FNN, FNN2, is then trained to govern $(q_s(t), \dot{q}_s(t))$ with motion commands $\mathrm{EP}_l(t)$. Note that, although motion commands $\mathrm{EP}_l(t)$ are obtained through the learning for governing $(q_s(t), \dot{q}_s(t)), \mathrm{EP}_l(t)$ will generate a new simplified motion $(q_l(t), \dot{q}_l(t))$ instead of $(q_s(t), \dot{q}_s(t))$. This is because motion commands $\mathrm{EP}_s(t)$ are in the form of a series of square pulses, while $\mathrm{EP}_l(t)$ are in continuous forms. Finally, we evaluated the similarity of the fuzzy rule sets in these two FNNs that govern motions $(q_c(t), \dot{q}_c(t))$, and $(q_l(t), \dot{q}_l(t))$, performed rule combination when the similarity between rules was above a given threshold, and analyzed whether simplified motion trajectories would correspond to simplified FNN structures. Detailed discussions on how to perform the FNN rule similarity measurement and combination can be found in Appendix B [2], [13], [27].

Another issue of interest is whether, when the fundamental feature of the motion becomes evident in the resulting trajectory via simplification, the corresponding FNN rule distribution will also possess a similar characteristic. Simulations have been performed to investigate the relationship between simplified motion trajectories, motion commands, and FNN rule distributions, as described in the next section.

V. Simulation Results and Analyses

To demonstrate the effectiveness of the proposed motion command simplification and scaling schemes, they were applied to generate human signatures and imitate orchestra conducting for the two-joint planar robot manipulator, shown in Fig. 7. Three adult subjects were asked to provide motion samples. They practiced writing on the digital tablet for a while, and their samples were recorded after they were confident with using the digital tablet. The subjects were told to move quickly to generate more natural motions, and to select satisfactory samples from what they generated, according to their own standards. The selected samples were then mapped into Cartesian trajectories $(P_c(t), V_c(t))$ in the robot workspace using the learning scheme for accurate motion tracking described in Section III. Via a learning process, the equilibrium point trajectories $\mathrm{EP}_c(t)$ were derived, which in turn generated trajectories $(P_t(t), V_t(t))$ approximating $(P_c(t), V_c(t))$. The two-joint planar robot manipulator was used to simulate the hand and pen system, and its dynamic equations are expressed as

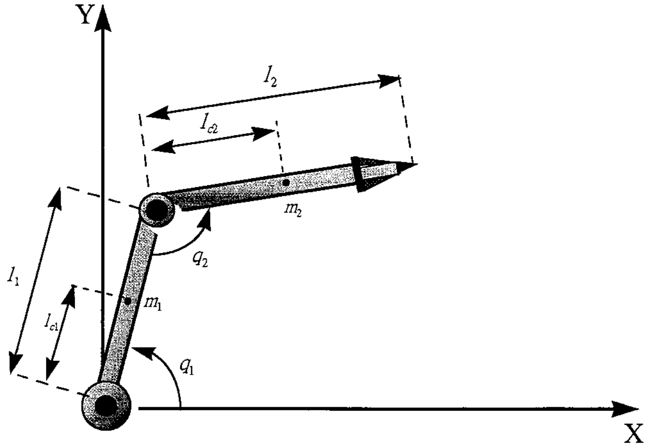


Fig. 7. Two-joint planar robot manipulator.

$$\begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} = \begin{bmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{bmatrix} \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} + \begin{bmatrix} -c\dot{\theta}_2^2 - 2c\dot{\theta}_1\dot{\theta}_2 \\ c\dot{\theta}_1^2 \end{bmatrix} \tag{8}$$

where

$$H_{11} = m_1 l_{c1}^2 + m_2 l_1^2 + m_2 l_{c2}^2 + 2m_2 l_1 l_{c2}\cos(\theta_2) + I_1 + I_2 \tag{9}$$

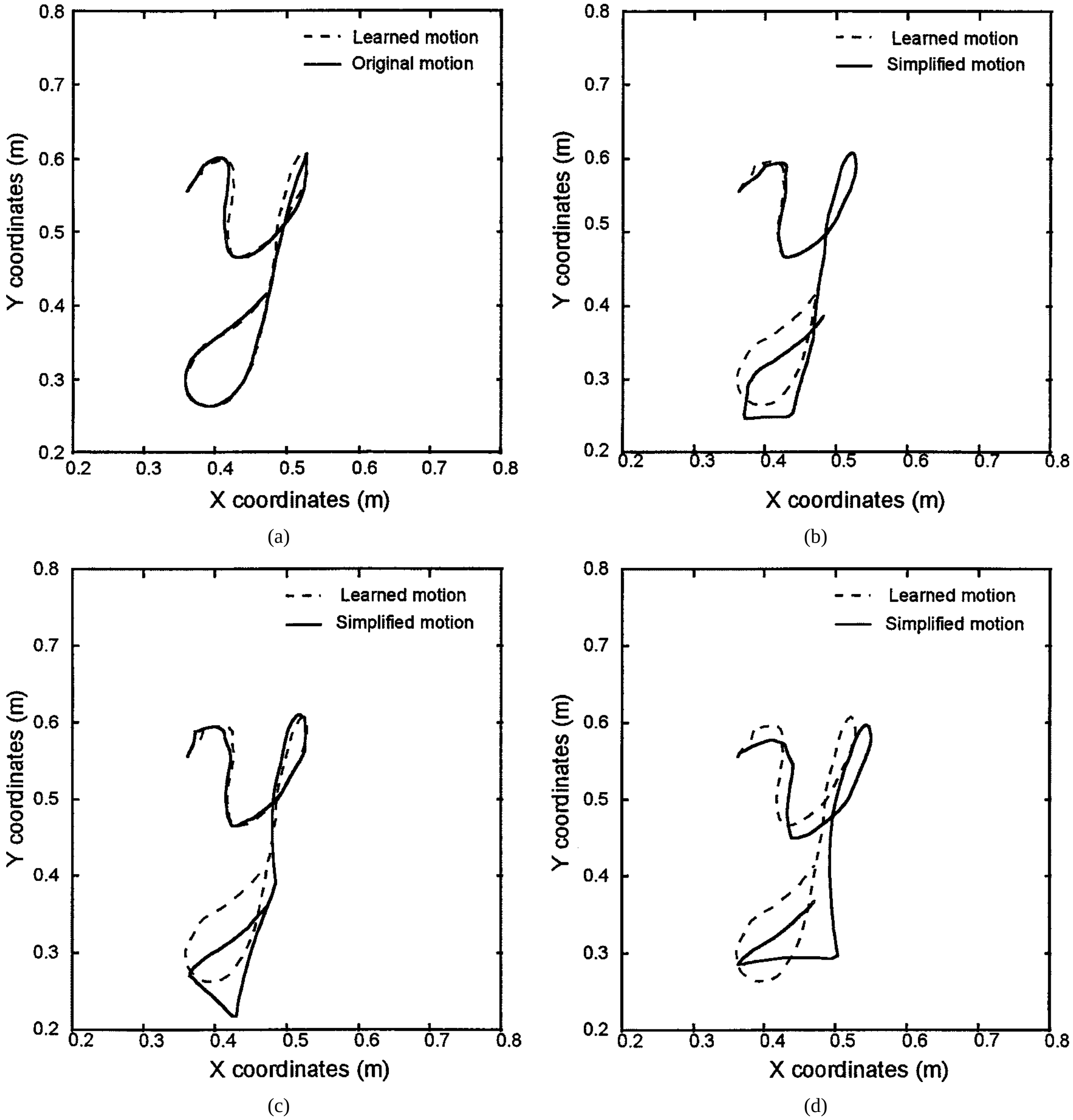
$$H_{12} = m_2 l_{c2}^2 + m_2 l_1 l_{c2}\cos(\theta_2) + I_2 \tag{10}$$

