The 2010 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 18-22, 2010, Taipei, Taiwan

Constructing of Optimal Database Structure by Imitation Learning based on Evolutionary Algorithm

Ga-Ram Park, ChangHwan Kim

Abstract— **This paper explores the efficient construction of the database structure for the human-like arm motion generation using an evolutionary algorithm-based an imitation learning in real-time. The framework of the arm motion generation consists of two processes, imitation learning of human arm motions and generating of a human-like arm motion using the motion database evolved by the learning process in real-time. We aim at constructing the optimized database structure which have the minimum number of one. We compare a human-likeness, similarity of a motion and robot property which minimize a sum of a robot's joint torques for three database structure. We applied our method to the task of teaching a humanoid robot how to make naturally looking movements like catching the cup on the table.**

I. Introduction

People expect that humanoid robot moves like human. In spite of a rapidly developing humanoid technology, the present situation is not enough to put to practical use in real life. Generating of motion for humanoid is particularly difficult problem because of large numbers of joints, redundancies and self-avoidance. Besides, the problem of how to make apt motions depending on the circumstances is very important issue. Generating of a human-like motion is one of the key schemes for the humanoid robot to act as a human and collaborate with people.

Imitation learning might be an efficient approach to make a robot enable to generate natural and abundant motions [1]. When someone needs to learn a new skill or a new sports motion, he/she first watches the actions of an expert and subsequently uses those actions as a seed for learning.A. Ijspeert et al.[2] presented a control policy that was defined in form of a differential equation to represent robot joint angles and end-effector's positions. The parameters and the Gaussian functions in the differential equation were determined in aspect of minimizing the trajectory error. S. Calinon et al.[3] presented a method of probabilistic representation of the movement using Gaussian Mixture Model(GMM) and Gaussian Mixture Regression for reproduction of robot arm motions. The studies above have been done for generating human-like robot motions in the robot kinematics point of view from imitation learning of human behaviors. Although the human motions contain kinematics and dynamics properties, the dynamics property of a robot was not reflected into the generated motions. Therefore, a method to involve the dynamics property as well as human-likeness in generating motions is needed.

We proposed the motion generation framework by the imitation learning using the Evolutionary Algorithm(EA).[4] The proposed framework consists of off-line process and on-line process. In off-line imitation learning, we evolved the motion database that was initially full of the motions kinematically converted from human motion capture data, considering the minimal joint torques for human motion data. For this, we also introduced a genetic operator which consists of the principal component analysis(PCA)-based a local optimization with respect to robot dynamic properties. After several evolving the database, the resultant database could contain not only the human-likeness, but also the dynamic property of a robot, which have the minimum sum of joint torques. Using the principal components of the motion primitives in the evolved database, a humanoid robot was able to generate naturally looking arm motions with requiring the minimum sum of joint torques in real-time. In this proposed framework, there are two important factors. One is to construct the database structure. The other is to evolve the database, which reflects a dynamic property of a robot.

In this paper, we discuss how to construct the optimized database structure. We design the three kinds of the database structure. One is composed the existing database structure.[4] Another is composed the clustering database structure, which divides the workspace into six clusters which is a fixed size. The other also is composed the clustering database structure, but its size of the cluster is bigger than previous thing. The number of the database's elements also has smaller than it.

The following section introduces the off-line process that is to train the obtained motion primitives so as to reflect both the properties of human and robot by using the Evolutionary Algorithm(EA)-based imitation learning. Section III then describes as to generating motions by using evolved motion database in real-time. Section IV carries out the task of generating the motion which catches the cup on the table.

II. Evolution of Motion Database Using EA-based Imitation Learning

When a humanoid robot is requested to do such tasks as moving an object on the table, grasping something on the shelf, handling a switch on the wall or an appliance, catching a ball etc., the robot needs to be able to reach the arm to the target position from the initial pose. People want the humanoid robot to perform those tasks as a human does. This work proposes a framework to learn human motions

This work was supported in part by Ministry Knowledge Economy(MKE) & Institute for Information Technology Advancement(IITA) through IT Leading R&D support project.

