GENERATING WHOLE BODY MOTIONS FOR A BIPED HUMANOID ROBOT FROM CAPTURED HUMAN DANCES
人間の舞踊を模倣するロボットの全身動作の生成

by

Shinichiro Nakaoka
中岡 慎一郎

A Master Thesis
修士論文

Submitted to
the Graduate School of Information Science and Technology
the University of Tokyo
on January 28, 2003
in Partial Fulfillment of the Requirements
for the Degree of Master of Information Science and Technology
in Computer Science

Thesis Supervisor: Katsushi Ikeuchi　池内 克史
Professor of Computer Science

ABSTRACT

Recently, the technology of biped humanoid robots has progressed to the point where it has the potential to realize complex tasks with the whole body. The goal of this study is imitating human dance motions for preservation of traditional dances. The problem with dance imitation is the difference of body structure and physical properties between humans and robots. This study describes how we solve the problem by using filters to acquire feasible robot motions from the original human motions and developing a motion generation method based on motion primitives.

Human dance motions are acquired by motion caputuring systems. Analyzing the trajectories of particular marker positions enables us to extract a high level structure of primitive motions. Initial arm motion is generated by inverse kinematics of the joint positions and is modified into a stable motion which statisfys the mechanical structure and capacity of actuators. Leg motion is generated from the sequence of primitive motions, considering constraints of the robot. Then, a waist trajectory which satisfies the desired balanced point between the body and the ground is calculated. This process enables the robot to keep its balance without falling down.

Generated motions were tested on the OpenHRP system, which consists of a dynamics simulator and actual robots. In this test, the validity of this study has been certified.

論文要旨

近年歩行ヒューマノイドロボットの技術が発達し、ロボットの全身を使った高度な動作の研究が盛んである。本研究では人間の舞踊の模倣動作を対象とし、ロボットによる実演を通して伝統舞踊の保存・伝達に役立てることを目的としている。模倣動作の生成においては人間とロボットの身体構造や物理特性の違いが問題となる。本研究では、ロボットへの適応のための動作フィルタと、基本動作要素に基づいた動作認識・生成により、この問題を解決している。人間の舞踊動作はモーションキャプチャにより各関節の座標値として取り込まれる。得られた動作データに対して、注視点の軌道を分析し舞踊を構成する基本動作要素の抽出を行う。腕の初期関節角軌道は関節座標の逆運動学から算出し、角度・角速度制限フィルタを用いてロボットの機構や駆動能力を満たした安定な関節角軌道へと修正する。脚の関節角軌道は抽出された基本動作要素列からロボットの制約を考慮しつつ生成する。最終的に足と床との理想的なつりあい位置を実現する腰軌道を算出し、動力学バランスを保った転倒しない動きへ統合する。生成されたロボット動作の挙動を動力学シミュレータと実機ロボットからなる OpenHRP システム上で検証し、本手法の有効性を確認した。

Acknowledgements

I would like to express my sincere gratitude to my current theses advisor, Prof. Ikeuchi Katsushi, for his invaluable support,

I am very grateful to Dr. Nakazawa Atsushi, for giving me much support in my research.

I am very grateful to Prof. Kimura Hiroshi and Prof. Komura Taku, for giving me much useful advice and encouragement.

I am very grateful to members of the Humanoid Robotics Group of AIST, Dr. Hirukawa Hirohisa, Dr. Yokoi Kauzhito, Dr. Kajita Shuji, Dr. Kaneko Kenji, Dr. Kanehiro Fumio, Dr. Harada Kensuke and Dr. Fujiwara Kiyoshi, for providing a excellent robot platform.

I am also very grateful to all the Ikeuchi Laboratory lab members who gave me much help in various ways.

Contents

List of Figures

Chapter 1

Introduction

The final goal of this study is to contribute to the preservation of traditional dances by developing a dancing humanoid robot that can imitate human dances. This study also includes the following topics in the field of robotics: Learning from observation, and balance control. In this chapter, we discuss the background, the purpose of this study, and related studies, with regard to those topics.

1.1 Preservation of Traditional Dances

The goal of this study is to contribute to the preservation of traditional dances as an intangible heritage. There are a number of cultural heritages in the world. that contribute to our knowledge of human culture and life in past times. However, due to their age, environment or political reasons, many of them are in danger of being destoyed. In particular, intangible heritages such as the dance may be lost because of the lack of humans who are interested in, or capable of, performing them.

Recently, there has been an attempt to preserve these heritages by the use of computer vision technology. Once entered into a computer, these heritages can be recorded forever and may possibly be recreated in the future. For example, Ikeuchi et al.[9] have developed a method of modeling cultural heritages from observation.

This thesis deals with dances as intangible heritages. There are many traditional folk dances in Japan, some of which have disappeared for the lack of successors. The goal of this study is to preserve these dances by using the tech-

nology of computer vision and robotics.

One typical approach to preserve dances is to utilize a motion capturing system to capture human motion. This captured data enables us to observe dances from various viewpoints. Realizing a realistic movie of a performance by computer graphics is possible. However, just a movie replay is not very different from video recording. Captured motion data has potential for various applications. There are more advanced attempts which enable applications beyond just a replay of a movie.

For example, Nakamura et al. [21] have proposed a method, which uses labanotation [7] to store and replay dances. Labanotation is a symbolic description method for dances which enables various applications of dance motions such as inputting, editing, searching, and analyzing based on symbols of labanotation. However, the replay in 3D animation of this system has not yet realized realistic motion. The replay motion tends to become awkward. This result is due to the direct use of labanotation. The sampling of motion in labanotation is too rough to express details of the dance. In order to solve this problem, Hattori et al. [4] have attempted to extend labanotation for computer use. They have tried to add symbols required for detailed expression, but the description becomes too complicated to be considered as symbolic description. It seems to lose the nature of labanotation.

Soga [27] has proposed a dance description and composition system which is specific to ballet performance. The system enables interactive simulation of ballet choreography with any human models. However, a minimal unit of description and composition is a large block of motion, and the motion connection between the units is not smooth. This becomes a problem because the lack of a smooth connection reduces the usefulness of the composition facility. Though local motion of replay becomes smooth, the whole motion becomes awkward.

The 'BUYO-FU' system developed by yukawa et al. [35] is one of the most practical applications. This system has the facility of description and composition for dance motion, called 'BUYO-FU'. In contrast with Soga's system, the target dances are general ones. The minimal motion unit of this system is a small block of motion, which is associated with symbols according to the kinds of choreography, and the system has smooth connecting method for the motion units. The system realizes symbolic description, practical composition, and realistic replay

simultaneously. The problem with this system is that a human has to segment motion and classify the segments. The system requires many dance performances and much preprocessing by a skilled human.

It should be noted that representation is realized only by computer animation in all the systems mentioned above. We, however, take a different approach for representation.

1.2 Need for a Dancing Robot

The facility of computer animation is insufficient for complete preservation because we cannot watch the actual performance by a real dancer. "Watching a dance movie" is not the same as "Watching an actual dance by a real dancer". In addition, it is difficult to learn and master dances only by watching such movies. A student needs an instructor who knows important parts of the dance.

Therefore actual dance performance is necessary. In this study, a humanoid robot is used to realize actual dances.

For entertainment use, some dancing robots have been developed. Pollard et al.[25] have developed a robot which imitates human dances. Their robot is fixed with a stand at the waist and the motion is limited to the arms. Compared with their study, we use a biped humanoid robot which stands on its legs, and whose dance performance includes leg actions. Sony has produced dancing robot which can perform with the whole body. However, their robot performs patterned motion created from scratch. This study describes how the robot performs dances by imitating human dances.

There are studies about dancing robots like the above, but a dance performance by the whole body, including leg actions, by a human-sized robot had never been realized. There had also been no study which attempted to do it. Note that we do not attempt to develop a new robot hardware especially for dance performances; rather, we use existing humanoid robots and have developed software for the purpose, because we want to develop a generic system which can be adapt to any kind of robot hardware.

Some may be of the opinion that robot performance cannot contribute to dance preservation because the performance cannot be a complete imitation due to the difference between a robot body and the human body. However, one of the

objections to this opinion is that the experience of a real performance must give a stronger impression than that of video movies, 3D computer graphics, or even representations by virtual reality.

Additionally, mastery of a dance does not necessarily mean following the exact same movement as the original because no high skilled dancers have the same bodily characteristics; nor do they perform exactly the same synchronized trajectory for the same dance. From this consideration, we can safely say that the nature of dance motion may be revealed through the process of robot imitation despite the differences between a robot and a human.

1.3 Learning from Observation

A robot is general-purpose machine and it should be able to master various tasks in various situations, but programming control software to realize the tasks is difficult not only for a nonspecialist but also for a robot engineer. It is desirable that a robot learn and master tasks by observing human behavior. Furthermore, this attempt may help us to understand the learning mechanism of humans. With regard to this consideration, many previous studies attempted to develop such a robot. In particular, for the purpose of this study, dance performance is also considered to be a task which should be acquired by observing a human dancer. Thus, one of the problems of this study is how to observe a human dance and how to generate an imitating dance motion from the observation.

