Proceedings of the 17th World Congress
The International Federation of Automatic Control
Seoul, Korea, July 6-11, 2008

Imitation Learning of Robot Movement Using Evolutionary Algorithm

GaLam Park [*,**] **Syungkwon Ra** [*] **ChangHwan Kim** [*] **JaeBok Song** [**]

[*] *Center for Cognitive Robotics Research, Korea Institute of Science and Technology, Seoul, 136-791, Republic of Korea (Fax:+82-2-9585749; e-mail: mysoulmate79@gmail.com, syungkwon.ra@gmail.com, ckim@ kist.re.kr)*
[**] *Dept. of Mechanical Engineering, Korea University, Seoul, Republic of Korea, (e-mail: jbsong@korea.ac.kr)*

Abstract: This paper presents a new framework to generate human-like movement of a humanoid robot in real time using the movement primitive database of a human. The framework consists of two processes: (1) the offline motion imitation learning based on Evolutionary Algorithm and (2) the movement generation of a humanoid robot using the database updated by the motion imitation learning. For the offline process, the initial database contains the kinetic characteristics of a human, since it is full of humans captured motions. The database then develops through the proposed framework of motion learning based on Evolutionary Algorithm, having characteristics of a humanoid in aspect of minimal torque. The humanoid generates a human-like movement corresponding to a given task in real-time by linearly interpolating the primitive movements in the developed database. The proposed framework is a systematic methodology for a humanoid robot to learn human motions, considering the dynamics of the robot. The experiment of catching a ball thrown by a man is performed to show the feasibility of the proposed framework.

1. INTRODUCTION

Recently, humanoid robots have increasingly resembled human beings in such motion control ability as walking control, running control, etc. as well as in appearance. Although the recent progress in motion generation for humanoid robots has yield useful results in many cases, a general approach for automatic motion generation is still under development. Therefore it is necessary to propose an efficient way in which a robot can coordinate its motions for numerous kinds of tasks required in our daily life.

In spite of the great differences between a vertebrate system and robotic system, the fields of biology and neuroscience have tried to derive benefits from the theories and procedures that may help the control of an artificial multi-joint system : motor learning (R.A. Schmidt, 1988) and motor primitives (F. A. Mussa-Ivaldi et al., 1994). Hollerbach and Flash (J. M. Hollerbach et al., 1982) proved experimentally that brain may carry out inverse kinematics and dynamic computations when a human moves in a purposeful. In addition, Raibert (M. Raibert et al., 1978) observed that inverse dynamics can be represented as the operation of a memory that associates a vector of joint torques, angles, angular velocities and angular accelerations-a computational device like look-up table. Mussa-Ivaldi and colleagues (F. A. Mussa-Ivaldi et al., 1994, 2000) insisted that the central nervous system could create, update and exploit internal representations of limb dynamics like a memory and motor pattern generator in order to deal with the computation complexity of inverse dynamics. They regarded the vectorial combination of the force field exerted by limbs as the representation. Williamson (M.M.Williamson, 1996) applied this combination to a robotic system in form of postural primitives.

Imitation Learning might be an efficient approach to enable a robot to generate natural and abundant motions (S. Schaal, 2002). When someone needs to learn a new skill or new sports motion, he/she watches the actions of a skilled person, and, subsequently, uses the demonstrated movement as a seed to start his/her own movement. In cognitive science work, imitation style learning has been investigated as a source of higher order intelligence and fast acquisition of knowledge (S. Schaal, 1999). Recorded human motions have been a good means for teaching various movements. Through some statistical analysis on human motions, Mataric and colleagues have designed the automated derivation method of kinematic-level primitives and used it to classify movement and reconstruct the original movement (A. Fod et al., 2002). Park and colleagues appended the inverse dynamics-based optimization to Matarics reconstruction algorithm for generating diverse movement of robot (B. Lim et al., 2005). Their optimization method is not good enough in computation efficiency so that it may not be suitable in online robot motion programming. Nakanishi et al. developed the methods of deriving robust imitation of joint trajectories using dynamical movement primitives based on non-linear oscillators (J. Nakanishi et al., 2004).