All the authors are with the Center for Cognitive Robotics Research, Korea Institute of Science and Technology, Seoul, 136-791, Republic of Korea mysoulmate79@gmail.com, ckim@kist.re.kr

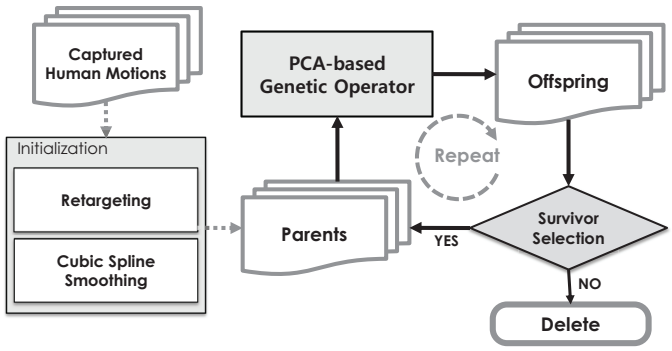


Fig. 1. Evolutionary Process

and generate a human-like reaching motion in real-time. The framework consists of two processes of imitation learning and the real-time motion generating.

To optimize the motion primitive database, we use the Evolutionary Algorithm(EA). EA is stochastic search methods that mimic the metaphor of natural biological evolution. It has several merits. To begin with, EA is possible to search a population of points in parallel, not just a single point. Second, It does not require derivative information or other auxiliary knowledge. Only the objective function and corresponding fitness levels influence the directions of search. Third, it is generally more straightforward about an application, because there are no restrictions for the definition of the objective function. Finally, it provides a number of potential solutions to a given problem.

The imitation learning constructs the motion database that contains human-likeness derived from human arm motions and is optimized for proving minimal torques. The human arm motion capture data are first converted to the motions available to the humanoid robot, considering the geometric differences between the robot and a human. More details on this conversion process refer to [5]. The motion conversion provides not only human-like motions to the humanoid robot in the kinematics sense, but also the accumulated and optimized motion experiences immanent in the human motions, since the human motions are dynamically consistent and optimized through a motion repetition for a long lifetime.

In addition to the geometric differences, there are many differences in the dynamic properties between a human and a robot like a mass, a center of mass, a type and a capacity of actuators. To involve one of robot's dynamic properties, this work requires a torque minimization for moving robot arms. For this purpose, the database of arm joint trajectories converted above is then updated by using an Evolutionary Algorithm(EA). Consequently, the humanoid robot is able to generate skillful motions using this database. In other words, the humanoid robot can learn task skills from a human. More details on the imitation learning based on EA are explored in the following subsections.

A. Evolutionary Process

The evolutionary process to update the database converted from human motion capture data is given as a typical form as seen in Fig. 1.

1) Capturing Human Motions: For a given task, an actor executes the required motions several times and those motions are captured with a motion capturing system. Each of the captured motions has its own condition. The condition denotes the factors that correspond to the human motion. For example, when the task is to grasp a cup on the table, the condition can be the pose (position and orientation) of cup. As another example, for the case of catching a ball, the condition can be the ball position and the flying direction or velocity of ball. A user defines appropriate conditions for a given task. In this step, the human arm motion capture data are converted to the motions for a humanoid robot using the method in [5].

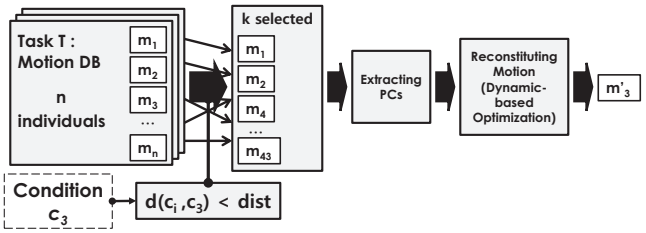
2) Parent: Parents consist of the collection of individuals, which are defined as motion primitives. The initial individuals are the motions converted from the human motion capture data and will be updated by a genetic operator and survivor selection in the Fig. 1. Therefore, the initial parents before evolution contain only the kinematic and dynamic characteristics of the actor. It is noticed that the number of individuals is the same as that of conditions, since each motion primitive has its own condition.

