An Efficient Method for Composing Whole Body Motions of a Humanoid Robot

Shinichiro NAKAOKA[†] Atsushi NAKAZAWA[‡] Katsushi IKEUCHI[†]
[†] *Institute of Industrial Science, The University of Tokyo*
{nakaoka, ki}@cvl.iis.u-tokyo.ac.jp
4-6-1 Komaba, Meguro-ku, Tokyo, 153-8505, Japan
[‡] *Cyber Media Center, Osaka University*
1-21 Machikaneyama, Toyonaka, Osaka 560-0043, Japan
nakazawa@ime.cmc.osaka-u.ac.jp

Abstract. This paper describes a method for composing whole body motions of a humanoid robot. Recently there have been many studies of controlling a humanoid robot, and robots have been acquiring the ability to perform various tasks. However, it is still complicated work for us to create complex whole body motions like humans. In order to solve this problem, we propose a motion composition method that uses primitive motions. A whole motion sequence is decomposed into several kinds of primitives. By acquiring primitives from a human motion, or editing primitives manually, we can easily generate a motion that can be performed by a robot. We have developed a software platform for motion composition, and generated robot performances of traditional folk dances based on human performances.
Keywords: humanoid robot, human body motion, motion primitive, robot simulation

1. Introduction

Recently, development of humanoid robots has been increasing [1][2][3]. In addition to hardware developments, there have been numerous studies about controlling whole body motion of a robot. However, designing a whole body motion of a humanoid robot remains a difficult problem because the body structure of humanoid robot is complex (the robot has a high degree of freedom) and a robot must perform under mechanical constraints and laws of physics. The purpose of this study is to develop a method which enables us to efficiently compose human-like whole body motions. By using this method, we have realized traditional folk dance performances by a humanoid robot.

When we try to enable a humanoid robot to move, we generally have two purposes in mind. One is to make a robot do some practical tasks, and the other is to make it perform a particular style of motion itself. This classification is associated with the discussion of why humanoid robots are required. One answer is that it is easy to work in human life environments if a robot has a human-like body rather than another kind of body. Another answer is that a humanoid robot is necessary for expressing those motions that only humans could express.

There are many studies about practical tasks. Humanoid Robotics Project (HRP) [4] has developed robots that can work for humans. One task example is the cooperative task of moving building materials [5]. This kind of practical task requires a combination of various basic motions such as walking, picking up some objects, lying down, getting up [6], and so on. For walking, the methods for generating stable walking patterns to follow arbitrary footprints are proposed [7][8]. These methods provide one of the most fundamental facilities for a humanoid robot. Other basic motions have been studied and robots have been able to perform various motions. However, appearances of these kinds of basic motions are usually simple and patterned because these methods do not emphasize motion appearance so much.

Generally, in these basic motions, the factors to be considered important are practical ones such as stability, safety and efficiency. For example, margins for joint angle limit or actuator's limit are taken enough, or energy consumption in walking motion is optimized [9]. These factors are important for practical use, but the resulting motion tends to be conservative, patterned and machine-like.

On the other hand, as mentioned above, there are studies whose purpose is to make a robot perform a particular style of motion. In this case, appearance of motion is emphasized rather than safety or efficiency in practical use. Entertainment use is one application for this purpose. Sony [2] has developed a small sized robot for entertainment and its control scheme has been proposed [10]. This robot can perform whole body motions such as dances, but a detailed method of composing motion is not well described. In general, manually creating complex whole body motions is difficult work. One solution is to acquire motion from human performances. Pollard et al. [11] realized a dancing robot by using a human motion acquired by a motion capturing system. However, their robot was fixed at the waist. Supporting the body by its own legs was not considered. Using human motion for leg motions of a robot is more difficult problem because a foot must contact stably with a floor and maintain body balance.

In the field of computer graphics, synthesizing human motion has been aggressively studied. There are various kinds of approaches as follows: editing motion manually by an efficient user interface [12], synthesizing motion by signal processing [13], generating motion by optimization with spacetime constraints [14], motion generation considering physical law [15] and re-composition from captured human motions [16]. These studies are suggestive for humanoid robots, but most methods cannot be used directly for a humanoid robot, because it is sufficient for computer graphics (CG) animation that motion is seen as almost natural by a human. CG animation does not strictly consider dynamics and body constraints. However, these factors must be strictly considered to move a humanoid robot.