The goal of this paper is to develop a framework that helps a humanoid robot to learn human-like motions using a movement primitive database, which is advanced

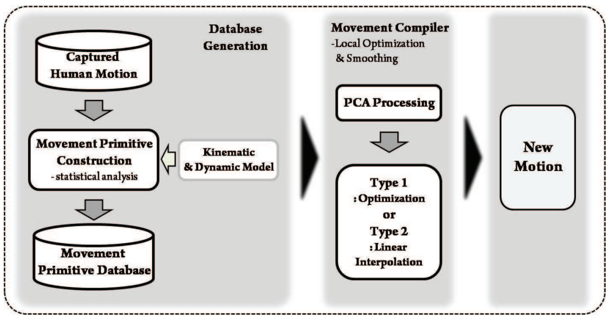
Captured Human Motion
Database Generation
Movement Primitive Construction - statistical analysis
Kinematic & Dynamic Model
Movement Primitive Database
Movement Compiler -Local Optimization & Smoothing
PCA Processing
Type 1 : Optimization or Type 2 : Linear Interpolation
New Motion

Fig. 1. The overall procedure of motion generation

with an imitation learning process. The following section introduces the motion generation based on principal components analysis. Section 3 then describes proposed framework of learning process for a humanoid robot using Evolutionary Algorithm. For a Sec. 4, the simulation of catching a ball is performed, showing the feasibility of the proposed framework.

2. PCA-BASED MOTION GENERATION

Beyond all things, this section is described for the motion generation in the proposed imitation learning, which uses the principal components analysis. It consists of using the optimization and linear interpolation. One is utilized in the genetic operator which is the core factor in the proposed imitation learning.(will be explained in Sec. 3.1) The other is utilized in the online motion generation.

To generate the motion database for a given task, we first capture human motions with various conditions during the performing of the task. We call the joint trajectories of such motions stored in the database as movement primitives (see Fig. 1). The inverse kinematics solver based on optimization in (B. Lim et al., 2005) is used to obtain the joint trajectories from the marker trajectories of the motion capture system. It should be noticed that such movement primitives seem to have not only kinematics information of the human but also be dynamically consistent and optimized, since they are generated by the human behaviors based on the several years of experiences.

For a given task like catching a ball that will be discussed later as an example, we need to define the conditions for the task. In this case, the conditions are the ball positions caught in the air. For the given task with various conditions like NC different ball positions, we find NC movement primitives and conditions. When a condition like an arbitrary ball position is given, p movement primitives in the database are selected using the distance difference between the conditions in database and the given condition. The principal components analysis (PCA) on these selected p movement primitives is performed and the most dominant few principal components are obtained. Subsequently, these principal components are used as bases for interpolation and optimization to generate new humanlike motions with minimal torques. We call this procedure the movement compiler as shown in Fig.1.

2.1 Movement Compiler via Interpolation

When a condition (in the example, a ball catching position) is given, it corresponds to the joint positions of a robot arm. Satisfying these final positions, the joint trajectories of the arm need to be determined. The method devised by (A. Fod et al., 2002) and (T. D. Sanger, 2000) is employed to do this. Using principal components of the selected p movement primitives for the given condition, each of the arm joints is given in terms of the mean trajectory, the first three dominant principal components, and an unknown constant as follows:

$$q(t) = q_{mean}(t) + \sum_{i=1}^{3} x_i \cdot q_{pc_i}(t) + x_4 \tag{1}$$

where $q(t)$ is the joint trajectory, $q_{mean}(t)$ is the mean trajectory obtained from N captured human motions, $q_{pc_i}(t)$ is the i-th most dominant principal component, and $x_i (i = 1, 2, 3)$ are the scalar weighting coefficient. x_4 denotes an unknown constant to represent a remaining error term. As the condition (a set of initial and final joint positions and velocities) is given as

$$q(t_0) = q_0,\ q(t_f) = q_f,\ \dot{q}(t_0) = \dot{q}_0,\ \dot{q}(t_f) = \dot{q}_f \tag{2}$$

we can solve these four linear equations for the three unknowns, $x_i (i = 1, 2, 3)$.