3) Applying of Genetic Operator: The genetic operator to create new individuals (motion primitives) from the parents is designed, aiming at satisfying user's requirement. The requirement of this study is to minimize the toques that are necessary to finish the given task. Section II-C describes the details on designing this operator.

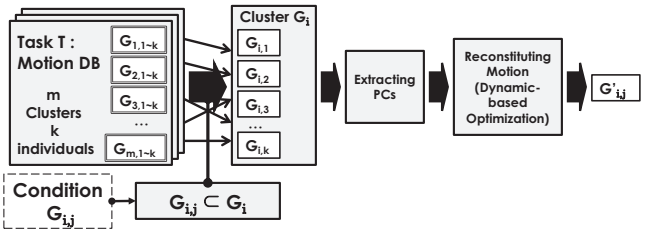
4) Offspring: As many new individuals (so called offspring) as parents are obtained, since the genetic operator is applied the same number of times as that of conditions. In other words, two individuals (motion primitives) for offspring and parent, respectively, corresponding to one of the conditions is given.

5) Survivor Selection: The offspring and the parent are compared with each other and one of them is selected in sense of most satisfying the user requirement. This selection makes the survivor become the better individual for the next generation like *the survival of the fittest*. The evolutionary process stops, if the fitness evaluation defined by a user converges into a certain limit. As an example, the fitness evaluation for the reaching problem discussed later is the minimal torques without violating joint limits.

6) Repeating: The above procedures are repeated until the value of fitness evaluation converges into a limit defined by a user. As the evolution processes, the individuals (motion primitives) gradually lose the characteristics of human motions and finally become adapted to the physical feature of the humanoid robot. For this purpose, a proper number of generations of the evolution need to be chosen.



(a) For general database structure



(b) For clustered database structure

Fig. 2. PCA-based Genetic Operator

B. Clustered Database Structure

The database structure is an important problem in the proposed imitation learning. A memory of the humanoid robot has a fixed volume. The humanoid robot also has to carry out the many tasks in a human environment.

In previous paper, we construct the database structure which is uniform in an interval. This database structure provides a stable and an excellent performance. However, it is not efficient. If the tasks of humanoid robot is increased, it will be needed much memory.

Our aim is to generate naturally looking motions by using the minimum and optimal database structure. Generally, In the fixed area, the motions of human being are very similar to each other for a same task. By deriving inspiration from the fact, we compose the clustered database structure. Its size and the number of individuals for one cluster are decided by user. In this paper, we designed the three database structure. One is a existing database structure and the others are a clustered database as you see in Fig. 5.

C. PCA-based Genetic Operator

We defined a motion primitive itself as an individual before. Because a motion primitive is the vector of joint angle trajectories in time, the genotype of an individual in EA is represented by real-valued vectors. For instance, if a motion primitive is a 2 minute joint angle trajectory with sampling rate 120Hz, its genotype is 14400-dimensional real-valued vector (14440 = 2 min $\times$ 120 Hz $\times$ 60 sec). And if the motion primitive also has 3 degrees of freedom, its genotype consists of three of 14400-dimensional vector.

The genetic operator proposed for the evolution in this work is shown in Fig. 2. In Fig. 2(a), it is a kind of recombination operator, since it merges information from several genotypes into one offspring genotype. When there are a set of n motion primitives for a task T, the i^{th} motion primitive m_i in the set has its own condition c_i. Assuming that we need the motion to satisfy with the condition c_3, then we select k motion primitives whose conditions are analogous to the condition c_3 from the set of n motion primitives. the condition means a target position of DB motions. The analogousness is determined by a suitable distance metric $d(c_i, c_3)$. If the condition is the 3-dimensional position vector, the Euclidean distance between two conditions may be a good distance metric to determine their similarity.