A framework of learning from observation s proposed by Ikeuchi et al. [8]. Within that framework, a task model which represents essential elements of the task is defined by a human. Observation and reproduction of the task is done based on that task model. For example, Takamatsu et al. [29] have proposed a model of an assembly task for manipulators. In their model, the state of assembling objects is defined based on contact relation of the objects. Manipulations which changes the state are defined as primitive motions. On this model, a robot observes a human demonstration and extracts a transition of the state. Then the robot reproduces operations as a sequence of primitives which follows changes of the state. The essence of this framework is that a human defines essential elements of tasks beforehand. The reason for this approach is based on the assumption that the human also has an a priori framework of ability by nature and learning is done

on the basis of it. The problem is how to extracts elements of the framework from observation and how to produce skilled behavior within the framework.

There are more generic approaches to learning from observation. Inamura et al. [10] [11] have proposed a framework called 'mimesis loop'. Their framework deals with general motion and uses common primitives for every kind of motion. Patterns of low-level joint angles are employed as common primitives. Recognition and reproduction of motion is represented by a hidden-markov of primitive sequences. Then, recognition and reproduction are iterated several times and representation well matched to the original motion is acquired. In their method, motion can be well classified according to its characteristics. For example, walking styles such as those of old or young, male or female, tired or vital, etc. can be recognized. However, since the method is limited to an outline of brief motion, reproduction of precise operation is difficult. It cannot be applied to complicated tasks such as assembly operation or dance performance so far.

In this study, dance imitation for the robot is done on the basis of some type of task model for a dance. The main element of this model is primitive motion, which is a minimal unit of choreography. Whole dance motion can be composed of a sequence of primitive motions. The same kind of primitive motion may appear several times in a dance. These primitives are clustered into the same primitive, and have some parameters for detail characteristics.

Dance motion can be basically symbolized by primitive motions. The problem is the extraction of the primitive motions from the original motion data and dance reproduction from the primitive motions. Our model is well matched to human dance learning. A human dancer often uses a 'dance notation', which is a symbolic description like musical scores. Therefore, the extraction of the primitives is also the attempt to generate dance notation automatically. If a robot generates a dance notation by observing a dance and performs the dance according to the dance notation, we can say that the robot learns and performs dances in the same way that a human does.

For a performance, motion is reproduced basically as a sequence of primitive motions. However, it is not enough to reproduce feasible motion because other aspects besides just an expression exist. Aspects such as body balance, rhythm of motion, and constraints of the robot must be taken into account. Adaptation to these aspects is considered to be the skill part of this framework. In a task

like assembly, the skill exists in the function of each operation in the task, such as parts insertion or adhesion by welding. The skill is how to achieve an accurate and effective result of the functions. However, the meaning of the skill of a dance task is different from that of an assembly task because the main purpose of dance task is not its function, but rather its expression. In other words, the function in a dance task is different from the usual meaning. Given that the function of dance task is to follow a sequence of motion primitives under the constraints of a robot, e.g., mechanical structure, capacity of actuator, and balance control, the skill of a dance task can be considered as how to acquire better expression in the conflict between ideal expression and the constraints. In this study, the process of adaptation to the constraints considers acquiring an expression similar to the original dance.

1.4 Balance Control of a Biped Humanoid Robot

Recently, walking biped humanoid robots have become popular in the field of robotics. Several studies have been made on biped walking, and walking ability has been advanced. To achieve stable walking, both hardware ability and control software are important.

In terms of hardware, several biped humanoid robot have been developed. Honda [5] has produced P2 and P3; these robots are very likely the first robots which have the hardware ability of stable dynamic walking for a long time. Research on biped walking has been active again since Honda released P2 and their products brought public attention to humanoid robots. One of the significant features of Honda's products is the harmonic gear motor used as a joint actuator. This actuator has quite powerful torque, which increases control ability to realize stable walking. Another significant feature is shock absorbers in the ankle joints. It enables the robot to adapt to uncertain reactions when it comes in contact with the floor. However, details of the hardware and the control software have not yet been opened to public.

As a product of research institutes, the Humanoid Robot Project has produced HRP-2P [19] which is designed to enable a wider range of tasks than Honda's. For example, it has no backpack which would restrict motion because of it weight and size. Articulate coxa is designed to be more compact so that possibility of

self collision is reduced. That expands the range of leg motion. Compared with Honda's product, detail specification of HRP-2P is open.

Another product by research institutes, the humanoid robot H6, which is designed by Kagami et al.[13] has been developed. A significant feature of it is toe joints, which expand leg motion.

In terms of control methods, some walking pattern generation methods have been proposed. Takanishi [36] proposed a method which uses compensation motion of the upper body. Since a walking motion is recognized as cyclic pattern, the motion can be expressed by the Fourier series. From this expression, upper body modification which satisfies dynamic balance of walking is acquired. Kajita et al. [16] [15] [14] have proposed the method '3D-LIPM'. In their method, walking is modeled as the behavior of an inverted-pendulum. Center of mass of a robot corresponds to pendulum sphere and leg corresponds to pendulum rod. If the height of the pendulum is supposed to be constant, a stable walking pattern is generated from behavior of the pendulum. For the contribution of this study, now a simple patterned walking is stable enough. The approach of this study is that a model of dynamics is defined and motion is acquired by a calculation formula on the model. Another approach exists. Endo et al. [1] have proposed a method which uses a generic algorithm to generate walking pattern. However, these methods have not yet achieved a stable, efficient walking pattern. Currently, the method based on modeling and formulation is practical.

In addition to walking, control methods for running also have been proposed. For example, Nagasaki et al. [20] have proposed the running pattern generation method. Cyclic running trajectory of center of mass is calculated by expressions derived from dynamics analysis. This method realizes some steps of running on simulation. However, experiment on actual robots cannot carried out because of mechanical constraints and uncertain behavior in the real world.

Recent topics have shifted to more complex control such as adaptive walking on rough terrain or motion which requires whole body correlation.

Yokoi et al. [34] have proposed a control method which enables walking on rough terrein. According to input of an ankle force sensor, waist position is translated to stabilize the body. This controller is embedded in robot farmware and it functions during walking in real-time. The controller realizes stable walking and walking on slightly rough terrains.

Fujiwara et al. [3] have investigated how to minimize damage to a robot when it falls to the ground. They have proposed a method to control the attitude of a robot while it is falling down, so that the robot lands on its shock-absorbing parts. Their study is not only an interesting topic of robot control, but also useful to other studies of robot control, because it enables actual experiments of unstable motion and expands the domain of robot control.

Dance performance is a complex task; probably it is one of the most difficult tasks because it includes arm motion over a wide range, very quick motion, unstable posture, etc. Although balance control is necessary, the task must not ignore appearance in dance motion.

There are some generic control methods which consider dynamics. Tamiya et al. [30] have proposed a method to keep the robot balanced when it is supported by a single leg. Although this method is useful, it is insufficient for total balance control for dance performances. Yamane et al. [32] have proposed the concept of a 'Dynamics Filter', which modifies the given motion to satisfy dynamics law. However, the result does not necessarily mean that the robot does not fall down with modified motion. In the field of computer animation, Tak et al. [28] proposed the approach of 'Balance Filter'. They proposed a framework to acquire a balanced motion from an unbalanced motion. In the first step, the dynamic property of the original motion is calculated. Next, the property is modified to satisfy the dynamic condition of balance. Finally, the original motion is modified to satisfy the modified dynamic property. This process is reasonable. However, a concrete algorithm for modification is not stated in their study. The algorithm is required above all things.

In this study, we employ the balance control method proposed by Nishiwaki et al. [23] [22] That method defines a model which assumes that all the body segments move the same amount parallel to one another. On this model, balance is maintained by translating the upper body. This method is useful because it can generally be adapted to various cases. However, all this method deals with is how to modify base motion to acquiring the desired balance condition. How to create base motion and what is the better balance condition are not provided. This study proposes a method to solve these issues in order to develop the total technology of dance performance with balance keeping.

1.5 Organization of This Thesis

The remainder of the thesis is organized as follows.

First the overview of this study is presented in Chapter 2. Then detailed algorithms of the process are described in Chapters 3 and 4. Chapter 3 is about recognition of human motion and Chapter 4 is about generation of motion for a robot. Next the experiments of virtual and actual robots are described in Chapter 5. Finally, we summarize this study and discuss the future work in Chapter 6.

Chapter 2

System Overview

In this chapter, we outline a whole process from a performance of a human dancer to the performance of a robot. The system for this process is shown in Fig. 2.1.

First, dance motion is acquired as digital data from a human performance. This process is carried out with a motion capturing system.

Since captured motion data cannot be directly imported into a robot, the data must be converted to enable it to drive a robot. This process mainly consists of two parts: recognizing dance motion from captured motion data and generating motion data for a robot. Using the recognition process, symbolic representation of the dance is acquired. The representation is composed of 'primitive motions, which are minimal units of dance motion. The generation process uses both the captured data and the result of the recognition. Through this conversion process, several elements of performance such as composition of the dance or playing speed can be edited. The whole conversion process can be automated by 'Robot Motion Composer' software.

As the final step, a robot performs a dance according to the generated motion data. We use a HRP robot platform which has a common control interface between virtual robots for simulation and actual robots. The validity of the generated motion data is tested by simulation on a virtual robot. Then a performance of an actual robot is realized with the same motion data.

In the following sections, details of each process are described.