A humanoid robot can then generate the desired motion that satisfies the given condition. It is noticed that the bases, principal components in Eq.(1), may play a key role to characterize the motion. The motion can not only looks like human's but also needs possibly minimal torque, if the motions in the database are optimized to minimize torque and the principal components of such motions are used in Eq.(1). It is therefore important how to update the motion database. We employ the concept of Evolutionary Algorithm to update the database, requiring minimal torques at the joints and maintaining human-likeness for a certain condition. The following section describes this optimization problem.

2.2 Movement Compiler via Optimization

This subsection describes the strategy of dynamics-based optimization using PCA in (B. Lim et al., 2005). In addition to Eq.(1), one more principal component is introduced to make an optimization problem to minimize the joint torques as

$$q(t) = q_{mean}(t) + \sum_{i=1}^{4} x_i \cdot q_{pc_i}(t) + x_5 \tag{3}$$

where x_5 is defined due to the same reason as x_4 in Eq.(1).

The equations of motion for the humanoid robot, which is modeled as a set of coupled rigid bodies, are given as

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + N(q, \dot{q}) = \tau \tag{4}$$

where $M(q) \in \Re^{n \times n}$ is the mass matrix, $C(q, \dot{q}) \in \Re^{n \times n}$ is the coriolis matrix, $N(q, \dot{q}) \in \Re^{n}$ includes gravity and other forces, and τ is the joint torque vector. n is the degrees of freedom of the robot. For the optimization problem, we minimize the torques as

$$\min_{x} \frac{1}{2} \int_{t_0}^{t_f} \|\tau(q, \dot{q}, \ddot{q})\|^2 dt \tag{5}$$

subject to Eq.(4) and the boundary conditions in Eq.(2). More details on this formulation are referred to (S.Lee et al., 2005 and J.E. Bobrow et al., 2001). Though this procedure requires somewhat computational efforts, the

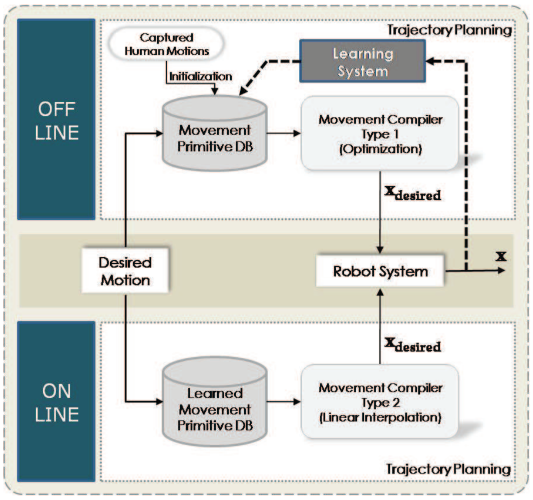

Fig. 2. The overall process of the proposed imitation learning

subsequent optimal motion is reflective of dynamic feature of the humanoid robot and also closely resembles the captured human movements as well. This problem will be resolved to update the human motion database by imposing the dynamics characteristics of the robot (actually minimal torques).

3. IMITATION LEARNING USING EA

This section describes the framework to update the motion database, which is initially full of only captured human motions, using an Evolutionary Algorithm.

When a humanoid robot wants to move its arm to do such motions as grasping a cup on the table or catching a ball in the air, it should generate the joint trajectories to reach its arm to the target conditions like a cup position or a ball catching position. These target conditions are arbitrary such that the robot needs to generate the appropriate joint motions that look like human's motions and require minimal torques. For this reason, the paper introduces a framework of learning to imitate a human motion for a given task.