To sum up, our purpose is to create not CG animation, but rather motion of a real humanoid robot, and not patterned motion but various styled motions.

2. Overview of the Composition Method

2.1 Importing Human Motion

Our system enables the use of motion data captured from a human. In the field of computer graphics, using captured data is one of the standard ways to create human body animation. In contrast to CG animation, a robot cannot completely follow captured motion data because the robot must move under its mechanical constraints and physical law. However, using the captured data as base motion is still effective for a robot.

Motion capturing systems track several markers attached to a human body so that the captured motion is expressed by discrete position sequences of the markers. This data is not directly edited, but is used for creating motion elements of a robot described below.

2.2 Available Motion Elements

Our system provides three elements for motion editing: key poses, joint sequences and motion primitives. A user can use the appropriate one according to the situation.

A key pose represents a particular characteristic pose, or an intermediate pose of a motion sequence. Key poses can be created by both editing manually and importing from a captured human motion.

A joint sequence represents a particular motion of a particular body part (particular joints). This data can be acquired by converting from a captured human motion, or interpolating key poses.

Motion primitive is a key feature of our system. A primitive represents a unit of basic motion. Primitives can be detected from a captured human motion or created manually. By editing sequence order and some parameters of each primitive, a user can compose robot motion effectively. Since primitives are designed to create a robot motion, a feasible robot motion can be easily created. On the other hand, although key poses and joint sequences are conventional elements for creating CG animation, such data can not always be used directly for a humanoid robot. Motion primitives are more reasonable elements for composing robot motions. The details of motion primitives are described in Section 3.

In the current system, joint sequences are usually used to create upper body motion, and motion primitives are usually used to create lower body (legs) motion. Key poses are used to edit both joint sequences and motion primitives.

2.3 Motion Mapping onto a Robot

After joint and primitive sequences are edited, robotic motion can be generated from them. According to the constraints of a target robot, motion elements must be constrained in mapping.

For joint sequences, the data must be under the angle limit and angular velocity limit of each joint. To solve this problem, we employ the method proposed by Pollard et al. [11].

For motion primitives, robot motion is generated according to its parameters. Sometimes parameters are tuned when they give problems such as self collisions and over-limits of joint properties.

In mapping, dynamic body balance is a significant factor to consider. If the generated robot motion satisfies the mechanical constraints of the robot but does not satisfy the dynamics, the robot falls down. As the final process of mapping, robot motion is slightly modified in order to satisfy dynamics. This process requires calculation of dynamics property and how to modify robot motion to acquire balanced motion. In Section 4, we describe this topic.

2.4 Integrated Software Platform

We have implemented our method on integrated software for robot motion composition (Fig. 1). This software has user friendly GUI and 3D visualization viewer. A user can efficiently compose robot motion with help of graphics. Edited robot motion is instantly certified by kinematics simulation with contact constraints (described in Section 4).

The software can read standard formats of motion capturing systems. Captured data can be viewed by 3D CG animation. The format of robot model is the same as the model format defined by OpenHRP [17]. That format is based on H-Anim format [18] and extends it. Motion for any humanoid robot defined by this format can be generated. The software can also read 3D scene models as environments and backgrounds. It is useful for simulating performances on a stage. A user can simulate edited motions in combination with several robot models, captured human models, and environments.

3. Motion Primitives

Motion Primitive is a key feature of our composition system. In this section, we describe the concept of motion primitives, defined primitives of the current system, recognition from captured human motion and generation of robot motion from the primitives.