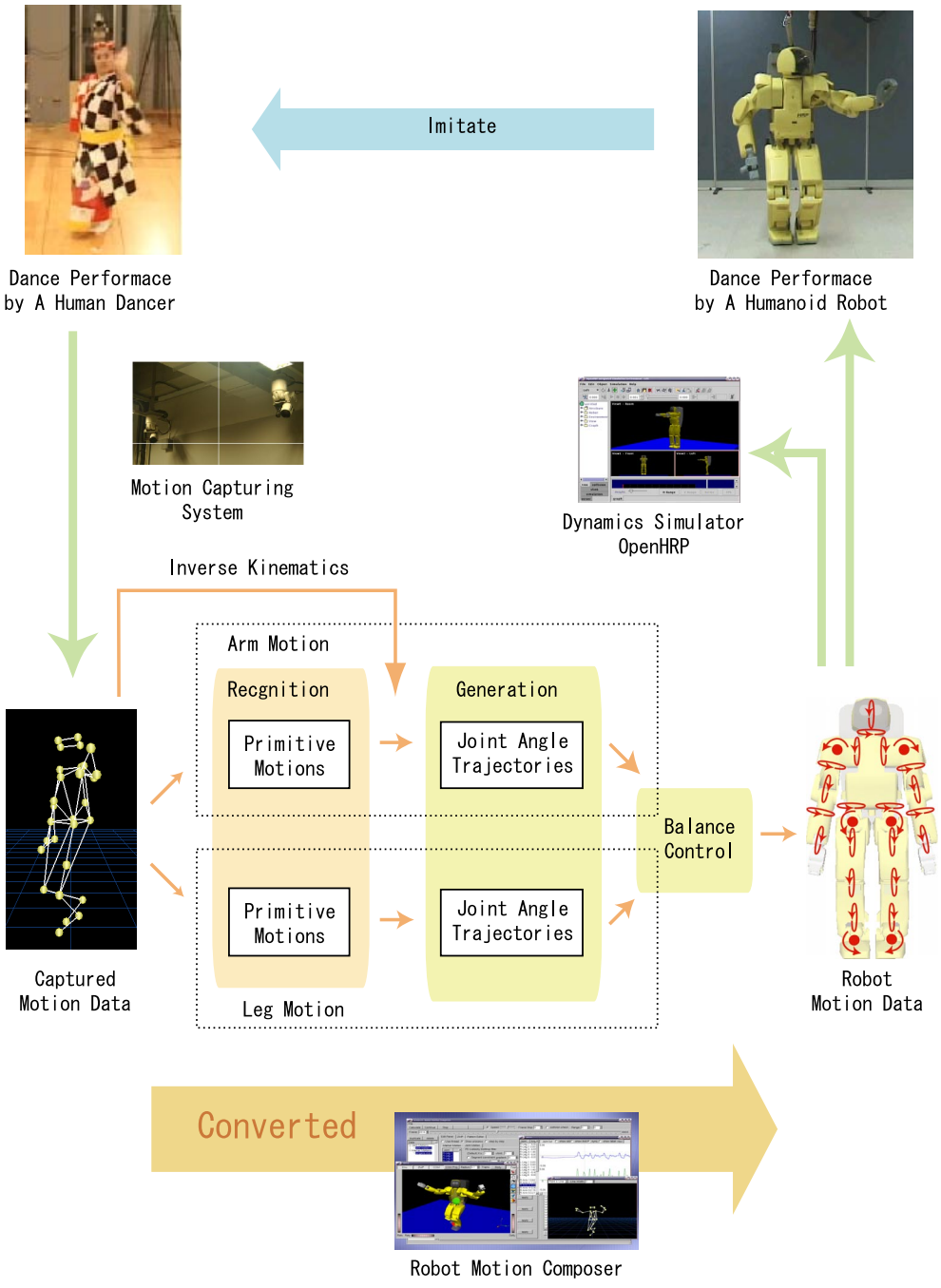

Figure 2.1: Overview of the System

Figure 2.2: multi view camera system

2.1 Capturing Human Motion

First of all, it is necessary to acquire dance motion of human as digital data. Many humanoid robots have CCD cameras mounted on their heads. Ideally, the robot can recognize dance motion from movies made by these cameras. This facility has not been achieved in previous studies, and that problem is not our present concern. Instead, motion-capturing systems are used to acquire human motion are the first step.

2.1.1 Motion Capturing System

A motion-capturing system is one which can acquire time series properties of posture such as joint position or joint angle. We use two popular types of the system: the magnetic method type and the optical method type. Both systems track positions of several markers in 3D space. Magnetic systems also track orientations of markers. Markers are attached to a performer and body motion of the performer is captured as a sequence of each marker.

The magnetic system consists of a magnetic field generator and magnetic sensors as position markers. Sensors are electronically driven by their controller. The controller is attached to a human body and acquired data are transmitted

by radio. This complicated sensor system restricts the total number of markers. However, since sensors recognize both position and orientation, the number of markers is sufficient in most cases.

Rather than the restriction of markers, the magnetic field is disturbed by metal objects in the environment, a critical occurance. In a room of a reinforced concrete building, precise data are difficult to acquire. Additionally, the available capture area is restricted within a small area because of the capacity of the magnetic field generator. Although the system has the above restrictions, it is useful because accurate data is captured in real time under proper conditions. It also has the merit of portability.

The optical system consists of several cameras and optically impressive markers and lights. Figure 2.2 shows cameras of the system. In most systems, an infrared ray is used to capture markers. In that case, cameras are infrared ones, markers are made of a material which reflects infrared ray well, and lights are projectors of infrared rays. In order to distinguish markers clearly from the background, usually the human must wear a single-colored suit with the markers on it. Each camera has a different position and orientation from the others, which enable it to shoot markers which may be hidden from some cameras, and to calculate 3D position of markers by integrating several movies obtained from different viewpoints with the principle of triangular surveying. Figure 2.3 shows images of the same scene taken by eight cameras.

Compared with the magnetic system, the optical system can acquire only the position of markers, not their orientation. However, this is not a problem because the system can have any number of markers to acquire detailed motion as far as the markers can be distinguished. Also, the optical system capturing area can be wider than that of the magnetic system, depending upon the distribution of cameras. On the other hand, the optical system requires postprocessing by a human operator for calculation of the markers, and the calculation process consumes much CPU time. Although the optical system has the disadvantage of the complex task, it does have the advantage of scalability.

2.1.2 Captured Digital Motion

Figure 2.4 is an example of a human dance performance and the digital motion data captured from it. This dance is one of our target dances, 'Jongara-Bushi',

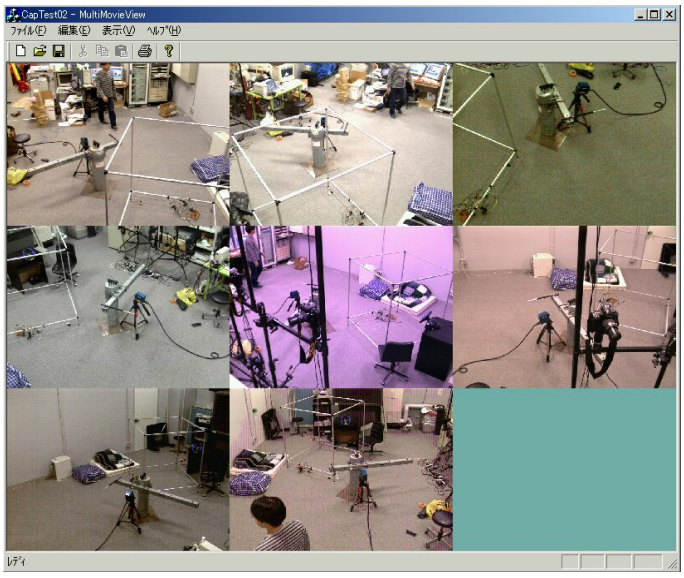

Figure 2.3: Acquire images from eight cameras.

the traditional folk dance in Tsugaru, Japan.

In this example, the dancer performs the dance while wearing markers on the joints as Fig. 2.5, and the dance is captured by an optical type system 'Vicon' which has eight cameras and 33 markers, and a capturing rate of 200 frames per second. The number of markers is sufficient to express a dance motion except for details of the finger movements. Figure 2.6 shows the layout of the markers.

The frame rate also matches the needs. This data becomes the basis of imitating motion.

The motion data are stored in the computer as discrete sampled motion, which is a sequence of poses in a sampling interval. Each pose is called a 'frame'. A frame is stored as the array of marker positions, and motion data are stored as the array of the frames in the order of time.

2.2 Conversion of Motion Data

Dance imitation is achieved through the conversion processes from captured motion data to motion data for a robot. This is the main part of this study.