The proposed framework of imitation learning is divided into the two parts as seen in Fig.2: the offline and the online process. The offline process updates the motion database considering minimal torques and using Evolutionary Algorithm. The online process then uses this database to generate the human-like movement in real time for performing a given task.

3.1 Offline Process : imitation learning

For a task like catching a ball or grasping a cup, this process updates the database, which is initially full of only human captured motions that are recorded as many times as possible using a motion capture system or others. There are several reasons to update the database. One is that it may be hard to obtain all the necessary motions from a human for doing the given task. Therefore, the database needs to be enriched to make up the motions that are not recorded from the human. Secondly there are needs for involving the dynamics characteristics of a humanoid

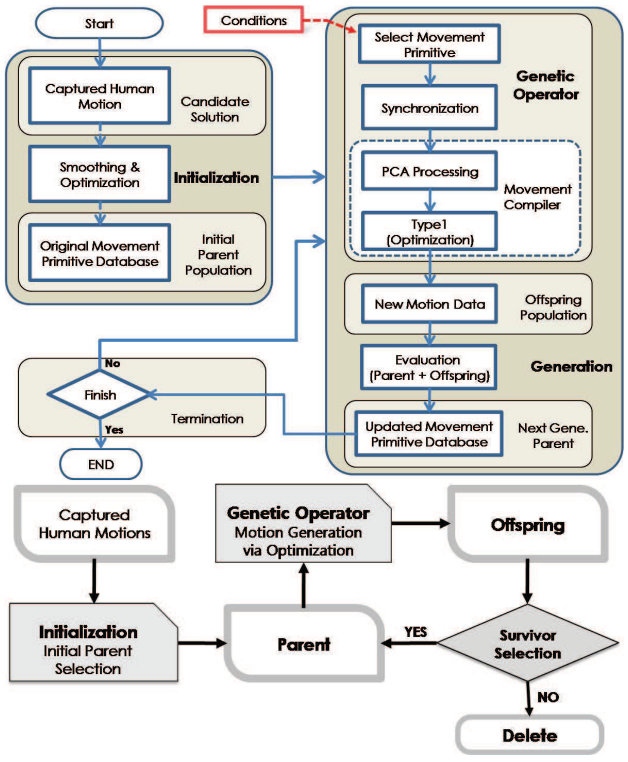

Fig. 3. The overall offline process

robot, which is to require minimal torques. The other one is to maintain human-likeness in movement.

Due to the reasons above, the paper uses an Evolutionary Algorithm(EA) to deal with optimizing the database, not a single objective function. Evolutionary Algorithm(EA)s are a family of optimization and searching techniques inspired by the Darwin's Theory of Evolution (A.E. Eiben et al., 2003). EA is known to be proper not only to get a single candidate solution but to obtain a whole collection of candidate solutions simultaneously like our motion database.

Figure3 shows the overall procedure of the imitation learning in the offline part. We explain the procedure in sequence.

1) Initialization : We use the captured human motions data as an original movement primitive database. As population is the collection of candidate solutions(movement primitives), so-called individuals. Because a movement primitive is vector time series data of joint angle, the individual is represented by real-value vectors.

2) Parent : Parents are the collection of movements, which denotes to the movement primitive database. An initial parent uses the original movement primitive database. The role of next generation parent selection is to distinguish among individuals based on their quality, in particular, to allow the better individuals to become new parents for the next generation. In the problem of motion generation, such a conditions as initial joint angle, final joint angle, joint angular velocity vary according to intrinsic characteristic. So it is hard to define a general metric of quality. Therefore, before parent selection, we create a set of conditions that has a strong likelihood of happening in online stage. This set of conditions is used repeatedly in the subsequent evo-