$$H_{21} = H_{12} \tag{11}$$

$$H_{22} = m_2 l_{c2}^2 + I_2 \tag{12}$$

$$c = m_2 l_1 l_{c2}\sin(\theta_2) \tag{13}$$

with τ_1 and τ_2 standing for the torques, θ_1 and θ_2 the joint variables, $m_1 = 2.815$ kg, $m_2 = 1.64$ kg, $l_1 = 0.5$ m, $l_2 = 0.6$ m,



(a)

(b)

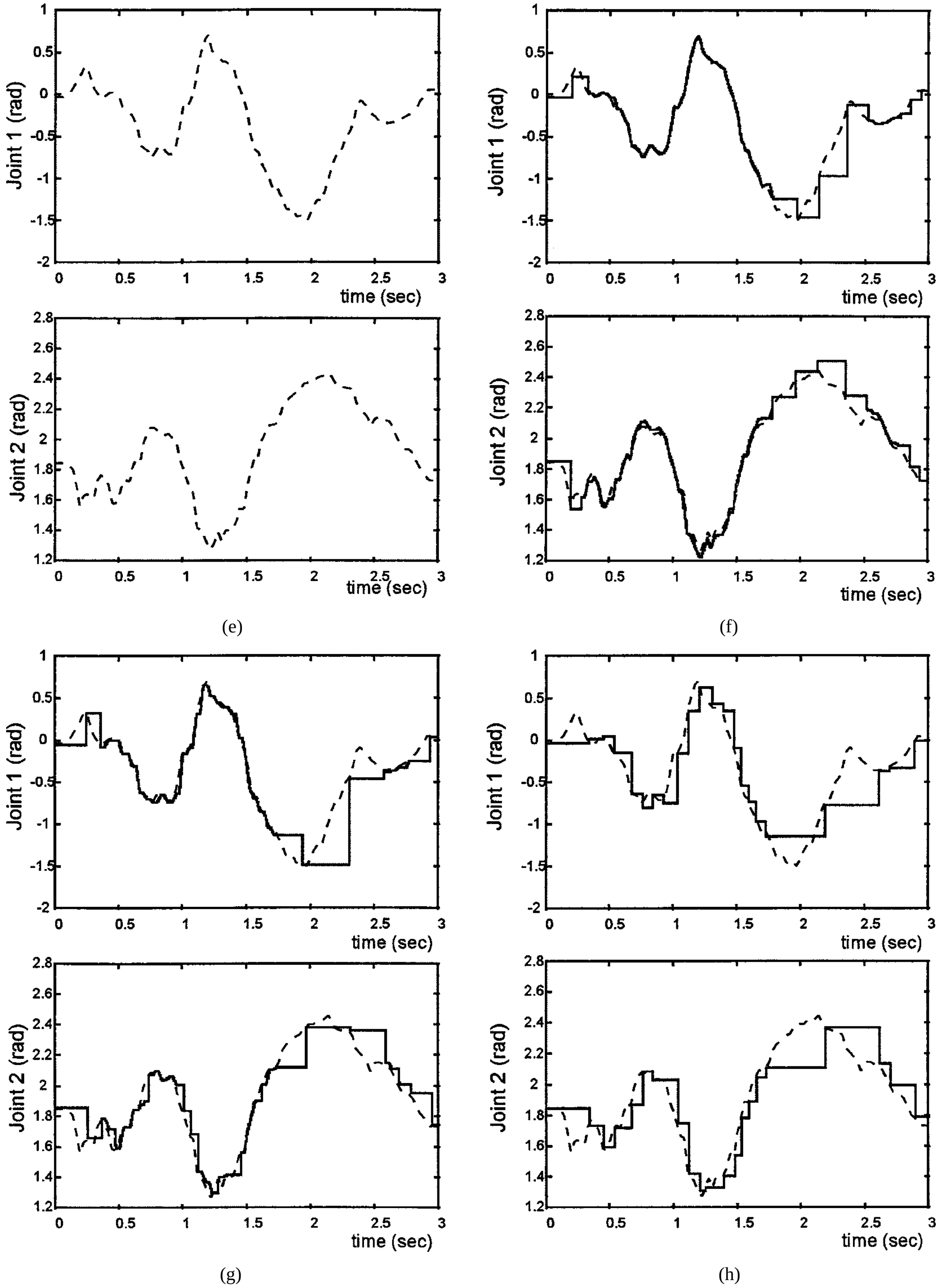
(c)

(d)

Fig. 8. Motion command simplification for the character *y*. Cartesian position trajectories: (a) accurate motion tracking, (b) motion command simplification with $h = 2.5$, (c) motion command simplification with $h = 5$, and (d) motion command simplification with $h = 7.5$.

$l_{c1} = 0.25\,\mathrm{m}, l_{c2} = 0.3\,\mathrm{m}$, and $I_1 = I_2 = 0.0234\,\mathrm{kgm}^2$. The effects of load and gravity were ignored in the formulation, and the sampling time in the simulation was 2 ms. For all schemes, each joint of the robot manipulator was equipped with an FNN and a local controller. In each FNN, there were two nodes in Layer 1, eight nodes in Layer 2, 16 nodes each in Layers 3 and 4, and one node in Layer 5. The local controller gains were selected to ensure system stability and performance, and they varied along with different motions.