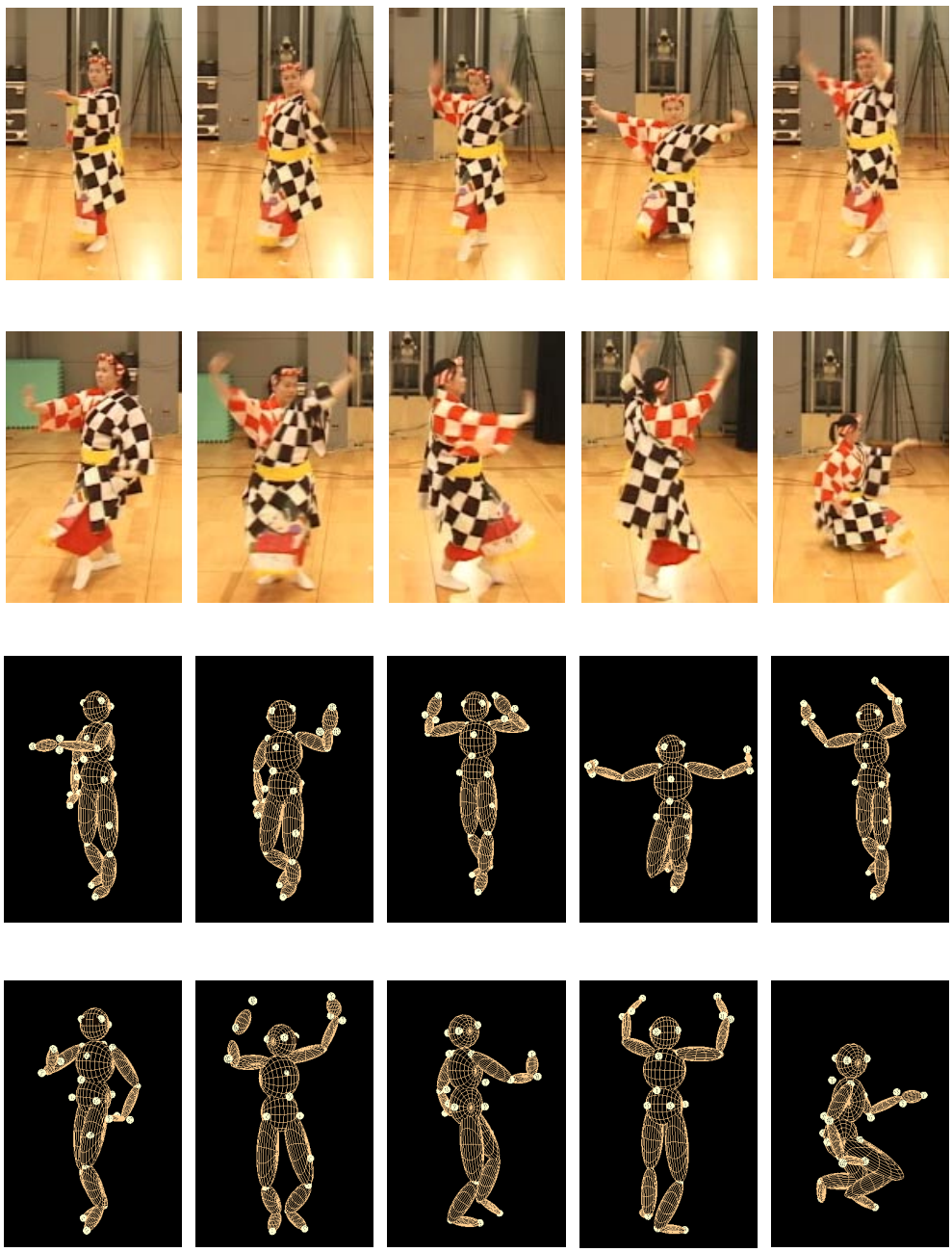

Figure 2.4: Jongara-Bushi

Figure 2.5: Capturing Scene

One of the reasons why the conversion process are required is that the original motion cannot be adapted to actual robots directly. Robots are usually controlled not by captured marker positions, but by joint properties such as the angle, angular velocity, or torque of each joint. Furthermore, there is a difference in body structure and muscular ability between present robots and the human body. This difference becomes a considerable constraint in expressing the same motion as human performance. Therefore, a conversion process from marker positions to joint property is required and it must consider the constraints of the robot. Through this process, the original motion data is adapted to the motion data required by a controller of a robot.

2.2.1 Recognition Process in Conversion

There is fundamental function more than just adaptation in the conversion process. The attempt of this study is not to develop just a dance recorder / player, but rather to develop an ability for a robot to recognize dance motion and to perform dances according to that recognition as a human does. This approach brings a higher flexibility to performance than simple replay. For example, re-edit of dance composition, adaptation to various stage condition, or interactive performance can be listed as potential of this approach. For this approach, the conversion

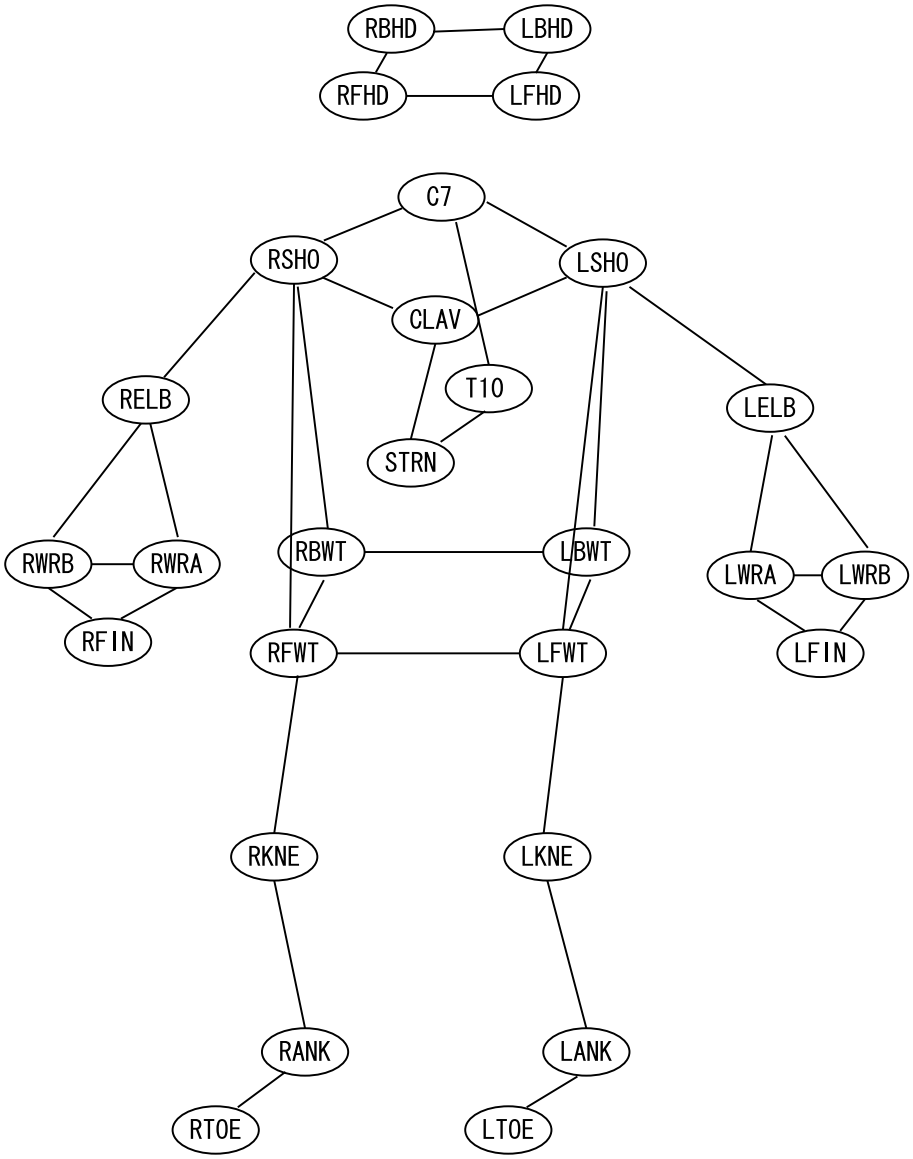

Figure 2.6: Layout of markers

processes must include the facility of recognizing dance motion. The process has a high level structure which utilizes the result of the recognition, in contrast with a simple conversion process based only on the captured motion data.

Also, this approach is useful from the practical viewpoint of robot control. If the simple conversion process is employed, there is the case that the difference in body structure mentioned above makes the process too complicated. In that case, it is more reasonable that robot motion is generated from scratch, according to the recognition result.

This case especially applies to leg motion. Present robots have severe constraints of leg structure in movable range, self collision, power of actuators, and contact flexibility between sole and ground, etc. Therefore, If a leg motion for a robot is simply converted from the original data, the motion must easily include unfeasible motion over these constraints. The modification process for these constraints becomes too complicated because of many elements of the constraints and high frequency of unfeasible part. However, in this case, a feasible motion can be more simply generated from scratch according to recognized leg actions, while, at the same time, considering the constraints.

2.2.2 Conversion System

The outline of the conversion process is depicted in Fig. 2.7. The different process is taken between leg motion and arm motion. Leg motion is generated from scratch according to recognition results. This way is derived from the above discussion. On the other hand, arm motion is generated not from scratch, but from the original data by inverse kinematics of the marker positions. This generation process considers recognition results and robot constraints. The reason for selecting this way is that direct conversion in arm motion is not as complicated as leg motion, and it would be easy to generate abundant expressions rather than generating from scratch. The recognition result is also used in the process to adapt robot constraints, in order to preserve important expressions of the dance. After initial motions of arms and legs are generated, they are integrated into the robot body. This motion still cannot be performed by the robot, because it does not satisfy dynamic balance between the robot and the floor. The motion must be modified to satisfy balance consistency.