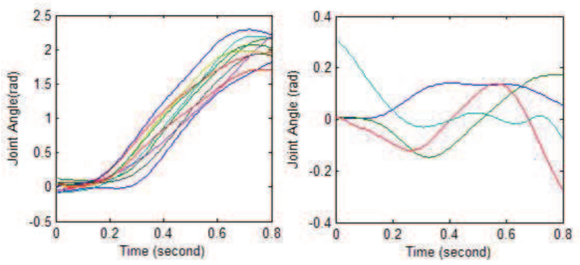

Fig. 4. Trajectories of a specific joint in selected movement primitives (left), and their first four dominant principal components (right)

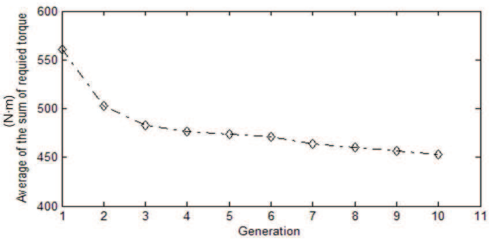

Fig. 5. The average of the sum of required torque for each movement primitive database

lution process, the learned movement primitive database that is the output of our learning system is full of optimal motions with respect to the set of conditions.

3) Genetic Operator : Genetic Operator is the core factor procedure in EA. This is the variation operator to create new individuals from old ones. This operation is fully conducted by movement compiler via optimization with respect to a set of conditions given in parent selection component. For instance, assuming that we need the motion satisfying one condition, then we select n movement primitives which are analogous to the condition from a movement primitive database. The analogousness is determined by a suitable distance metric, e.g. the distance for position vector condition. Using then the movement compiler via optimization, we get the resultant movement satisfying the condition. The above procedure is performed for the all conditions, e.g. if the number of condition is , we can acquire resultant movement primitives.

4) Offspring : As the result of Genetic Operator procedure, there are as many movements as the number of condition, so-called 'offspring'. For each generation, we can require movements, having a same end-position, because of setting the same conditions for it.

5) Survivor Selection : Now, we have to select survivor among offspring and parent based on their quality. The one cycle of EA is called a generation. The survivor becomes the individual for the next generation as the environmental pressure causes natural selection in nature. We choose the sum of torque needed to operate the movement as a quality criterion, so we intend to gradually improve the movement primitive database into inverse dynamics-based optimal data set. The fitness function to be minimized is defined as Eq.(5). The joint torque vector is calculated by robot inverse dynamics theory, given the joint trajectory

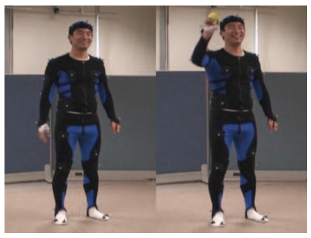

Fig. 6. The motion capture system

vector. we then discard the wrong offspring that a joint trajectories exceed the allowable limit of robot.

6) Repeating : The above procedure is repeated until the total number of fitness evaluations reaches a given limit. As the evolution processes, the individuals lose the characteristics of human motions more and become adapted to the physical feature of the robot. So, for the purpose of the harmony of two characters, a proper number of generations of evolution have to be chosen.

3.2 Online Process : real-time motion generation

In this online process, the humanoid robot can generate an arbitrary motion in real time satisfying a desired motion, which can be done using the resultant of learning process. The motion generation by the humanoid is accomplished using 'the movement compiler via interpolation'. Although the result robot motion is just mathematically interpolated, its basis functions are based on human motions initially and reflect the dynamic property of robotic system. So we regard it as natural looking and pseudo-optimal robot movement.

4. SIMULATION : CATCHING A BALL

In this section, we show the performance of proposed imitation learning method in application of learning the task to catch a ball by humanoid robot that are called 'Mahru I' and developed by KIST (B. J. You et al., 2005). When someone throws a ball to a robot, it has to generate within a short time catching motion of 7 d.o.f. serial manipulator of waist and right arm. Actually, for robot to catch a ball, it has to get ability to estimate the trajectory of a flying ball, to stretch out its arm to a prospective falling point and to grasp the ball. Nevertheless, since our interest is the human-like motion generation, we focus on how it stretches out its arm. We assume that the estimated falling point of a ball is given.