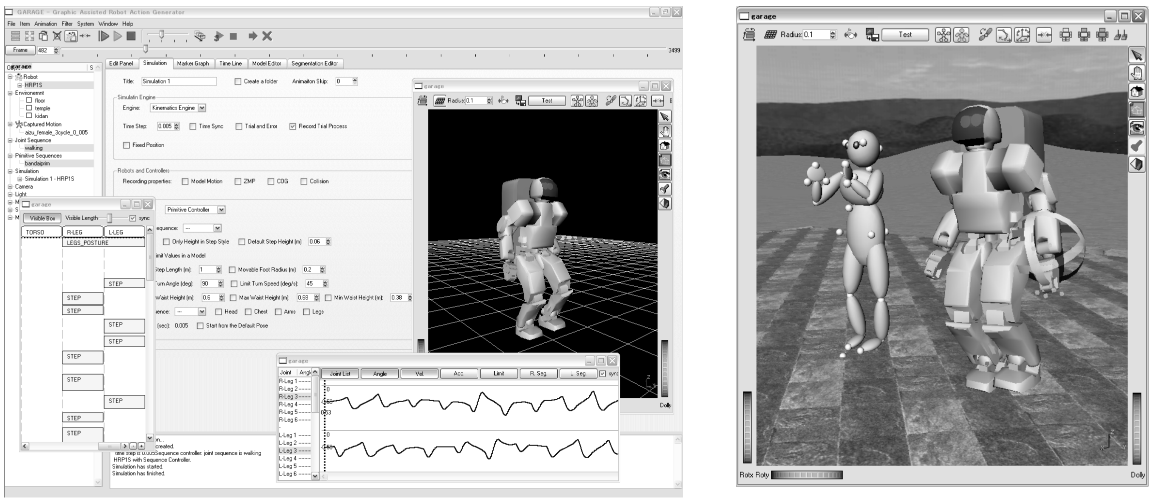
on...
created.
time step is 0.005Sequence controller: joint sequence is walking
HRP1S with Sequence Controller.
Simulation has started.
Simulation has finished.

Figure 1. Integrated composition software developed by us

3.1 Concept of Motion Primitives

Motion primitive is a concept for representing a basic unit of motion. Primitives are described symbolically by a sort of motion and some parameters which express characteristics of that sort of motion. A whole motion sequence is decomposed into a sequence of several primitives, and the primitive sequence can compose the whole motion sequence again. Usually motion data which directly represents human body motion is a joint angle sequence. By comparing primitives with joint angle sequences, merits of primitives and why primitives are required can be clearly explained.

First, a motion expressed by primitives is easier to understand than that of a joint angle sequence. A user can recognize a unit of motion, and characteristics of each unit are clearly expressed by some key points. These features enable motion editing in an easy to understand way.

Second, primitive-based motion can be used for multi-purpose. The same primitive sequence can be applied to any kinds of humanoid robots which may have different joint structure, degree of freedom, and actuator's capacity. According to situations, moving speed and scale can be easily changed because primitives just express key points of motion. Robotic motion can be generated under body structure and constraints of the robot. The robot should just express key points as much as possible in this time. In other words, while a robot has various constraints, it can reflect users' intention in performance. If motion is expressed by joint angle sequence, modifying motion data in order to adapt robot constraints and situation becomes more difficult. If adaptation were to be completed, the resulting motion may lose important characteristics that a user desired. In order to preserve those characteristics, key information like those primitives have is required after all.

Motion primitives might seem a similar concept of key poses in CG animation. Different from key poses, primitives do not necessarily have a particular pose. There is a case that a primitive represents a functional motion such as supporting the upper body. If a primitive is associated with a particular pose, the primitive does not necessarily have a whole body pose but key points of the pose such as the end of a foot. In addition, primitives have other properties such as timing and balance points.

In summary, motion primitives are required for (1) easy and efficient motion editing and (2) generating feasible motions of a robot, which has various constraints, without losing important motion characteristics.

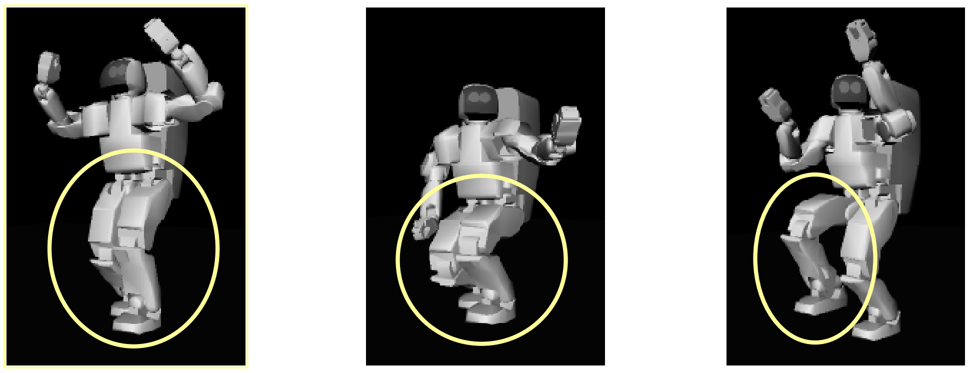

Figure 2. Defenition of Leg Motion Primitives.