In Fig. 2(b), it uses the clustered database structure. If we generate the motion satisfying the condition $G_{i,j}$ and the condition is included in the cluster G_i, we select the k individuals in G_i. The rest of the entire process is the same above.

By applying Principal Component Analysis (PCA) on the selected k motion primitives, the sample mean and the principal components of joint angle trajectories are obtained. This analysis is performed for each joint of arm. The principal components are used as the basis functions in order to reconstitute a robot motion satisfying the condition c_3. PCA projects high-dimensional data onto a lower-dimensional subspace in a way that is optimal in a sum-squared error sense. We use it to extract the dominant principal components(PC) from a group of motion primitives. The dominant PCs have the highest eigenvalues during PCA. High rank four PCs occupy more than 90% on the related contribution.

An arbitrary joint angle trajectory is described by a low-dimensional superposition of principal components as follows:

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

where $q(t)$ is a joint trajectory, $q_{mean}(t)$ is a mean trajectory, $q_{pc_i}(t)$ is the i^{th} most dominant principal component and $\mathbf{x} = [x_1 \;\; x_2 ... x_5]^T$ is an unknown scalar weighting coefficient vector. Especially, x_5 denotes an unknown constant to represent a remaining error. The boundary conditions at the initial time, t_0, and the final time, t_f, are given as follows :

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

These boundary conditions are not sufficient to determine five unknowns x_i. So we introduce local optimization with respect to robot dynamics. The equations of motion for a robot, which are modeled as a coupled rigid body system with s joints, are given as

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + N(q, \dot{q}) = \tau \tag{3}$$

where $M(q) \in \Re^{s \times s}$ is the mass matrix, $C(q, \dot{q}) \in \Re^s$ is the Coriolis vector, $N(q, \dot{q}) \in \Re^s$ includes the gravitational and other forces, and $\tau \in \Re^s$ is the joint torque vector. To acquire optimal motions with respect to some physical criterion (in this paper, we are intent on minimum torque motion), we interest in minimizing the objective function in the following form subject to (3) :

$$\min_{\mathbf{x}} \frac{1}{2} \int_{t_0}^{t_f} \left\| \tau(q, \dot{q}, \ddot{q}) \right\|^2 dt \tag{4}$$

where the boundary conditions in (2) are also used as optimization constraints. Through descent-based optimization method with analytic gradient and hessian information [6][7], the offspring corresponding to c_3 is constructed for the next generation.

The offspring m_3',$G_{i,j}'$ in Fig. 2 has the same condition c_3 as the parent m_3. They, however, might be different, since the offspring m_3' is made up from several individuals (i.e. motion primitives) including the parent m_3. In other words, the offspring might be another motion suited to the condition c_3. The offspring m_3' competes with its parent m_3 for the next generation. One of them is selected as the survivor selection component. The genetic operator is applied repeatedly from a condition c_1 to a condition c_n, so that as many offspring as parents are made.

As a quality criterion for the evolutionary process, the fitness function is defined as the total sum of the torques to operate all the motion primitives that correspond to all the conditions. When a value of fitness function becomes less than before, the loop is stopped it in the evolutionary process. Using this fitness function we may gradually evolve the motion primitives into torque efficient and optimal ones. At each generation of the evolutionary process, the optimal motion primitive, which requires the minimal torque, for each condition is obtained. The optimal motion primitives construct a new motion database(next generation) that has better properties than the previous generation. In other words, through this process the Evolutionary Algorithm(EA) works as a global optimizer and the genetic operator works as a local optimizer for evolving motion primitives into torque-efficient and human-like ones.