We have developed software, the Robot Motion Composer (MOCO), which can

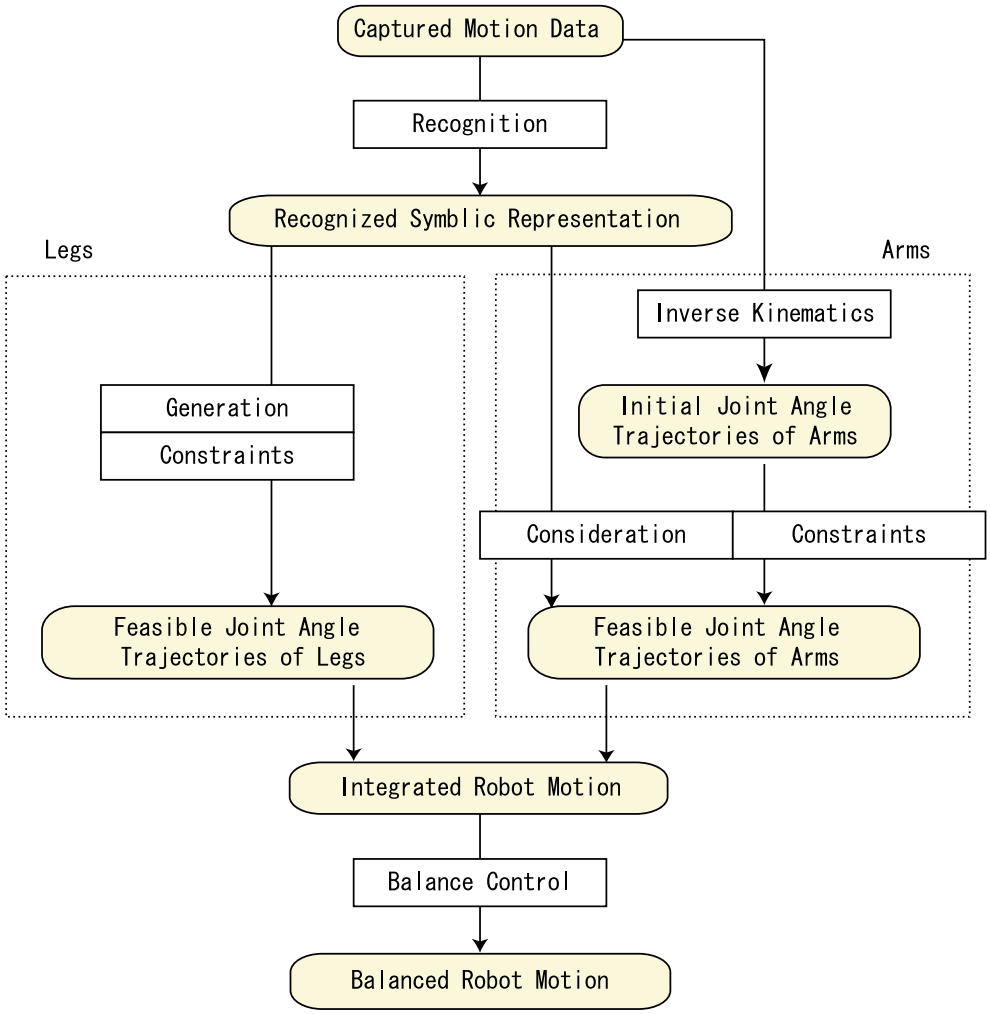

Figure 2.7: Conversion process

automate all of the conversion process described above. Once given the original captured data and a robot model to target, MOCO generates motion data that enables the robot to perform the original dance. Standard file formats of captured motion files and robot model files such as VPM or OpenHRP model file can be used. A user can see a performance of both the original motion and the generated motion by 3D animation with additional information such as joint angle, center of mass, and dynamic balanced position. A user can also specify parameters of each process interactively. This software is useful for general use or research of generating robot motion.

2.3 Platform of Humanoid Robot

In this study, we use the HRP system as a platform of the humanoid robot, which has been developed by Humanoid Robot Project (HRP) [12]. The HRP system consists of some kind of humanoid robots and dynamics simulator OpenHRP.

Three types of humanoid robots, HRP-1S, HRP-2P, and HRP-2 have been developed so far. HRP-1S is basis of P3 developed by Honda. It has a 28-DOF joint structure, 7-DOF in each arm, 6-DOF in each leg, and 2-DOF in the head, shown in Fig. 2.8. Its embedded control system was improved from the original P3 by Yokoi et al. [34]. It can be controlled on the framework of the HRP platform, and a new controller can be developed. HRP-2P is the prototype version of a new HRP robot, which was developed from scratch in open architecture [19].Though it has a structure similar to hat of HRP-1S, more slim legs and more compact articulatio coxa enable more flexible, wider leg motion. It has an additional 2-DOF in its waist which can expand upper body motion. HRP-2 is the official release version on the basis of HRP-2P. A main difference from HRP2P is just the look, which was designed by a professional robot designer. In this study, we use HRP-1S for experiments.

OpenHRP [18] is the software of a dynamics simulation system for humanoid robots. It is released as an open architecture system. Its specifications are open so that users can define their original robot model and can program a new control system. For example, since a robot model is described by extension format of VRML 2.0, a standard modeling tool can be used to develop a new model. The system consists of several distributed components such as controller, dynamics

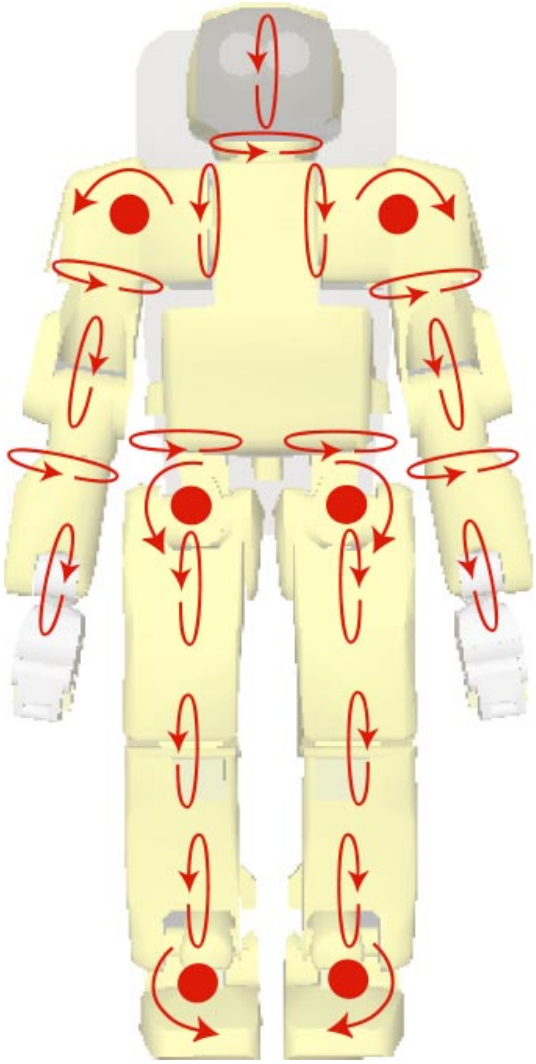

Figure 2.8: Joint Structure of HRP-1S

calculator, and viewer, on the basis of CORBA. This composition enables parallel computation and independent use of each component. It employs the dynamic computation method proposed by Yamane et al. [33] That method has a virtual link system which enables simulation including interaction between a robot and environment objects. The method also focusses on calculation time. It succeeds in realizing O(log N) complexity in simulation time. In simulations like walking, a contact model between the robot and the floor is important to acquire realistic results. A simulation of the model of perfectly elastic collision cannot match the real behavior in most cases, because of uncertain reaction in contact. For this issue, Hirukawa et al. [6] have proposed a method which uses a spring-dumper model in contact. OpenHRP employs this method and achieves realistic simulation results.

In the HRP system, the same control programs can be used between virtual robots for simulation and actual robots [17]. This makes development of the controller easy, and reduces the behavior difference between simulation and real performance.

Chapter 3

Recognition of Dance Motion

For recognition of dance motion, a framework of symbolic representation for dance is defined. Then the elements of the framework are extracted from a dance performance by analyzing its motion data. The recognition result is used in the process to generate robot motion data. Also, the extracted representation itself can be useful to enable understanding of the dance.

In this chapter, we first describes the basic idea of the recognition process. Then we describe the concrete algorithms to extract elements of dance motion. The algorithms are different for the arm motion and the leg motion.

3.1 Basic Framework of Recognition

As described in Section 1.3, a dance performance is supposed to be composed as a sequence of primitive motions, which are minimal units of choreography, and one primitive may appear several times in the dance. In this study, to recognize a dance performance is to extract primitive motions from it. However, what are the primitives ? In order to recognize a dance performance, it must be defined first.

3.1.1 Primitive Motions

In our analysis of dance motion, particular distinct poses frequently appear in a dance. A dancer pauses motion for a moment with these poses. It seems that one of the essentials of a dance is to take these poses on the rhythm. These poses are called as 'Key Pose'. Figure 3.1 shows the key poses in Jongara-Bushi. Just seeing

Figure 3.1: Key Poses in Jongara-Bushi

these poses, we can roughly imagine what kind of dance it is. From the viewpoint of key pose, the transition motion into a particular key pose can be regarded as a primitive motion. Note that the primitive motions can be not only a whole body motion, but also a motion of particular parts of the body. It is possible that arms and legs perform primitives independent of each other in their own timing. In other words, synchronized primitive motions of some parts can be integrated into one primitive.