3.2 Defined Primitives

In the current system, primitives for leg motion are defined. Figure 2 shows images of three defined primitives.

The *STAND* primitive represents standing motion. In this motion, the upper body is supported by both legs and body balance is maintained. This primitive has a parameter of *Zero Moment Point (ZMP)*. The concept detail of ZMP is described in Section 4.2. At this time, consider ZMP to be a point of a floor on which a robot put its weight. When the upper body is supported by both legs, a pose becomes slightly different depending on whether the body weight is put on the right foot or left foot. ZMP parameter is used for representing those differences.

The *STEP* primitive represents one stepping motion. That is, one leg is swinging up and it returns back down to the floor again while another leg is on the floor. One leg is called *swing leg* and the other leg is called *support leg*. In this primitive, two timings are considered as key states. One is when the swing leg reaches the highest position and the other is when the swing leg gets back down. For these two timings, position and orientation of the swing leg is expressed on the coordinate of the support foot, and these timings are also stored as a parameter. Parameters about the initial pose are not required because the primitives inherit the last pose of the previous motion as their initial pose. Although the STEP primitive has only two states as parameters, it can express various styles of stepping motions. The STEP primitive does not have a ZMP parameter like STAND because ZMP in a stepping motion is always underneath the support foot.

The *SQUAT* primitive represents one squatting motion. A key state of this primitive is the pose when the waist takes the lowest position. This primitive has timing and the waist height of the lowest position, and timing when the body takes the previous pose again.

For the above leg primitives, maintaining upper body balance is a necessary factor when a robot performs on a floor. For the STEP primitive, a ZMP parameter is associated with body balance. For other primitives, the ZMP can be automatically determined and used for maintaining balance. Details of this process are described in Section 4.

Other leg primitives such as JUMP should be required for expressing more variety of motions. In addition, although primitives for upper body motion are not defined currently, they will extend the efficiency and usefulness of our system. These are topics for our future work.

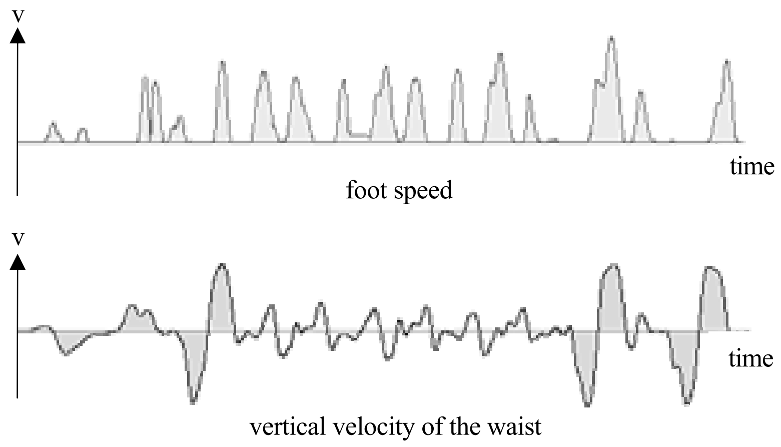

Figure 3. Graphs for primitive detection.

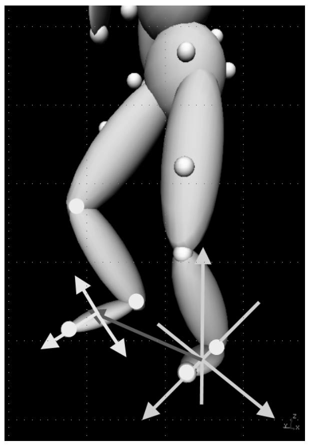

Figure 4. Paremeter Extraction of a Swing Foot in STEP

3.3 Detecting Primitives from a Captured Human Motion

Although a primitive sequence can be created manually from scratch, it can also be created from a captured human motion. If users have examples of human motion, it is easier for them to create complex robot motions. Once primitives have been acquired, they can be edited, if necessary.

By analyzing some captured markers which are associated with a particular primitive type, primitives and their parameters are detected. Detection is basically processed as follows. First, each primitive segment is detected by analyzing the speed of particular markers. Then primitive parameters are extracted by analyzing spatial correlation of associated markers.