III. Real-time Motion Generation Using Evolved Motion Database

A. Real-time Motion Generation

When a task with a condition, c, is given, we can generate an appropriate motion in real-time using the motion database evolved in Sec.II, the boundary conditions in (2) and the following equation,

$$q(t) = q_{mean}(t) + \sum_{i=1}^{3} x_i \; q_{pc_i}(t) + x_4 \qquad (5)$$

where all the variables and boundary conditions denote the same as (1). Since the unknown terms are four, we can obtain the solution of the equation above by using the boundary conditions. PCA here is done in the same manner as in Sec.II-C using the distance metric $d(c_i, c)$ for $i = 1 \sim n$. The resultant joint angle trajectory from (5) can reflect both of the human-likeness and the dynamic property of minimal torque, since the motion primitives already contains such characteristics from the evolutionary process and the human motion capture data.

IV. Experiment

We simulate that a man catch the cup on the table. And we designed the three database structure. In Fig.5(a),

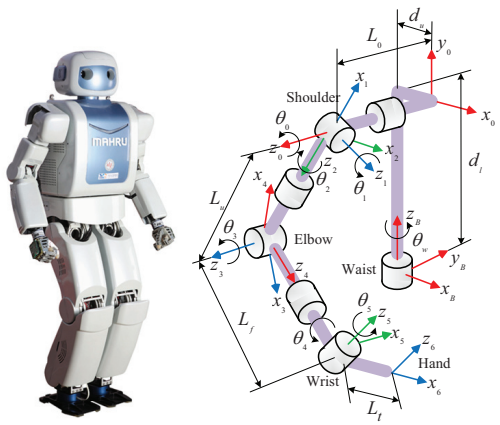


Fig. 3. Coordinates of 6-DOF arm of *MAHRU* with waist

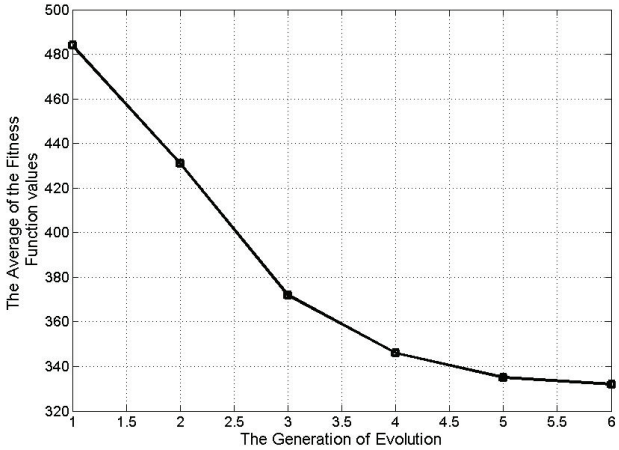
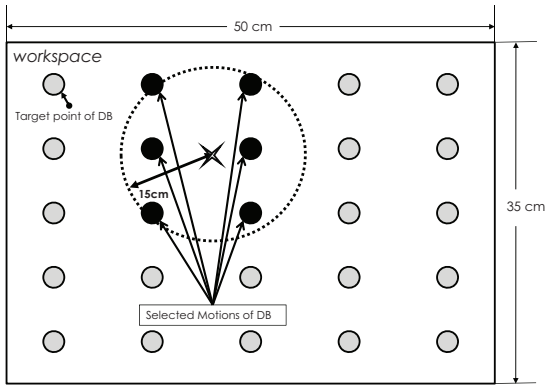
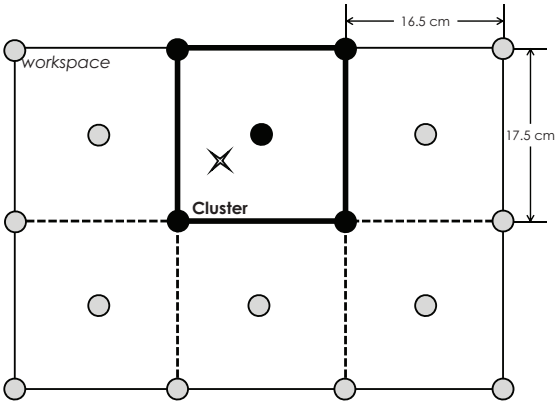