Fig. 8(a)–(d) shows the resulting position trajectories for an input handwritten character *y* by applying the learning scheme for Fig. 8(a), accurate motion tracking; Fig. 8(b)–(d), motion command simplification with the empirical similarity index $h = 2.5, 5$, and 7.5, respectively; and Fig. 8(e)–(h), the corresponding motion commands. In Fig. 8(a), the original input handwritten *y* trajectory used for reference is marked by the solid line and the learned trajectory by the dashed line, and in Fig. 8(b)–(d), the simplified trajectories are marked by the solid line. In Fig. 8(e)–(h), the continuous $\mathrm{EP}_c(t)$ for accurate motion tracking were simplified to be $\mathrm{EP}_s(t)$, consisting of a series of square pulses, and the number of square pulses decreases along with the increase of the similarity index h. Correspondingly, motion trajectories in Fig. 8(b)–(d) deviated from the original one gradually, while they were still recognizable. We then applied the procedure for FNN rule similarity measurement and combination, shown in Fig. 6, to find whether the corresponding FNN structure might be simplified for the simplified motion trajectory. As an example, Fig. 9 shows the output membership functions in Layer 4 of the FNN for governing the learned trajectory and the simplified trajectory with $h = 5$ in Fig. 8(c). The fuzzy rule numbers reduced from eleven to five, and from 22 to five for joints one and two, respectively, as the trajectory was simplified. A more complicated sample, the name *Liu*, was also studied. Fig. 10(a)–(d) shows the resulting position trajectories for *Liu* by applying the learning scheme for Fig. 10(a), accurate motion



(e)

(f)

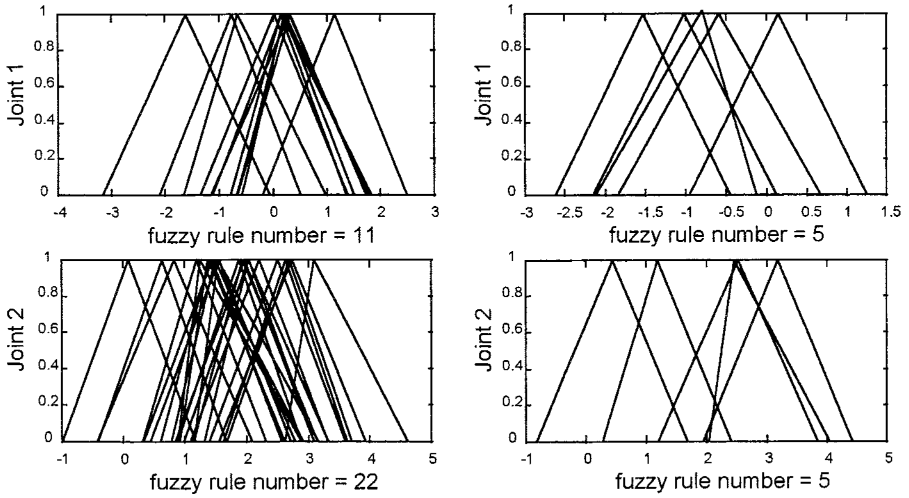
(g)

(h)

Fig. 8. (*Continued.*) Corresponding motion commands: (e) accurate motion tracking, (f) motion command simplification with $h = 2.5$, (g) motion command simplification with $h = 5$, and (h) motion command simplification with $h = 7.5$.

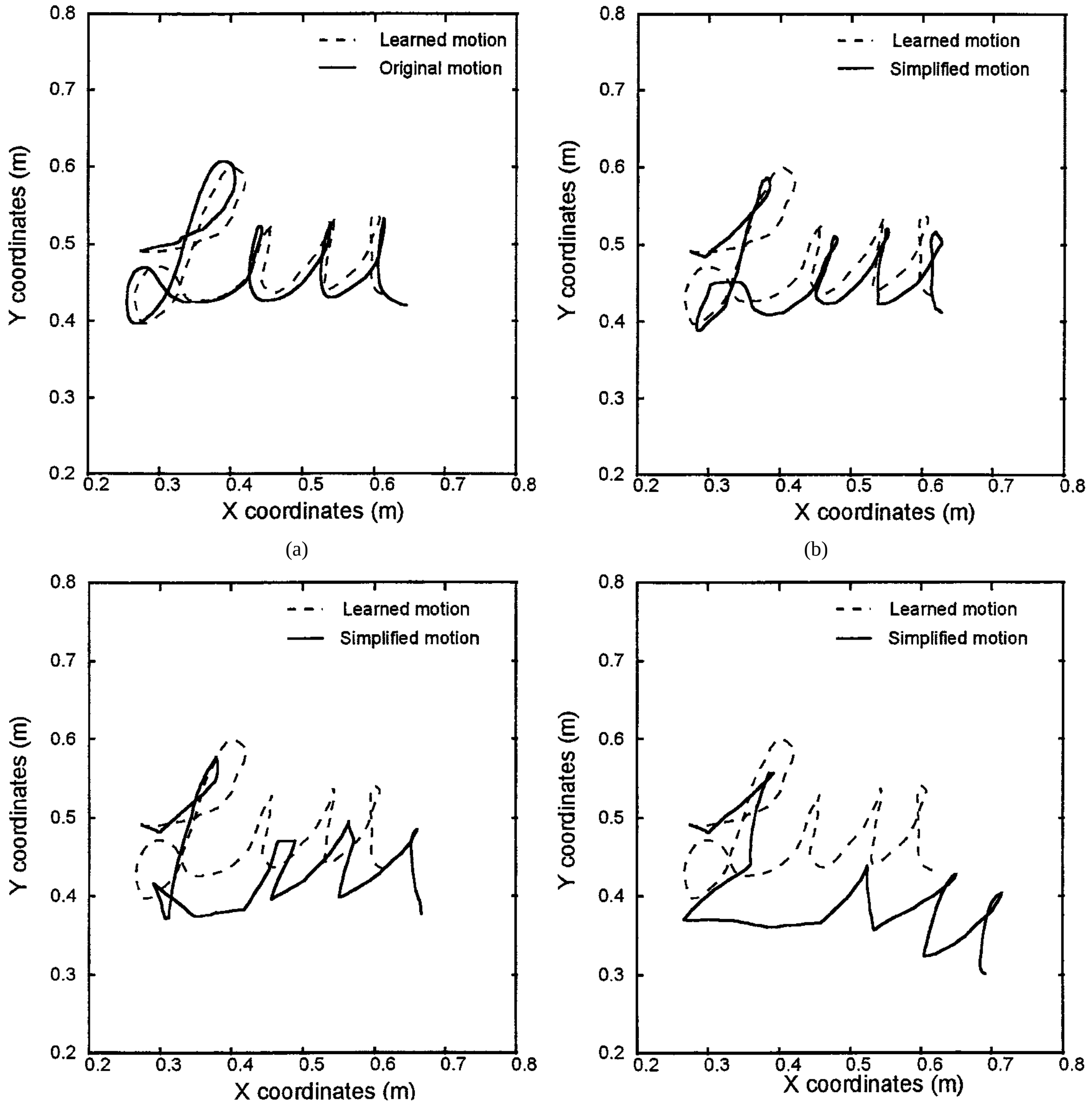
tracking; Fig. 10(b)–(d), motion command simplification with the empirical similarity index $h = 2, 4$, and 6, respectively; and Fig. 10(e)–(h), the corresponding motion commands. The tracking errors for *Liu* were larger than those for the character *y* and the shapes of the simplified square-pulse motion commands varied more abruptly, as expected, while the simplified trajectories for *Liu* were still recognizable. In this case, the FNN structure corresponding to the simplified trajectories were also simplified. From the results, the learning scheme for motion command simplification can simplify motion commands to a prespecified degree of similarity between originals and derived trajectories, and the corresponding FNN structures can also be simplified.