For STEP primitives, their segments are detected by analyzing the speed of the foot (the end of a leg) marker. The upper graph of Fig. 3 is an example of STEP detection. In this graph, the areas where foot is moving are represented as convex shapes. In these areas, the foot is basically swinging. In other areas, the foot is basically supporting the body at the same position. Thus these convex shapes in the graph are detected as segments of the STEP primitive. Note that short sliding motions of a foot also appear to be the same shape. These segments should be eliminated by an appropriate threshold about integrated area size (moving distance).

After the segments are detected, a parameter set of each segment is extracted. First, position and orientation of the support foot are extracted as shown in Figure 4. Then position and orientation of the swing foot are extracted for two timings (the highest and getting down) and these values are expressed as relative values on the support foot coordinate.

STAND primitives are detected as the segments where STEP primitives are not detected. If a force sensor plate is available on capturing, the ZMP parameter is determined from output of the force sensor. Otherwise, weight distribution of the human model is estimated and ZMP value is approximately calculated from that model.

For the SQUAT primitive, a segment is detected by analyzing a vertical velocity of the waist motion (the lower graph of Fig. 3). In this graph, squatting motions appear as a pair of lower curve (squatting down) and upper curve (getting up again). This kind of curve pair is considered to be a SQUAT segment. Like STEP primitives, small areas are eliminated because these are not squatting but rather just a small swinging. The lowest waist height is determined from the waist markers at the timing when the analyzed curve changes from lower to upper. Note that the SQUAT primitive can be overlapped with other primitives.

4 Generating Robot Motion

4.1 Basic Generation Process

As mentioned in Section 2.2, there are three elements for composing motion; key poses, joint angle sequences and motion primitives. Key poses are used just for editing the latter two elements. A controller of a robot can read a combination of joint angle sequences and a primitive sequence, and the controller generates integrated robot motion from them.

For both motion elements, it is important that the motion is limited within the constraints of the robot such as joint angle range, angular velocity limit, and foot contact condition. Foot contact condition means that a support foot must make contact with a floor in completely level form. Otherwise, the robot must fall down because the feet of current robots are rigid bodies with flat soles, their feet are stable only when their whole sole plane contacts the floor. In addition to these constraints, the motion should not involve self collisions.

In terms of joint angle sequences, a sequence is filtered so that it does not run over the joint angle limit and angular velocity limit. We employ the method proposed by Pollard el al. [10] for this process.

In terms of motion primitives, robot motion is basically generated by interpolating states the primitive has. For example, the STEP primitive has two states. One is the timing where the swing foot takes the highest position, and the other is the timing where the swing foot gets down to the floor. By interpolating these two states from the last states of the previous primitive, the trajectory of the swing foot is generated. This is just the trajectory of the end of the foot. Then, all the joint angles are calculated by inverse kinematics. This scheme enables adaptation to any kind of robot models.

The constraints are solved as follows. First, a primitive-based process satisfies the contact condition naturally because the primitive parameters are originally designed so that the feet cannot break the contact condition. In other words, for the final state of the STEP primitive, position and orientation of a swing foot can only take level values with a floor. Self collisions and joint limitations are solved by tuning the state. Self collisions usually occur around the neighbourhood of key states. If a collision occurs during the interpolation, the collision can be eliminated by slightly translating the position parameter of the key states away from the collision. In a similar way, joint limitations are solved. After this modification, robot motion becomes as similar to the original motion as possible under constraints of the robot. Figure 5 shows an example of generating the STEP primitive.