Fig. 4. The average of total torque sum at each generation

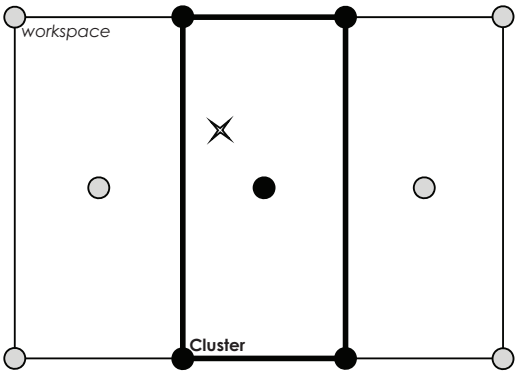
The man is captured motions for 30 times. In Fig.5(b),(c), each cluster include three individuals which is a motion data. The size of Fin.]reffig:st(c) is just lager than Fig.5(b). Using the proposed imitation learning, the motion primitive database was evolved up to the 6th generation. Through the imitation learning procedure, the motion primitive database was updated gradually by replacing old ones with better offspring. It is observed as seen in Fig. 4 that the average of total torque sum at each generation of evolution becomes smaller as the generation goes. This means that the database contains better optimal motions in the aspect of human-likeness and minimal torque. Imitation learning took five hours at Pentium IV computer with 2GB ram. We implemented the imitation learning using C++. DONLP2 library, a nonlinear optimizer library, was used for local optimization in the genetic operator and implemented as a sequential equality constrained quadratic programming problem[8]. As you see in Fig. 6, Fig. 7, we compared with four trajectories for the general database structure, the clustered database structure type1 and the clustered database structure type2. As you can see, the trajectories of the general database structure