In the second set of case studies, we applied the proposed learning schemes for imitating orchestra conducting. First, we used this orchestra conducting imitation to study motion command scaling. Fig. 11 shows the generation of three-beat conducting gestures under different size and velocity requirements



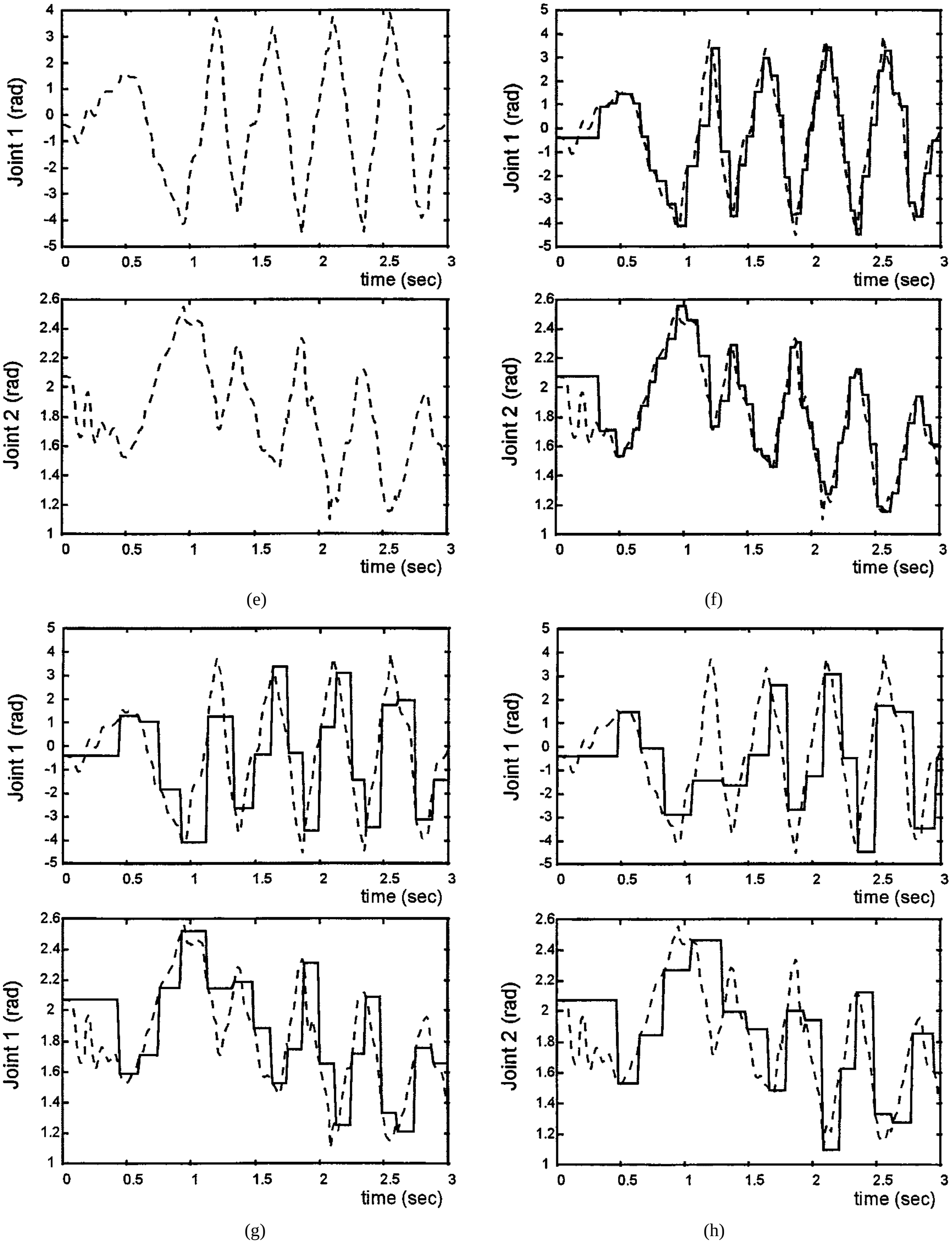
(a) (b)

Fig. 9. Output membership functions in Layer 4 of the FNN for governing the learned and simplified trajectories of *y*: (a) accurate motion tracking and (b) motion command simplification with $h = 5$.



(a) (b) (c) (d)

Fig. 10. Motion command simplification for the name *Liu*. Cartesian position trajectories: (a) accurate motion tracking, (b) motion command simplification with $h = 2$, (c) motion command simplification with $h = 4$, and (d) motion command simplification with $h = 6$.



(e)

(f)

(g)

(h)

Fig. 10. (*Continued.*) Corresponding motion commands: (e) accurate motion tracking, (f) motion command simplification with $h = 2$, (g) motion command simplification with $h = 4$, and (h) motion command simplification with $h = 6$.

and the corresponding motion commands. In Fig. 11(a), the original conducting motion used for reference is marked by the solid line and the learned motion by the dashed line. Fig. 11(b) and (c) show both a larger gesture and a normal gesture conducted more rapidly, marked by the solid line. The reference motions for error evaluation during motion command scaling were generated using (4)–(7), with the scaling constants $c_1 = 1.5$ and $c_2 = 0.5$ for the larger and faster gestures, respectively. Due to the increases in size and movement velocity, the similarity indices were increased accordingly. In Fig. 11(b) and (c), the continuous $\mathrm{EP}_c(t)$ in Fig. 11(a) were simplified and scaled into series of square-pulse trajectories. In Fig. 11(b), the simplified motion command trajectory is longer than the original one for generating a larger movement with the same movement velocity, while in Fig. 11(c), it is shorter for generating the same movement with a larger movement velocity. Both the larger and faster gestures were generated with errors within the similarity bounds, demonstrating the