In addition to key pose, dance motion has another essential element which appears in interactions between dancer's body and the environment. In particular, it appears between the legs and the floor, for example, motions such as stepping, squatting, spinning, and jumping. The nature of these motions is some kind of function in the interaction, rather than a particular pose in the motion. These functions are also considered as primitive motions. In most cases, the functions are under the constraints in the interaction. For example, leg actions are performed under severe constraints of dynamic law, otherwise performers cannot maintain balance. In other words, this kind of primitive motion has the aspect to solve the conflicts between the purpose of the function and constraints in the interaction. To know the accurate function of these motions is particularly important to reproduce the motions.

To sum up the above discussion, primitive motions are categorized by their nature; one is pose-oriented primitive, the motions into a key pose, the other is function-oriented primitive, the motions of which have some kind of function in interaction between a dancer and the environment. In other words, the former is classified according to its appearance, and the latter is classified according to its

function. In the latter, expressions of the same kind of primitive do not necessarily have the same look. A dance motion is composed of both kinds of primitives.

It should be noted that primitive motions are not completely distinguished by this category, because most motions cannot avoid some kind of interaction with the environment, and function-oriented primitives may also be related to some key poses. However, although most primitives have both aspects, they have one of these aspects as their main feature. If the stronger aspect can be distinguished, recognition and generation of the dance motion become more certain than those which do not consider the concept of these categories.

In this study, pose-oriented primitives are applied to upper body motions, and function-oriented primitives are applied to leg motions.

3.1.2 Extraction of Primitives

Now that primitive motions have been defined, we can discuss the next step: extraction of primitives. Concrete algorithms are described in the following sections. We describe a basic process of extraction here. The basic process consists of segmentation, classification, and structuring.

First of all, a whole motion must be segmented into units which become bases of primitive motions. The segmentation is usually carried out for each body part. First, boundaries of segments are detected. For pose oriented primitives, points of time where motion pauses are found. If the pose at the point is regarded as a key pose, the point becomes a segment boundary. For function-oriented primitives, the detection process pays attention to some kind of condition for each primitive. For example, the condition of the jump primitive is the contact state between the whole body and the floor. The conditions determine segment boundaries. Next, detected boundaries are merged in each body part. A range between boundaries usually becomes one segment.

After the segmentation, each segment is classified. For key pose- oriented primitives, the classification is carried out according to the looks of key pose and the motion of the segment. Segments which have similar looks are clustered into the same kind of primitive. On the other hand, for function-oriented primitives, the kind of primitive is usually determined at the segmentation, because the motion is segmented according to the particular condition for each kind of primitive. However, there still may be undetermined primitives at this time. For example,

overlapping primitives of the same body part may be integrated or separated. An ambiguous primitive which has some candidates of primitive type is determined according to the consequent appearance pattern around it.

The classification also requires extraction of properties which determine concrete characteristics of primitives. For a key pose-oriented primitive, the pose are important properties. For a function-oriented primitive, the parameters of the function are necessary. For example, the primitive of a jumping action requires parameters such as direction, length of jumping time, etc. These parameters must be designed to reflect the nature of the primitive so that a minimal set of parameters can reproduce the original motion well.

As a final step, the primitive sequence is structured according to its appearance pattern. For example, iteration of the same primitive can be considered as one coherent semantic unit. If the same pattern frequently appears, it can also be considered as one semantic unit. Additionally, synchronized patterns between different parts of the body can be considered as one primitive on the block. Through these processes, a structured sequence like dance notations described by a human can be acquired.

3.2 Preprocessing of Captured Motion Data

In the extraction of primitives, some captured marker positions are referrencd. However, the original data are usually not complete ones, because they include the lack of data and noises. Before the extraction, preprocessing of the original motion data is necessary to acquire a better extraction result.

The lack of positions is mainly due to occlusion of markers. There may be cases that some markers cannot be shot by any cameras. In order to restore the lacking positions, the positions of lacked frame are interpolated from the positions around them. Currently linear interpolation is employed for this process. Since the number of lacking positions are few in our environment, linear interpolation can satisfy the purpose. However, a more advanced interpolation method is desirable for general use.

The noise of the data has a significant effect on the extraction. Many processes in the extraction refer to velocity or acceleration derived from the original positions. In these derived data, the effect of noise tends to become largter. This

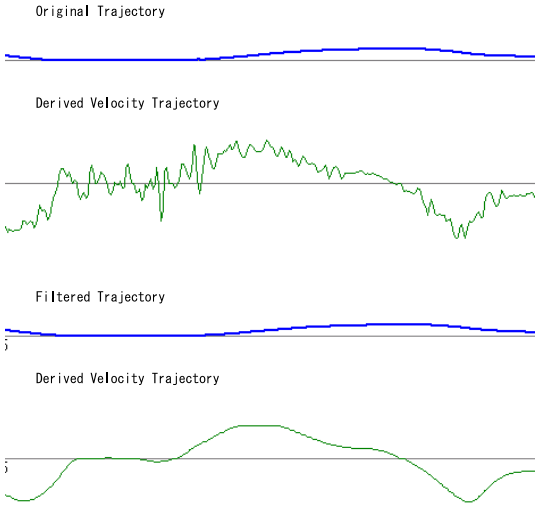

Figure 3.2: A result of filtering. The trajectory is the height of a hand marker and the velocity derived from it. The upper two graphs show the original trajectory, and lower two graphs show the filtered ones.

effect make extraction results incorrect. The noise of the data must be eliminated by some smoothing filters. We employ the gaussian smooth filter. In this method, the position of each frame is determined from the original positions around it as follows:

$$g(x) = \frac{\exp\left(\frac{-x^2}{2\sigma^2}\right)}{\sigma\sqrt{2\pi}} \tag{3.1}$$

$$g'(x) = \frac{g(x)}{\sum_{i=-R}^{R} g(i)} \tag{3.2}$$

$$p'_i = \sum_{j=-R}^{R} g'(j)p_{(i+R)} \tag{3.3}$$

where P_i is the original position of frame i, p'_i is the new position of frame i, and R is the calculation frame range from center.

Figure 3.2 shows a result of the filtering. The graph of the original trajectory is not smooth because of noise, and the velocity derived from them has much

more noise. After adapting the filter, both graphs become smooth. Since this difference is reflected to the difference of the extraction result, the preprocessing is important to acquire a correct recognition result.

3.3 Recognition of Arm Motion

For analysis of arm motion, we focus attention on the movement of a hand, the end of an arm. First, let investigate a speed graph of a hand. Figure 3.3 shows the speed graphs of both hands in Jongara-Bushi.

As the graph shows, the speed changes from approximately zero to some value in a cycle. That is, the hand motion is composed of cyclic movement in short intervals. The dancer moves a hand and stops it repeatedly to the rhythm of background music. A motion of one cycle is a relatively simple one, such as "Swing down the arm" or "Push forward arm with wrist twiddle". This tendency appears in other dances.

From this result, the motions of one cycle are considered as primitive motions. Since these motions aim at a particular pose, they are pose- oriented primitives. Arm motions are recognized on the basis of the pose-oriented primitive.

The whole recognition process is as follows:

1. find key poses of each arm according to speed of a hand
2. segment motion at the points of key pose
3. classify each segment according to a trajectory of a hand
4. structure a primitive sequence of the extraction result

The detailed algorithms of this process are described in the following sections.

3.3.1 Segmentation According to Key Pose

First the key poses must be detected. The key poses are poses at the frame where speed of a hand is approximately zero. In order to acquired the frames of a key pose, all we have to do is to detect such a frame where the value is approximately zero on the speed graph. The condition is as follows:

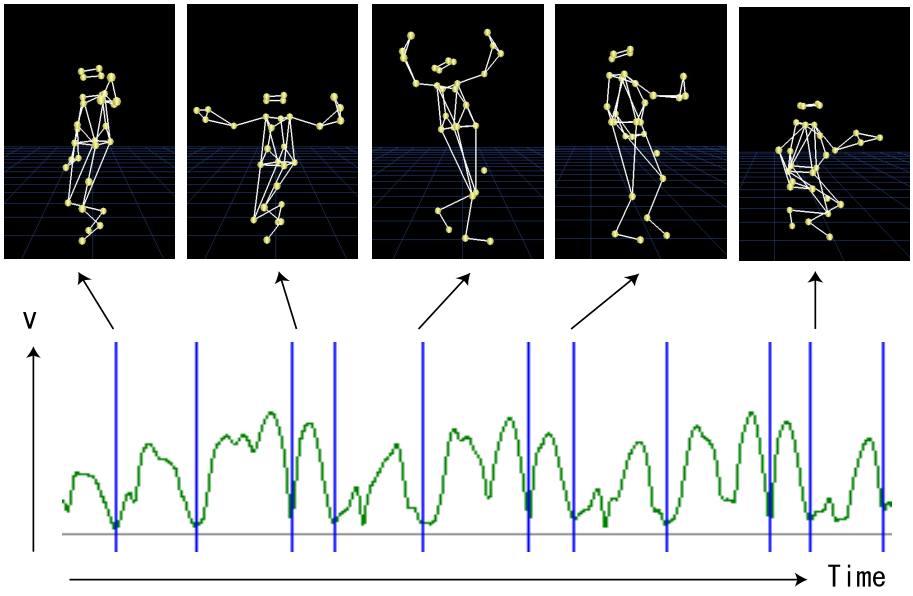

Figure 3.3: A speed graph of a hand movement and key pose examples

$$(s_i - s_{i-1}) \leq 0, (s_{i+1} - s_i) \geq 0 \tag{3.4}$$

$$s_i \leq T_s \tag{3.5}$$

where T_s is the speed threshold for judging stopping. Inequality 3.4 judges whether frame i is local minimum or not.