(a) general database structure



(b) clustered database structure type1



(c) clustered database structure type2

Fig. 5. PCA-based Genetic Operator

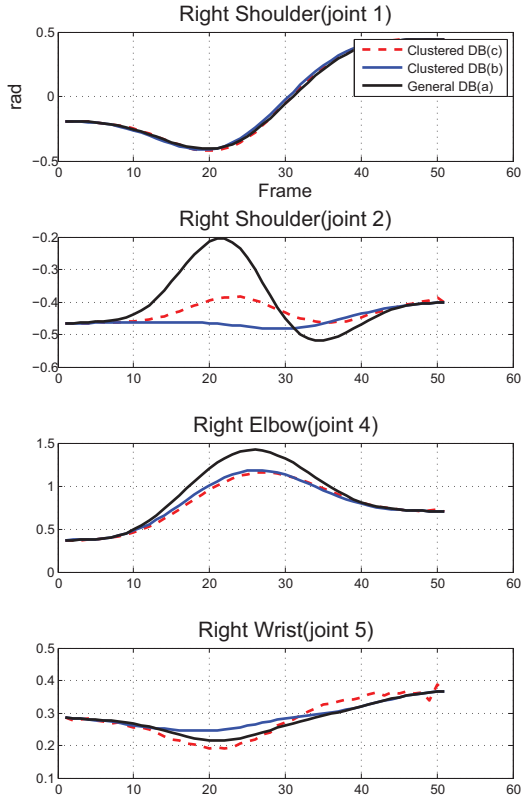


Fig. 6. The trajectory for desired target position A

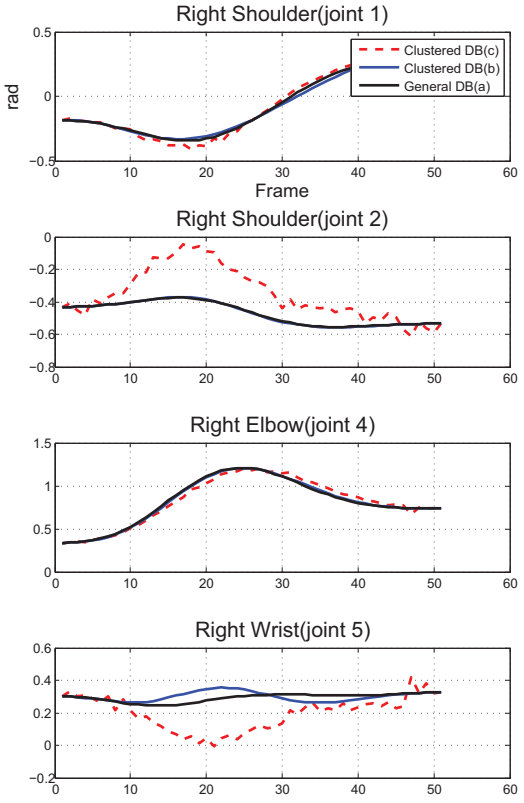


Fig. 7. The trajectory for desired target position B

and the cluster database structure type1 has a similar shape. For the clustered database structure type2, we can see the motion is very rough and strange. It is a cluster size that is a important factor. If the size is very large, then a generated motion is strange or rough.

V. Discussion

We presented a clustered database structure for the imitation learning based on the evolutionary algorithm which enables a humanoid robot to learn human motions as a human learns new motions and performs them. The clustered database structure is that the workspace of a humanoid robot is divided into the fixed area. The fixed area has minimum individuals.

In a simulation, we evolved the motion database that was initially full of the motions kinematically converted from human motion capture data, considering the minimal joint torques for human motion data. For this, we introduced a genetic operator, which consisted of PCA-based local optimization with respect to robot dynamics. After several generations of evolution, the resultant database could contain the human-likeness and the dynamics characteristics of minimal joint torques. Using PCA on the motion primitives in the evolved database, a humanoid robot was able to generate the human-like reaching arm motion with requiring minimal torques in real-time.[4] To optimize the number of motion primitives of database, we designed the cluster database structure into the workspace.

Through the simulation, we compared the general database structure with the clustered database structure. As a result, we know that the clustered database structure has not only a similar trajectory with comparing the general database structure, but also an efficient structure.

Although the proposed method, which is a clustered database structure, is a good method to decrease the size of database, there are several weaknesses such as the stability of generated motions within the sector of cluster, the limit of joint, the collision avoidance. The research to solve them is in progress. In the future work, we will deal with the stability, the limit of joint, modifying the trajectory on moving the robot.

References

[1] S. Schaal, "Learning robot control," in *The handbook of brain theory and neural networks, 2nd Edition*, M. A. Arbib, Ed. Cambridge, MA.: MIT Press.

[2] A. Ijspeert, J. Nakanishi, and S. Schaal, "Movement imitation with nonlinear dynamical systems in humanoid robots," in *Robotics and Automation, 2002. Proceedings. ICRA '02. IEEE International Conference on*, vol. 2, 2002, pp. 1398 –1403.

[3] S. Calinon, F. Guenter, and A. Billard, "On learning, representing, and generalizing a task in a humanoid robot," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 37, no. 2, pp. 286 –298, april 2007.

[4] G.-R. Park, K. Kim, C. Kim, M.-H. Jeong, B.-J. You, and S. Ra, "Human-like catching motion of humanoid using evolutionary algorithm(ea)-based imitation learning," 27 2009-oct. 2 2009, pp. 809 –815.

[5] C. Kim, D. Kim, and Y. Oh, "Adaptation of human motion capture data to humanoid robots for motion imitation using optimization," *Integrated Computer-Aided Engineering*, vol. 13.

[6] 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*, vol. 21, no. 4, pp. 657–667, 2005.

[7] J. E. Bobrow, B. Martin, G. Sohl, E. C. Wang, F. C. Park, and J. Kim, "Optimal robot motions for physical criteria," *Journal of Robotic Systems*, vol. 18, no. 12, pp. 785–795, 2001.

[8] P.Spellucci, "Donlp2-intv-dyn." [Online]. Available: http://www.mathematik.tu-darmstadt.de:8080/ags/ag8/Mitglieder/spellucci_de.html