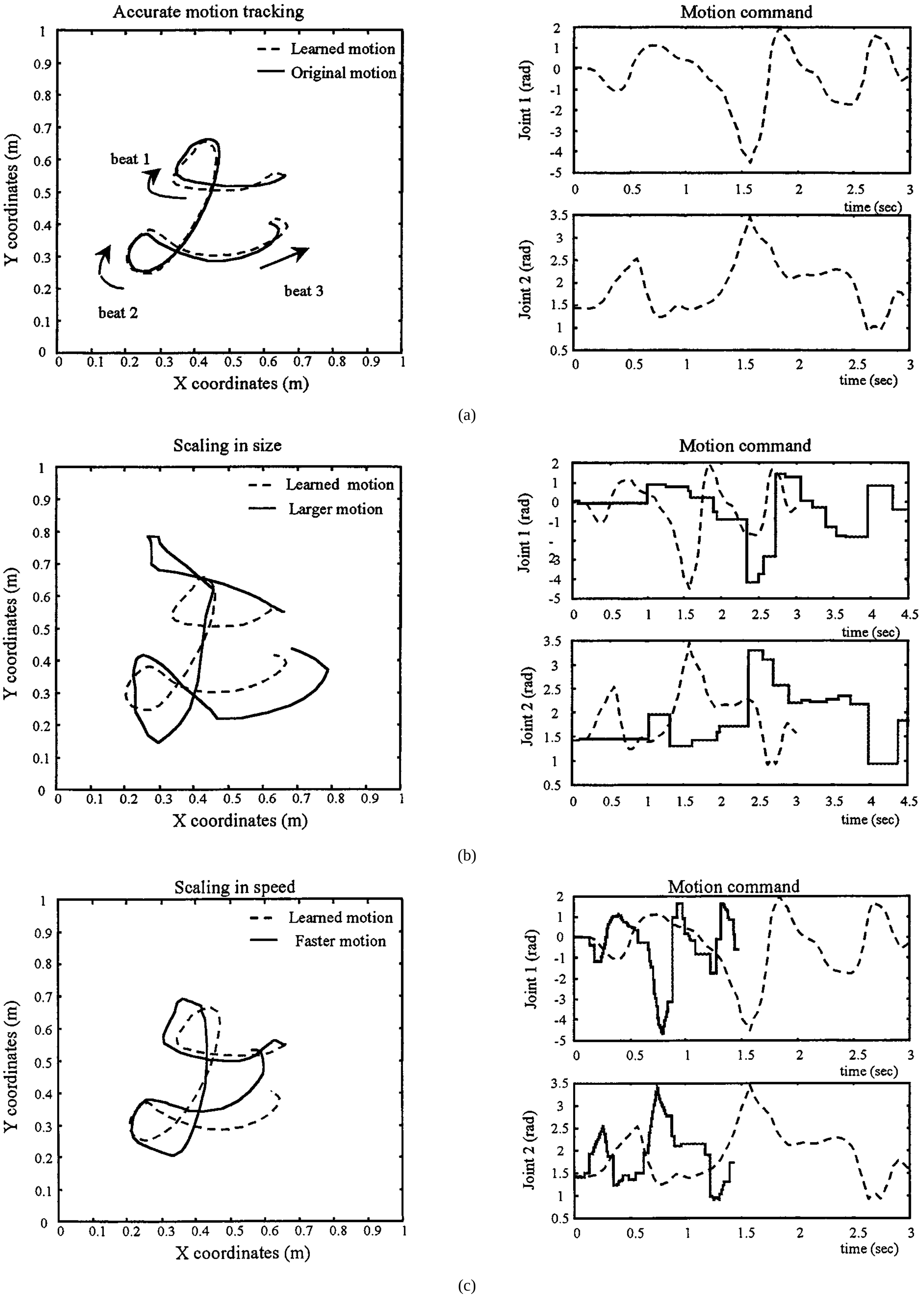


Fig. 11. Generation of three-beat conducting gestures under different size and velocity requirements and the corresponding motion commands. (a) Reference gesture. (b) Larger gesture. (c) Faster gesture.

feasibility of the proposed motion command scaling scheme. The relationship between simplified motion trajectories, motion commands, and FNN rule distributions was then studied by applying the fuzzy rule similarity measurement and combination