In Addition, the following condition must be satisfied for consecutive key pose at frame i_1 and i_2,

$$\sum_{i=i_1}^{i_2} s_i \geq L \tag{3.6}$$

where L is the threshold of the length of the trajectory which the hand passes along. This condition eliminates the motions which are too short to be considered as primitive motions.

Acquired frames are regarded as boundary of segments. Each arm motion is segmented at these boundaries and the segments become the base of primitive motions. Figure 3.3 also shows a segmentation result and some of its key poses.

3.3.2 Clustering Segments

After the segments are acquired, they must be clustered so that several segments which have similar motions are classified into the same primitive. In classification, we focus attention on the path trajectory of a hand. Each segment is classified. according to the shape of the trajectory. To evaluate the similarity between two trajectories, we employ DP matching distance. Distance between segments m and n $D(m,n)$ is calculated by the following equations.

$$D(m,n) = S(V_m, V_n) \tag{3.7}$$

$$S(k,l) = d_{k,l} + min(S_{k,l-1}, S_{k-1,l-1}, S_{k-1,l}) \tag{3.8}$$

$$d_{i,j} = |vm_i - vn_j| \tag{3.9}$$

where $V_m = \{vm_1, vm_2, ..., vm_{im} | vm_i \in R^3\}$, the sequence of positions which represents trajectory of segment m, V_n is that of segment n.

For all the combinations of the segments, the distance is calculated. Then the combinations which have small distance values less than the threshold are clustered into the same primitive by using the neighbor algorithm.

Figure 3.4 shows the result of clustering in Jongara-Bushi. Each segment is labeled by a number which represents a type of primitive motion.

3.3.3 Structuring

There is a case that the same patterns of primitives appears several times. The patterns indicate the high level structure of the dance sequence. The frequent patterns are considered as a combination unit of primitives. For example, in Fig. 3.4, the pattern 5-6-7 appears three times, so that the sequence of these three primitives can be considered as one high level unit.

To extract these frequent patterns, the apriori algorithm [2] is used.

Additionally, there are primitives synchronized between the right hand and the left hand, and some of the combinations of these synchronized primitives fre-

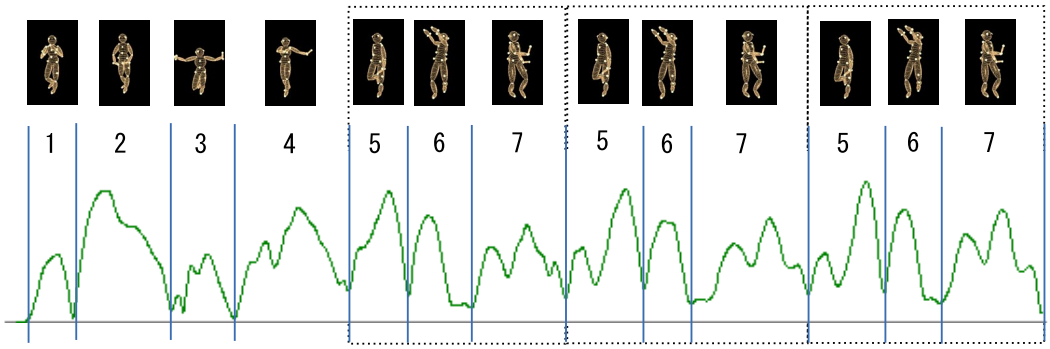

Figure 3.4: Classified primitive sequence in the right arm

quently appear. They are considered as primitive motions which consist of both arms. It is also high level structure of the sequence. To find these frequent synchronized combinations, the probability of synchronized combination of primitive A and B is calculated by the following expression:

$$co - occurrence(A, B) = \frac{f(A \cap B)^2}{f(A)f(B)} \tag{3.10}$$

where A is a primitive in right hand, B is a primitive in the left hand, $f(x)$ is the frequency of primitive x.

If this value is higher than the threshold, the combination is integrated into one unit of primitive.

A structured sequence makes it easier to survey the whole dance sequence.

3.4 Recognition of Leg Motion

Let us now investigate the speed graph of a foot, the end of a leg. Figure 3.5 is the speed graph in Jongara-Bushi. From this, it seems that cyclic movements similar to arm motion appear and that pose oriented primitives may be adapted. However, there is a difference that resting frames of which the speed is almost zero exist. During these frames, the foot is still on the floor. Is there no primitive motion during that time?

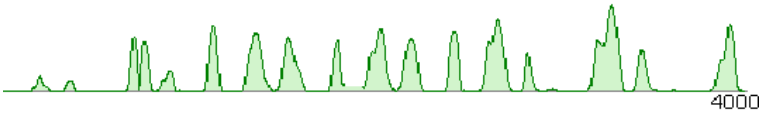

Figure 3.5: Speed of a foot

The fact is that the leg does not just keep the same shape even in that time. This means that the leg which is supporting the body on the floor is almost always moving to maintain balance. These motions cannot be ignored. They are necessary primitive motions, which have the function of supporting the body with balance keeping. Such a function cannot be recognized by the way of pose-oriented primitive.

Most leg motions are such kinds of motions, because legs are directly interacting with the floor. Therefore, a function-oriented primitive is reasonable for primitive motions of legs.

In function-oriented primitives, each primitive must be precedently defined by a human: What to do, how to recognize, parameters to express the motion, etc. In the following sections, definition of primitives for present target dances are described. Then the extraction methods of each primitive are described.

3.4.1 Primitives of Leg Motion

In our target dances, Jongara-Bushi and Aizu-Bandaisan, three basic actions are observed as primitive motions: standing, stepping, and squatting. We labeled the three primitives as STAND, STEP, and SQUAT. Figure 3.6 shows these actions. By the way, generally there may be other basic actions such as jumping, spinning, kicking, etc. However, since these actions do not appear in the present target dances, currently we do not focus attention on these actions.

STAND represents the motion in which both legs support the body and maintain balance. One of its parameters is the length of standing time. Another parameter is the projection position of the center of mass. This parameter is required because that position determines the shape of the leg positions. For example, in the pose where both feet are opened on both sides, the pose changes whether the projection of the center of mass is on each foot or center of both feet.

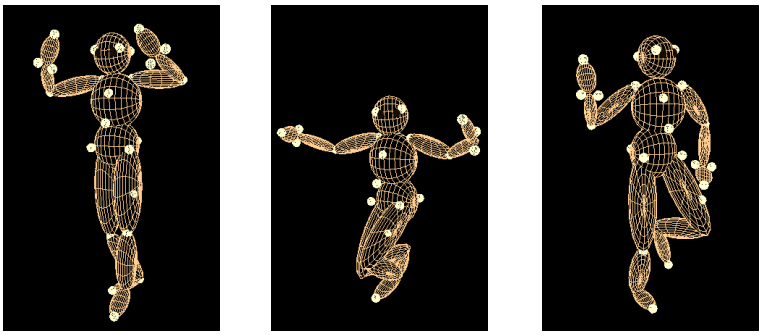

Figure 3.6: Motion primitives of the legs

In the former case, the pose becomes asymmetrical, one is shortened and the other is stretched. In the latter case, the pose becomes symmetric. To put it the other way round, some kind of motions can be expressed by changing the projection point.

STEP represents one stepping motion. To be precise, the motion that one foot is lifted up and down on the floor while the other foot is supporting the body. The former foot is the *swing foot* and the latter foot is the *support foot*. Running steps do not correspond to the STEP primitive. This primitive has various possibilities for performance. To express various performances, the primitive has medium time and final time, and has states at these times. The medium time is the time when the swing foot is at the highest position in the whole motion. The final time is the time when the swing foot lands on the floor again, at the end of the motion. Both time values are measured from the beginning of the motion. As the state of swing foot, the position and orientation of it at those times are required. In addition to them, a state of the waist orientation at the final time is required. Those properties are described as relative values from the support foot. The primitive does not require the initial state so it can be adapted to any initial poses. Also, projection position of the center of mass is not required because the contact point between foot and the floor is always under the support foot.

SQUAT represents one squatting motion, bending the knees down and then straightening up again. As well as STEP, this primitive has medium time and final time, but state is required only at medium time. The medium time of this

primitive is the time when the waist takes the lowest position. The state of medium time is the height of the waist.

The definitions of three primitives are summarized in table 3.1.

Table 3.1: Parameters of Leg Primitives

Primitive	Parameters
STAND	- standing time - projection of center of mass
SQUAT	- medium time, final time - the lowest waist height at the medium time
STEP	- medium time, final time - position and orientation of the swing foot at the medium time and the final time - waist orientation at the final time * The positions and the orientations are described on the support foot coordinate

Note that values of the primitive parameter which is concerned with the length must be normalized into some standard human model. The normalization enables unified description and adaptation to various robots.