4.1 Motion Capture and Extracting Basis Functions

A man executes the motion of catching a ball 311 times with various falling point of a ball.(see Fig.6) He use only right arm and waist and do not walk or stride. The principal components and observation sequences are calculated from joint angle trajectories of the 10 movements as shown in Fig.4.

4.2 Conditions and Distance Metric

A set of 300 conditions is randomly created. Each condition contains the information of final joint angle, velocity

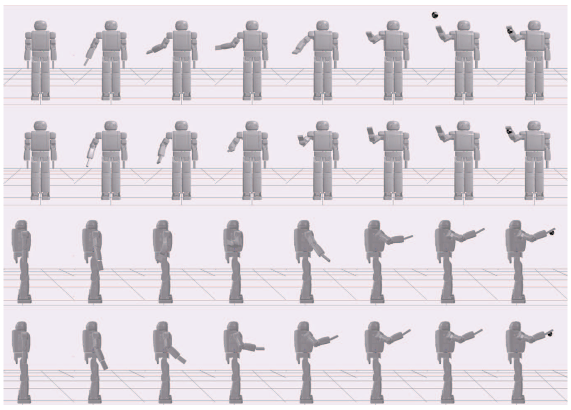

Fig. 7. Front and side view for comparison of catching motion in terms of 1th generation (the upper snapshots) and 10th generation(the lower snapshots) of movement primitive database while evolving

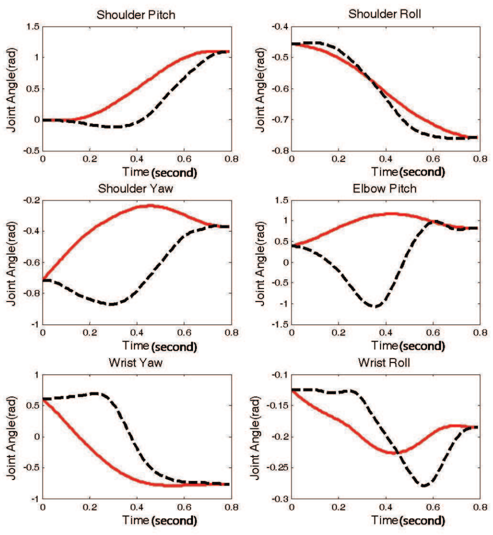

Fig. 8. The joint angles of the movement in the 10th generation (red solid line) and 1th generation (black dashed line) database. Each graph is present for right arm joint (3 shoulder, 1 elbow, 2 wrist) from upper to lower, respectively

and final position and rotation of right hand, i.e. the trunk and right arm posture when a robot catches a ball. As a matter of course, the movement in the database has its own condition. The similarity between them is determined by

$$d(c_i, c_j) = w_1\|p_i - p_j\| + w_2\|R_i^T R_j\| \qquad (6)$$

where (R_i, p_i) is the homogenous transformation matrix of c_i, and w_i is a scalar weighting coefficient. In here, we choose the movement primitives within a $15cm$ radius. In this instance, due to the orientation in catching a ball is almost equal for the movements within regular distance, we consider only the position.

4.3 Survivor Selection

For both human's property and robot's property(minimized torque) are reflected in the generated movement, we choice to survive the original movement primitive database(initial parent) for the any generation. Therefore, the any generation consists of the original movement primitive database and the survived offspring.