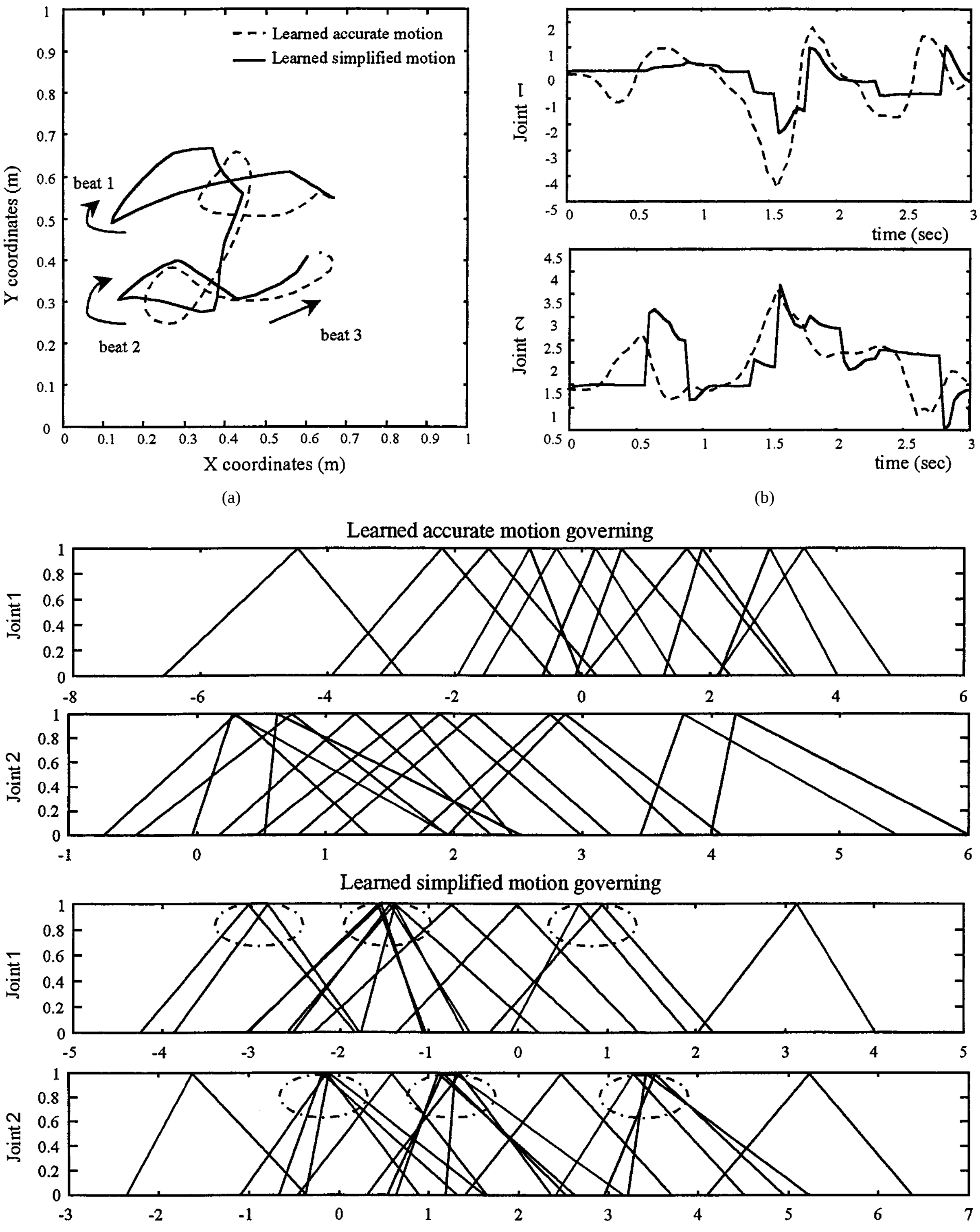
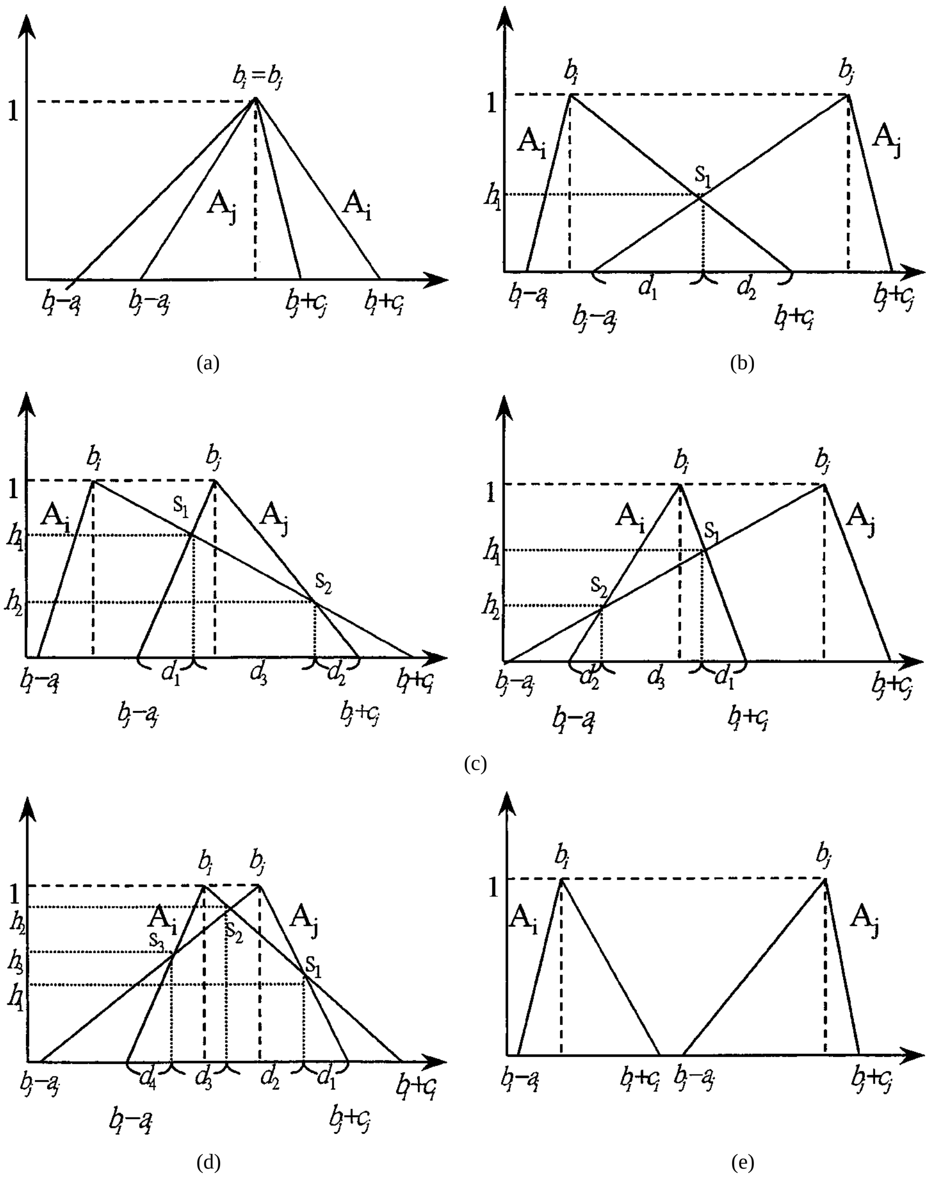


Fig. 12. (a) Learned accurate and simplified trajectories of the three-beat conducting gesture $(h = 6)$. (b) Corresponding motion commands. (c) Corresponding output membership functions in Layer 4 of the FNN.

on the governing FNNs for the original and simplified $(h = 6)$ conducting motion trajectories. Fig. 12(a) shows the learned accurate and learned simplified trajectories of the three-beat conducting gesture, marked by the dashed and solid lines, respectively, Fig. 12(b), the corresponding motion commands, and Fig. 12(c), the corresponding output membership functions in Layer 4 of the FNN after rule similarity measurement and combination. In Fig. 12, the fundamental feature of the motion trajectory is its three-beats, which is more evidently reflected in the motion commands for the learned simplified trajectory, and its corresponding FNN rule distribution also shows three evident groupings (marked by dotted circles). The results



(a) (b)

(c)

(d) (e)

Fig. 13. Similarity measure of two fuzzy sets, A_i and A_j. (a) $A_j \subseteq A_i$. (b) A_i and A_j have one intersection point. (c) A_i and A_j have two intersection points. (d) A_i and A_j have three intersection points. (e) $A_i \cap A_j = \emptyset$.

implicate that simple, basic motion commands that capture fundamental motion features may correspond to simple FNN structure and rule distribution, while further investigation is demanded for solid conclusions.

VI. Conclusion

This paper has developed learning schemes for robot motion command simplification and scaling. The proposed motion command simplification and scaling is taken as a second learning process after accurate motion tracking has been accomplished. Thus, the proposed learning schemes provide effective frameworks for achieving fast, simple control when a task does not demand high accuracy, and to transition between motion tracking and regulation according to the degree of motion accuracy given up. Simulation results based on using robot manipulators to generate human signatures and imitate orchestra conducting demonstrate the effectiveness of the proposed schemes. The results also show that, when the fundamental feature of the motion becomes evident in the resulting simplified trajectory, the corresponding FNN rules may also be similarly distributed. In our future work, we will perform further investigation on this issue. Both two- and three-dimensional human motions will be studied. In addition, we will also apply the proposed schemes to analyze fundamental motion features from the standpoint of learning, such that motions can be properly simplified, classified, and managed with learning mechanisms in simple structures.

Appendix A
Description of the FNN

The structure of the FNN used in the proposed schemes consists of five layers of nodes, all of which are of the same types within the same layer, as shown in Fig. 3. Each of the five layers performs one stage of the fuzzy inference process, as described below.