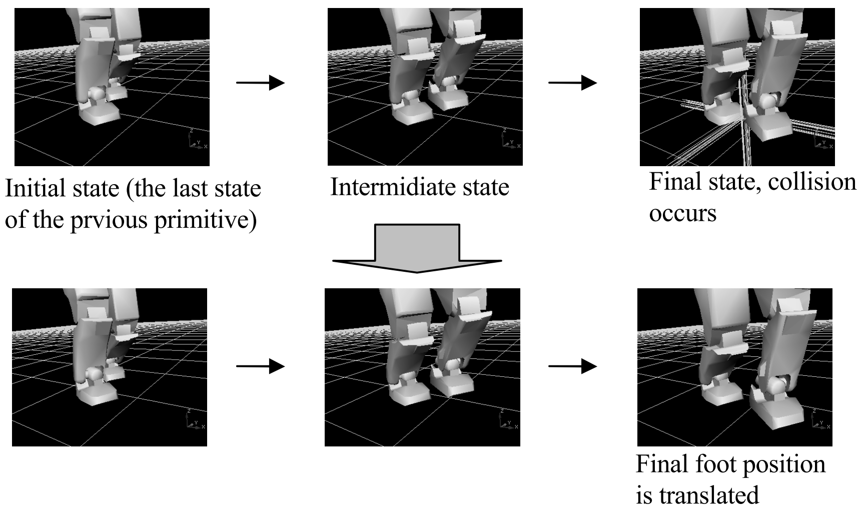
Initial state (the last state of the prvious primitive)
Intermidiate state
Final state, collision occurs
Final foot position is translated

Figure 5. STEP motion is generated by interpolating three states. In this case, The final state is slightly modified by the system in order to avoid collision.

4.2 Balanced Motion

The motion generated by the above process does not necessarily satisfy dynamic force balance. If the dynamics is not satisfied, the robot cannot maintain its balance and falls down. When we consider this problem, there is a useful concept called ZMP [19].

ZMP is a point where total vertical moment of a robot body is zero on the floor. In the real world performance, ZMP is considered as the representative point of all the reactive force the robot soles get from the floor. ZMP is always inside the convex area which supporting soles (contacting soles) make (called *suppor convexity*). On the other hand, if a particular motion is prepared and it is assumed that a supporting sole always makes contact with the floor to realize the desired motion, the theoretical ZMP for the assumed motion can be calculated. There is a case that the theoretical ZMP is out of the support convexity. This means the supporting sole rotates from the floor and the robot cannot follow the assumed motion if the robot is in the real world. On the contrary, if the theoretical ZMP satisfies the condition that ZMP is always inside the support convexity, the assumed motion will be performed as desired even in the real world. Therefore, to make the motion which satisfies this condition is the problem.

Nishiwaki et al. [20] have proposed a method to solve this problem. We have to prepare the original robot motion, a theoretical ZMP trajectory calculated from that motion and a desired ZMP trajectory which satisfies the condition. Then the method gives the horizontal translation value of the whole body to realize the desired ZMP. In fact, since it is impossible to translate the whole body, this modification approximates translation of the upper body and the modification is iterated.

At this time, our problem is how to make a desired ZMP trajectory. A desired ZMP can be determined from the motion primitives because distinction between a swing leg and a support leg is clearly described in the primitives for legs. Therefore, ZMP should be basically under a support foot. In the case that a support leg and a swing leg exchange positions, ZMP should move from the previous support foot to the next support foot. If the both legs are in a support state (in a STAND primitive), ZMP moves to the point determined by the ZMP parameter.

4.3 Efficient Simulation

For efficient motion composition, it is important that a user can quickly see a composed robot motion in visual form. In general, playback of the motion requires simulation, but dynamics simulation will take a long time. On the other hand, if playback is done only by kinematics calculation, a user can see the result more quickly but animation takes a form that a particular body part of a robot is fixed. For example, when the robot tries to perform walking motion, the robot does not walk but just flaps its legs in the same place. In this case, a user cannot confirm a realistic motion sufficiently.

This problem is solved by kinematics simulation with contact constraints. If the leg motion is composed of motion primitives, natural playback with movements in the world coordinate can be realized efficiently only by kinematics calculation. Since the system knows which leg is a support leg, the support foot should be taken as the root of kinematics calculation. In this case, the support foot is fixed on the floor and all the other body parts move on the world coordinate. As a consequence, the robot can move walking on the simulation.

This method is efficient but there are some restrictions. First, this method cannot apply to motions which include sliding of a contact area. This is not so significant because sliding of a foot is difficult in terms of current hardware ability. Another restriction is that this method does not consider running and jumping motions. This problem can be solved by calculating motion in the air from the positions before jumping and after jumping.