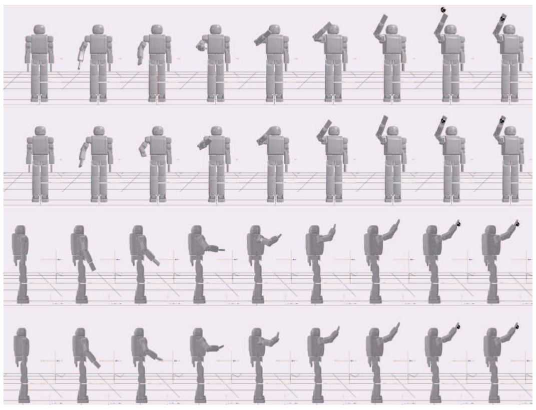

Fig. 9. Front view for comparison of catching motion in terms of 10th generation of movement primitive database (the upper snap shots: interpolation, the lower snap shots: optimization)

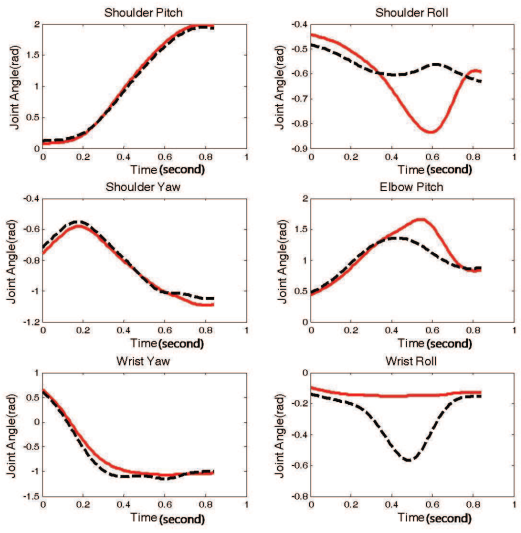

Fig. 10. The joint angles of the movement through optimization (red solid line) and interpolation (black dashed line) from 10th generation of movement primitive database. Each graph presents for right arm joint(3 shoulder, 1 elbow, 2 wrist) from upper to lower, respectively

4.4 Learned Movement Primitive Database

Our movement primitive database evolves 9 times from motion capture data to learned movements. It takes nine hours at Pentium IV computer with 2GB ram. We implement evolutionary algorithm using C++. DONLP2 library is used for local optimization at our genetic operator, which is a nonlinear optimizer library and implements sequential equality constrained quadratic programming method (P. Spellucci, DONLP2-INTV-DYN). Through survivor selection, the movement primitive database is updated gradually by replacing old ones with better offspring. Consequently, the average of required torque at each generation of evolution becomes smaller (see Fig.5). It shows that the joint angle patterns in database become optimal motions with the dynamic properties being fully reflected. The resulting motions are shown in Fig.7 and 8. The sum of require torque at all joint of the motion at $10th$ generation of evolution is $275.889N \cdot m$ and it is smaller than $370.088N \cdot m$, that of the motion at 1th generation. It shows that our learning method may work well with respect to inverse dynamics-optimization.

4.5 Online Motion Generation

Now, we can generate arbitrary motions using learned movement primitive database if the condition of the target motion is not largely different from a set of conditions used in learning system. The comparison motion generated by pure interpolation with one database by optimization using the output database of learning system is shown in Fig.9 and 10. As the learned database is sufficiently optimal, we can get human-like (meaning the robot's own dynamic property is considered) motions through simple mathematical calculation such as PCA and interpolation. As shown in Fig.10, the joint trajectory of motion by mathematical interpolation is similar to the result motion by dynamics-based optimization and has the main characteristics of the result motion by dynamics-based optimization. So we can generate pseudo-optimal motions, i.e. similar motions to optimized result without computational optimization process.

5. SUMMARY AND DISCUSSION

This paper has presented a methodology for evolving movement primitive database that enables a humanoid robot to imitate human motions as a human learns new motions and do them. Raibert mentioned that human's motion learning ability is based on the capacity for storage and retrieval of a great variety of optimal movement patterns.(M. Raibert et al., 1978) Similarly, our approach is divided into two processes: (1) the offline process for building motion database through the proposed learning procedure and (2) the online process to generate the motions in real time using the database. In the offline process, we update the movement primitive database using the proposed learning procedure with Evolutionary Algorithm, minimizing joint torques. In the online process, we generate a desired motion in real time by simply interpolating the principal components obtained from the learned movement primitive database.