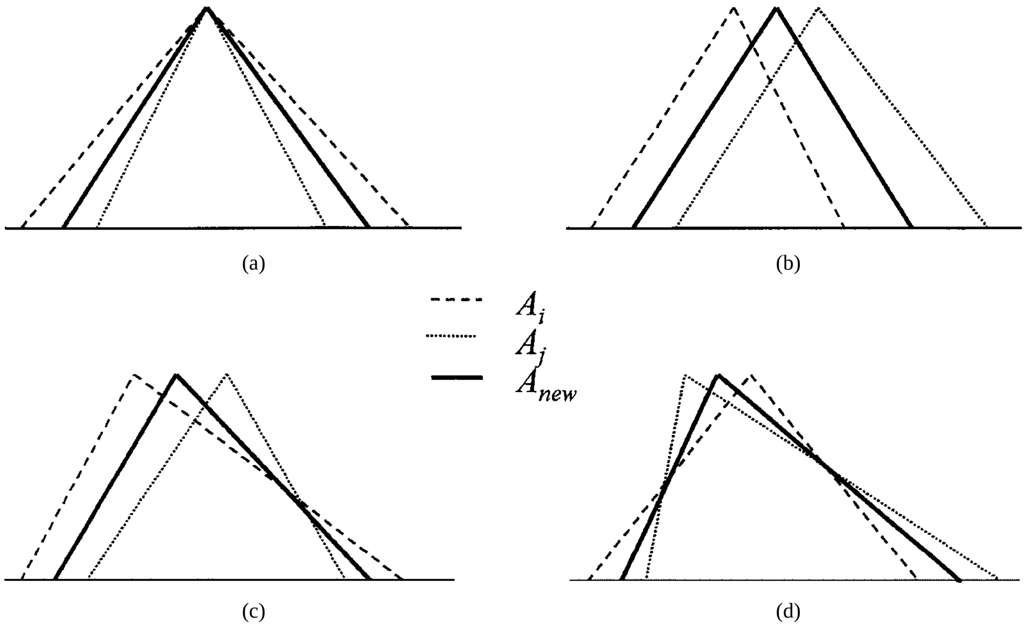


Fig. 14. Combination of two fuzzy sets, A_i and A_j, with a high degree of similarity. (a) $A_j \subseteq A_i$. (b) A_i and A_j have one intersection point. (c) A_i and A_j have two intersection points. (d) A_i and A_j have three intersection points.

Layer 1: The input layer transmits inputs directly to the next layer without performing any computation. As Fig. 3 shows, there are two nodes for two inputs, q_c and $\dot{q}_c$, for motions with a single degree of freedom.

Layer 2: The input membership function layer transforms input data into fuzzy data. Each node i in this layer has the node function

$$O_i^2 = \mu(x) \tag{A.1}$$

where $\mu : X \rightarrow [0, 1]$ is a membership function and x is the input to node i. The triangular membership function adopted is

$$\mu(x) = \begin{cases} 1 - \frac{(x-b)}{c}, & x \in [b, b+c] \\ 1 + \frac{(x-b)}{a}, & x \in [b-a, b] \\ 0, & \text{otherwise.} \end{cases} \tag{A.2}$$

Different membership grades at the same crisp point can be obtained by adjusting the parameter set (a, b, c).

Layer 3: The rule layer implements fuzzy rules. Each node in this layer corresponds to a rule, defined as a fuzzy conditional statement of the form

$$\text{Rule: IF } X \text{ is } A \text{ and } Y \text{ is } B \text{ THEN } Z \text{ is } C \tag{A.3}$$

where X and Y are fuzzy sets representing the inputs, Z represents the output, and A, B, and C represent linguistic variables, such as small, medium, and large. The number of rules involved in the input–output relationship is prespecified. In this layer, each node also outputs the firing strength of the rule, O_i^3, by performing the differentiable softmin operation [1]

$$O_i^3 = \frac{\sum_j O_j^2 \exp\left(-rO_j^2\right)}{\sum_j \exp\left(-rO_j^2\right)} \tag{A.4}$$

where O_j^2 is the output of the jth node in Layer 2 connected to the ith node in Layer 3, and r is a constant. When r approaches infinity, the softmin operator becomes a min operator, for finite r, O_i^3 is differentiable, which is required during the learning process.

Layer 4: This is the output membership function layer. Each node i in this layer performs an inversion of μ_i to locate the X coordinate of the centroid of the membership function, O_i^4, using the local mean-of-maximum method (LMOM) [1]

$$O_i^4 = \mu_i^{-1}(O_i^3). \tag{A.5}$$

Layer 5: The output layer has as many nodes as there are output action variables. Fig. 3 shows only one node is needed for the single motion command EP_c. The defuzzification approach adopted is the weighted averaging method:

$$O^5 = \frac{\sum_i O_i^3 O_i^4}{\sum_i O_i^3}. \tag{A.6}$$

Because the number of rules in Layer 3 is prespecified and weights for the input and output layers (Layers 1 and 5) are fixed, the parameters to be learned in this FNN are the modifiable weights present at the input links to Layers 2 and 4, which correspond to the input and output membership functions. When the FNN learns the input and output membership function parameters required to generate the motion command EP_c corresponding to a sampled motion, an error rate, related to the motion command EP_c and the resultant motion, is initially specified in the last layer (Layer 5). This error rate is then back-propagated to adjust the parameters from layer to layer sequentially. Because a concise form of the inverse dynamic model of the robot manipulator is not available, the error rate cannot be obtained directly by differentiating the error between the desired motion and the actual motion relative to the motion command. Instead, we use the combined feedback error of position (e) and velocity $(\dot{e})$ between the desired and actual motions, denoted as $E = G_p e + G_d \dot{e}$, to derive the error rate $(\partial E / \partial \mathrm{EP}_c)$ [10]:

$$\begin{aligned} \frac{\partial E}{\partial \mathrm{EP}_c} &= \frac{\partial E}{\partial O^5} \\ &= \eta(G_p e + G_d \dot{e}) \end{aligned} \tag{A.7}$$

where η is a learning rate and G_p and G_d are gains. The error rate $(\partial E / \partial \mathrm{EP}_c)$ in (A.7) is estimated, but not exact, for describing the differential relationship between the motion command EP_c and the resultant motion. Nevertheless, the results in [10] and this study show that the use of this error rate is appropriate for the learning. Using the error rate $(\partial E/\partial \mathrm{EP}_c)$ and some straightforward manipulation, we were able to derive updates for the parameters in Layers 2 and 4.

APPENDIX B
RULE SIMILARITY MEASUREMENT AND COMBINATION FOR THE FNN

Basically, the concepts described in [2], [13], and [27] were used to perform FNN rule similarity measurement and combination. After the FNN is trained to govern an input motion successfully, the similarities between the membership functions in the FNN are evaluated pair by pair. When there are similar membership functions, these indicate that some of the rules are unnecessary and can be eliminated. Similarities between fuzzy sets can be evaluated by comparing the areas covered by fuzzy sets according to geometric points. The similarity measure between two fuzzy sets, A_i and A_j, is thus defined as

$$E(A_i, A_j) = \frac{M(A_i \cap A_j)}{M(A_i \cup A_j)} \tag{B.1}$$

where $E(A_i, A_j)$ is the degree of similarity between A_i and A_j; $\cap$ and $\cup$ denote the intersection and union operators, respectively; and $M(\cdot)$ is the size of a fuzzy set, i.e., the area it covers. For the six cases of membership function overlapping between A_i and A_j shown in Fig. 13, (B.1) is used for similarity measurement [27]. Then, with a given threshold Υ between 0 and 1 and when $E(A_i, A_j) \geq \Upsilon$, the two fuzzy sets A_i and A_j were combined into a new fuzzy set A_{new} by computing their geometrical average. Fig. 14 shows the four cases of combination of two fuzzy sets when A_i and A_j are highly similar.