5. Experiments

We used our composition system to realize dance performances of a robot. Target dances were the Tsugaru Jongara-Bushi dance and the Aizu Bandaisan dance. Both are Japanese folk dances. We used a motion capturing system to acquire base motions. For Aizu Bandaisan, we captured several performances by different dancers: male and female, a dance professor and a beginner. It is desirable that the composition system can express individual characteristics of dance motions. After composition on this system was completed, we verified the motion on dynamics simulation. We used OpenHRP [17] for this purpose. Finally, we moved a real humanoid robot HRP-1S [21]. Currently, Jongara-Bushi dance was tested and the robot successfully performed the dance. Figure 6 shows the composed motion of Jongara-Bushi and the real performance by HRP-1S.

6. Conclusion

This paper described a method for efficiently composing a whole body motion of a humanoid robot. We have implemented this method into an integrated software platform. By using it, we realized human-like dance performances by a humanoid robot, based on human dance performances.

In the current system, motion primitives are defined only for lower body motions. In order to extend the merits of this method, motion primitives for upper body motions should be defined and made available. Furthermore, primitives should be refined to satisfy demand and sufficiency for expressing various human motions. For this purpose, we are going to capture many kinds of dances by several dancers, and test those motions on our system.

Acknowledgement

The robot model data of HRP-1S was provided by the Humanoid Robotics Group, Intelligent Systems Institute, National Institute of Advanced Industrial Science and Technology. Experiments on the HRP-1S were also performed in cooperation with them.

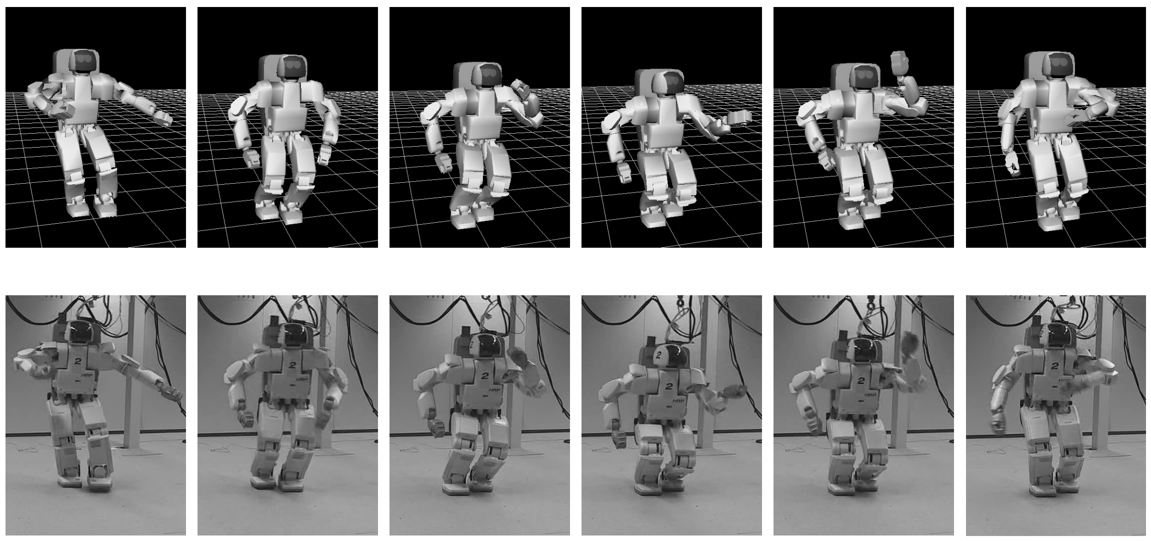

Figure 6. The composed motion (upper) and the performance by a real robot (lower).

References

[1] K.Kaneko, F.Kanehiro, S.Kajita, H.Hirukawa, T.Kawasaki, M.Hirata, K.Akachi, T.Isozumi, Humanoid Robot HRP-2. In Proceedings of the 2004 IEEE International Conference on Robotics and Automation, New Orleans, LA, USA, April, 2004.

[2] Y.Kuroki, M.Fujita, T.Ishida, K.Nagasaka and J.Yamaguchi, A Small Biped Entertainment Robot Exploring Attractive Applications. In Proceedings of IEEE International Conference on Robotics and Automation, Taipei, Taiwan, September, 2003.

[3] Y.Sakagami, R.Watanabe, C.Aoyama, S.Matsunaga, N.Higaki and K.Fujimura, The intelligent ASIMO: System overview and integration. In Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, Lausanne, Swizerland, October, 2002.