We are planning to extend our methodology to include more complex motions and full-body motions in the future. It was observed that some motions from the proposed framework were not good enough to behave like human or violated some of the joint limits. For resolving this, we would like to consider joint limits, self-collision prevention and so on in the online process.

This work was supported by the IT R&D program of MIC/IITA. [2006-S028-01, Development of Cooperative Network-based Humanoids Technology].

REFERENCES

A. E. Eiben, and J. E. Smith. Introduction to Evolutionary Computing. *Springer*, pages 15–24, 2003.

A. Fod, M. J. Mataric, and O. C. Jenkins. Automated Derivation of Primitives for Movement Classification. *Autonomous Robots*, volume 12, no 1, pages 39–54, 2002.

B. Lim, S. Ra, F. C. Park. Movement primitives, principal component analysis, and the efficient generation of natural motions. *IEEE Int. Conf. Intelligent Robots and Systems*, pages 4630–4635, 2005.

B. J. You, Y. Choi, M. Jeong,D. Kim, Y. H. Oh, C. Kim, J. S. Cho, M. Park, and S. Oh. Network-based Humanoids 'MAHRU' and 'AHRA'. *Int. Conf. Ubiquitous Robots and Ambient Intelligence*, pages 376–379, 2005.

F. A. Mussa-Ivaldi, E. Bizzi. Motor learning through the combination of primitive. *Philosophical Trans. of the Royal Society of London*, Series B, 355:1755–1769, 2000.

F. A. Mussa-Ivaldi, S. F. Giszter, and E. Bizzi. Linear combinations of primitives in vertebrate motor control. *Proc. of the National Academy of Sciences of the United States of America*, 91, pages 7534–7538, 1994.

J. E. Bobrow, B. Martin, G. Sohl, E. C. Wang, F. C. Park, and J. Kim. Optimal robot motions for physical criteria. *J. of Robotic Systems*, volume 18, no 12, pages 785–795, 2001.

J. M. Hollerbach and T. Flash. Dynamic interactions between limb segments during planar arm movement. *Biological Cybernetics*, volume 44, pages 67–77, 1982.

J. Nakanishi, J. Morimoto, G. Endo, G. Cheng, S. Schaal, and M. Kawato. Learning from demonstration and adaptation of biped locomotion. *Robotics and Autonomous System*, volume 47, pages 79–91, 2004.

M. An, T. Taura, and T. Shiose. A study on acquiring underlying behavioral criteria for manipulator motion by focusing on learning efficiency. *IEEE Trans. Systems, man, and cybernetics - A*, volume 37, No 4, 2007.

M. M. Williamson. Postural primitives: interactive behavior for a humanoid robot arm. *In Proc. of the Fourth International Conference on Simulation of Adaptive Behavior (SAB-96)*, 1996.

M. Raibert and B. Horn. A model for sensorimotor control and learning. *Biological Cybernetics*, volume 29, pages 29–36, 1978.

R. A. Schmidt. Motor Control and Learning: A Behavioral Emphasis. 2nd ed. Champaign, IL: Human Kinetics., 1988

S. Lee, J. Kim, F. C. Park, M. Kim, and J. E. Bobrow Newton-type algorithms for dynamics-based robot motion optimization. *IEEE Trans. Robotics*, volume 21, no 4, pages 657–667, 2005.

S. Schaal. Is imitation learning the route to humanoid robots? *Trends in Cognitive Sciences*, volume 3, pages 233–242, 1999.

S. Schaal. "Learning robot control" in The handbook of brain theory and neural networks. *2nd ed, M. A. Arbib, Ed. MIT Press*, pages 983–987, 2002.

T. D. Sanger. Human arm movements described by a low-dimensional superposition of principal components. *The J. of Neuroscience*, volume 20, no 3, pages 1066–1072, 2000.

P. Spellucci. Donlp2-intv-dyn. *http://www.mathematik.tu-darmstadt.de:8080/ags/ag8/Mitglieder/spellucci_de.html*