REFERENCES

[1] H. R. Berenji and P. Khedkar, "Learning and tuning fuzzy logic controllers through reinforcements," *IEEE Trans. Neural Networks*, vol. 3, pp. 724–740, Sept. 1992.

[2] C. T. Chao, Y. J. Chen, and C. C. Teng, "Simplification of fuzzy-neural systems using similarity analysis," *IEEE Trans. Syst., Man, Cybern. B*, vol. 26, pp. 344–354, Apr. 1996.

[3] P. M. Fitts, "The information capacity of the human motor program in controlling the amplitude of movement," *J. Exper. Psychol.*, vol. 47, pp. 381–391, 1954.

[4] T. Flash, "The control of hand equilibrium trajectories in multi-joint arm movements," *Biol. Cybern.*, vol. 57, pp. 257–274, 1987.

[5] H. Gomi and M. Kawato, "Equilibrium-point control hypothesis examined by measured arm stiffness during multijoint movement," *Science*, vol. 272, no. 5258, pp. 117–120, 1996.

[6] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*. Reading, MA: Addison-Wesley, 1992.

[7] G. L. Gottlieb, D. M. Corcos, and G. C. Agarwal, "Organizing principles for single-joint movements—I: A speed-insensitive strategy," *J. Neurophysiology*, vol. 62, no. 2, pp. 342–357, 1989.

[8] J. M. Hollerbach, "Dynamic scaling of manipulator trajectories," *ASME J. Dynamic Syst., Measure., and Contr.*, vol. 106, pp. 102–106, 1984.

[9] *Handbook of Physiology—The Nervous System II: Neural Control of Muscle Length and Tension*, vol. II, Amer. Physiology Soc., 1981, pp. 257–323.

[10] M. Kawato, K. Furukawa, and R. Suzuki, "A hierarchical neural-network model for control and learning of voluntary movement," *Biol. Cybern.*, vol. 57, pp. 169–185, 1987.

[11] T. Kuyel, W. Geisler, and J. Ghosh, "Retinally reconstructed images: Digital images having a resolution match with the human eye," *IEEE Trans. Syst., Man, Cybern. A*, vol. 29, pp. 235–243, Mar. 1999.

[12] S. L. Lehman, "Input identification depends on model complexity," in *Multiple Muscle Systems*. New York: Springer-Verlag, 1990, pp. 94–100.

[13] C.-T. Lin, "A neural fuzzy control system with structure and parameter learning," *Fuzzy Sets Syst.*, vol. 70, pp. 183–212, 1995.

[14] C.-T. Lin and C. S. G. Lee, "Reinforcement structure/parameter learning for neural-network-based fuzzy logic control systems," *IEEE Trans. Fuzzy Syst.*, vol. 2, pp. 46–63, Feb. 1994.

[15] N. Peterfreund and Y. Y. Zeevi, "Nonuniform image representation in area-of-interest systems," *IEEE Trans. Image Processing*, vol. 4, pp. 1202–1212, Sept. 1995.

[16] R. Plamondon and F. Maarse, "An evaluation of motor models of handwriting," *IEEE Trans. Syst., Man, Cybern.*, vol. 19, pp. 1060–1072, Oct. 1989.

[17] T. D. Sanger, "Neural network learning control of robot manipulators using gradually increasing task difficulty," *IEEE Trans. Robot. Automat.*, vol. 10, pp. 323–333, June 1994.

[18] R. A. Schmidt and T. D. Lee, *Motor Control and Learning: A Behavioral Emphasis*, 3rd ed. Champaign, IL: Human Kinetics, 1999.

[19] L. Sciavicco and B. Siciliano, *Modeling and Control of Robot Manipulators*. New York: McGraw-Hill, 1996.

[20] T. Shibata and T. Fukuda, "Hierarchical intelligent control for robotic motion," *IEEE Trans. Neural Networks*, vol. 5, pp. 823–832, Sept. 1994.

[21] M. Takegaki and S. Arimoto, "A new feedback method for dynamic control of manipulators," *ASME J. Dynam. Syst., Meas., Contr.*, vol. 103, no. 2, pp. 119–125, 1981.

[22] C. H. Wu, K. Y. Young, K. S. Hwang, and S. Lehman, "Voluntary movements for robotic control," *IEEE Control Syst. Mag.*, vol. 12, pp. 8–14, Feb. 1992.

[23] B.-H. Yang and H. Asada, "Progressive learning and its application to robot impedance learning," *IEEE Trans. Neural Networks*, vol. 7, pp. 941–952, July 1996.

[24] K. Y. Young and C. C. Fan, "An approach to simplify the learning space for robot learning control," *Fuzzy Sets Syst.*, vol. 95, pp. 23–38, Apr. 1998.

[25] K. Y. Young, J. F. Lee, and H. J. Jou, "Robot learning schemes that trade motion accuracy for command simplification," *Fuzzy Sets Syst.*, vol. 110, pp. 313–329, Mar. 2000.

[26] K. Y. Young and S. J. Shiah, "An approach to enlarge learning space coverage for robot learning control," *IEEE Trans. Fuzzy Syst.*, vol. 5, pp. 511–522, Nov. 1997.

[27] K. Y. Young and J. K. Wang, "Robot motion similarity analysis using an FNN learning algorithm," *Fuzzy Sets Syst.*, vol. 124, pp. 155–170, Dec. 2001.

Kuu-Young Young was born in Kaohsiung, Taiwan, R.O.C., on Dec. 22, 1961. He received the B.S. degree in electrical engineering from National Taiwan University, Taiwan, in 1983, and the M.S. and Ph.D. degrees in electrical engineering from Northwestern University, Evanston, IL, in 1987 and 1990, respectively.

Between 1983 and 1985, he served as an Electronics Officer in the Taiwan Navy. Since 1990, he has been with the Department of Electrical and Control Engineering, National Chiao-Tung University, Hsinchu, Taiwan, where he is currently a Professor. His research interests include robot compliance control, robot learning control, robot calibration and path planning, teleoperation, and VR/robot integration.

Dr. Young received the Excellent Young Electrical Engineer Award from the Chinese Institute of Electrical Engineering, Taiwan, in 2000.

Shi-Huei Liu was born in Taipei, Taiwan, R.O.C., on Nov. 12, 1973. He received the B.S. and M.S. degrees in electrical and control engineering from National Chiao-Tung University, Hsinchu, Taiwan, in 1996 and 1998, respectively.

His research interests lie in the areas of robotics, neural networks, fuzzy systems, learning control, and biological control systems.