[4] H.Inoue, S.Tachi, K.Tanie, K.Yokoi, S.Hirai, H.Hirukawa, K.Hirai, S.Nakayama, K.Sawada, T.Nishiyama, O.Miki, T.Itoko, H.Inaba and M.Sudo, HRP: Humanoid Robotics Project of MITI. In Proceedings of the First IEEE-RAS International Conference on Humanoid Robots, 2000.

[5] K.Yokoyama, H.Handa, T.Isozumi, Y.Fukase, K.Kaneko, F.Kanehiro, Y.Kawai, F.Tomita and H. Hirukawa, Cooperative Works by a Human and a Humanoid Robot. In Proceedings of IEEE International Conference on Robotics and Automation, Taipei, Taiwan, September, 2003.

[6] F.Kanehiro, K.Kaneko, K.Fujiwara, K.Harada, S.Kajita, K.Yokoi, H.Hirukawa, K.Akachi and T.Isozumi, The First Humanoid Robot that has the Same Size as a Human and that can Lie down and Get up, In Proceedings of the 2003 IEEE International Conference on Robotics and Automation, Taipei, Taiwan, September, 2003

[7] S.Kajita, A Realtime Pattern Generator for Biped Walking. In Proceedings of International Conference on Robotics and Automation, Washington DC, April, 2002.

[8] K.Nishiwaki, T.Sugihara, S.Kagami, M.Inaba and H.Inoue, Online Mixture and Connection of Basic Motions for Humanoid Walking Control by Footprint Specification. In Proceedings of IEEE International Conference on Robotics and Automation, Seoul, Korea, May, 2001.

[9] F.Yamasaki, K.Hosoda and M.Asada, An Energy Consumption Based Control for Humanoid Walking. In Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, Lausanne, Switzerland, October, 2002.

[10] K.Nagasaka, Y.Kuroki, S.Suzuki, Y.Itoh, J.Yamaguchi, Integrated Motion Control for Walking, Jumping and Running on a Small Bipedal Entertainment Robot. In Proceedings of IEEE International Conference on Robotics and Automation, New Orleans, LA, April, 2004.

[11] N.S.Pollard, J.K.Hodgins, M.J.Riley and C.G.Atkeson, Adapting Human Motion for the Control of a Humanoid Robot. In Proceedings of International Conference on Robotics and Automation, Washington DC, April, 2002.

[12] K.Yamane and Y. Nakamura, Natural Motion Animation through Constraining and Deconstraining at Will. IEE Transaction on Visualization and Computer Graphics, vol. 9, no. 3, pp352-360, 2003

[13] A.Bruderlin and L.Williams, Motion Signal Processing. In Proceedings of the 22nd annual conference on Computer graphics and interactive techniques, 1995.

[14] M.Gleicher, Motion Editing with Spacetime Constraints. In Proceedings of the 1997 Symposium on Interactive 3D Graphics, 1997.

[15] A.C.Fang and N.S.Pollard, Efficient Synthesis of Physically Valid Human Motion. In Proceedings of SIGGRAPH 2003.

[16] L.Kovar, M.Gleicher, F.Pighin, Motion Graphs. In Proceedings of SIGGRAPH 2002.

[17] F.Kanehiro, K.Fujiwara, S.Kajita, K.Yokoi, K.Kaneko, H.Hirukawa, Y.Nakamura and K.Yamane, Open Architecture Humanoid Robotics Platform. In Proceedings of International Conference on Robotics and Automation, Washington DC, April, 2002.

[18] http://www.h-anim.org/

[19] M.Vukobratovic, B.Borovac, D.Surla and D.Stokic, Biped Locomotion: Dynamics, Stability, Control and Application. volume 7 of Scientific Fundamentals of Robotics, Springer-Verlag.

[20] K.Nishiwaki, S.Kagami, Y.Kuniyoshi, M.Inaba and H.Inoue, Online Generation of Humanoid Walking Motion based on a Fast Generation Method of Motion Pattern that Follows Desired ZMP. In Proceedings of International Conference on Intelligent Robots and Systems, Lausanne, Swizerland, October, 2002.

[21] K.Yokoi, F.Kanehiro, K.Kaneko, K.Fujiwara , S.Kajita and H.Hirukawa, A Honda Humanoid Robot Controlled by AIST Software. In Proceedings of the IEEE-RAS International Conference on Humanoid Robots, Tokyo, November, 2001