3.4.2 Extracting Leg Primitives

To extract STEP primitives, the speed graph of a foot (Fig. 3.5) is analyzed. This graph is basically a sequence of bell-shaped curves. During one unit of the curves, the foot moves and stops. This movement is regarded as a stepping motion. Hence, the extraction process has to find one unit of the curve which satisfies the following conditions:

$$s_i \geq T_s (a \leq i \leq b) \tag{3.11}$$

$$\sum_{i=a}^{b} s_i \geq T_d \tag{3.12}$$

Figure 3.7: Speed of the waist

where s_i is the speed at frame i, a is initial frame, b is final frame ($a < b$), T_s is speed threshold, T_d is threshold of length which the foot passes along. Inequality 3.12 eliminates actions too small to be regarded as stepping actions.

From the above conditions, segments of the STEP primitive are extracted. Each segment is represented with initial frame, final frame, and medium frame at which the swing foot takes the highest position in the motion.

After each STEP segment is extracted, its parameters are examined. The position and the orientation of a foot is calculated from marker positions: the toe marker and the ankle marker. These properties for both the swing foot and the support foot are extracted at the medium frame and the final frame. Then the parameters are acquired as relative values of the swing foot from the support foot.

To extract SQUAT primitives, a velocity graph of the waist height (Fig. 3.7) is analyzed. In this graph, the squatting action appears as a set of a concave curve and a convex curve, that is, the movement to lower the waist and bring it up again. The extraction process has to find this set of curves. The following conditions must be satisfied:

$$s_i \leq 0 (a \leq i \leq b), s_i \geq 0 (b \leq i \leq c) \tag{3.13}$$

$$\sum_{i=a}^{b} s_i \leq T_d, \sum_{i=b}^{c} s_i \geq T_d \tag{3.14}$$

where s_i is the speed at frame i, a is initial frame, b is medium frame, c is final frame ($a < b < c$), T_s is speed threshold, T_d is threshold of length which the waist pass along. Inequality 3.12 eliminates actions nothing but a small swinging.

The medium frame and the final frame from the initial frame are stored as primitive parameters. Other parameters are the deepest waist height at the

medium frame and projection of center of mass at that time. Center of mass is approximately considered as the waist position.

The segment of STAND primitive corresponds to the frames where the speed remains approximately zero in both the speed graph of a foot (Fig. 3.5) and the speed graph of the waist height(Fig.3.7) at the same time. For the medium frame of each segment, projection of center of mass is examined, and this value is stored as the parameter.

Finally, values of parameters which are concerned with length are normalized. Currently, the model of HRP-1S is employed as the standard model. The values are scaled to fit to this model.

Figure 3.8 shows a extracted primitive sequence in Jongara-Bushi.

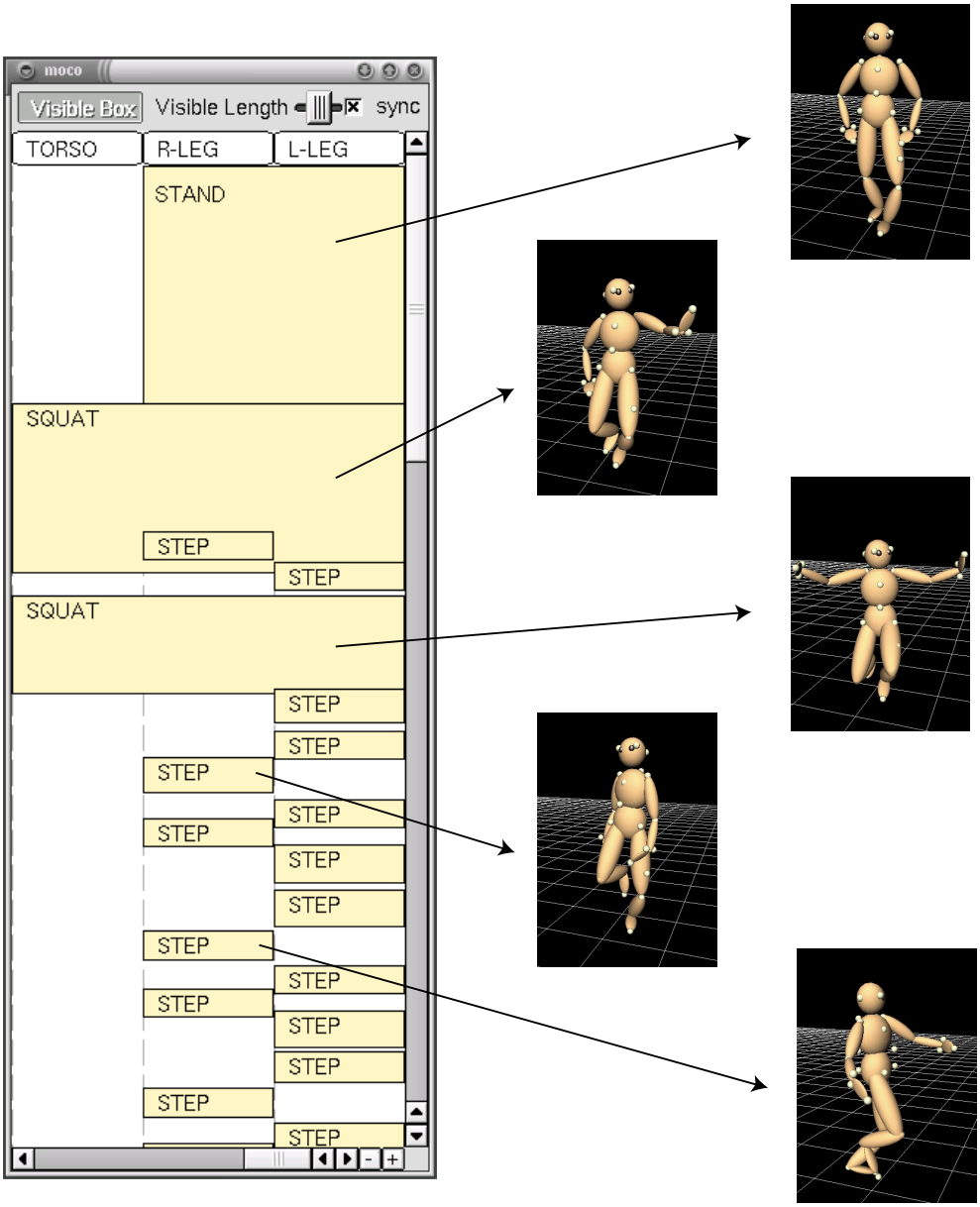

Figure 3.8: Extracted primitive sequence and poses in some primitives

Chapter 4

Generation of Robot Motion

Motion data for a robot is basically expressed as joint angle trajectories. The data is generated according to the recognition result. First, initial motion data for arms and legs are independently generated. This process considers constraints of the robot. Then arm motion and leg motion are integrated, and modified to satisfy dynamic balance.

In this chapter, we describe the constraints of a robot and the total method to generate robot motion under the constraints, using the recognition result.

4.1 Constraints of Robot

A humanoid robot is usually designed to have a body as similar to the human body as possible, and some kinds of human-like motion such as simple walking can be performed by it. However, there are still many differences between a robot body and the human body. The differences become constraints when motion data captured from humans are imported into the robot. The constraints make complete imitation impossible. The attempt to develop new robot hardware which eliminates the constraints is an important approach. However, we currently focus on realizing a seemingly good motion by using existing robots. In the generation of robot motion, knowledge of the details of the constraints is necessary, and a robot must manage to perform better expression under the constraints.

In this section, we describe the constraints of a robot from the viewpoint of mechanical structure and capacity of actuators. A target robot for discussion is

HRP-1S. Since HRP-1S is one of the typical humanoid robots which have been developed until now, the discussion is generally valid with regard to most present robots.

4.1.1 Constraints in Body Structure

It is a fact that a robot has quite a different joint structure from that of a human body.

One of the differences is degree of freedom (DOF), which represents the number of movable joint axes. HRP-1S has 28 DOF: 7-DOF in a arm $\times$ 2, 6-DOF in a leg $\times$ 2, and 2-DOF in a neck (Fig. 2.8) This seems sufficient for human-like movement. On the other hand, the human body is composed of quite complicated articulos and muscles, so that its DOF is over 100. This indicates that DOF of the robot is too restricted. This restriction becomes a considerable constraint for the dance motion, because many dances include actions which require many DOF such as torso twists. HRP-1S has no joint in the torso, so it cannot express torso twists. In the same way, adequate DOF is not available for some parts of dance motions.

In addition, the lack of toe joints also restricts the expression and movement ability of legs, because a human almost always uses the toe in leg action in order to express some motion, to control balance, or to move efficiently. Range of joint angle is also restricted. The robot cannot raise its arms or open its legs as well as a human does. Geometric shape of body parts also restricts motion. For example, a backpack of HRP-1S becomes an obstruction against some kinds of motions, such as swinging an arm behind the back. Or HRP-1S has fat legs, so leg motion easily causes self collision.

Another significant problem is that the robot structure has singular points. For example, the shoulder joint is constructed of a three rotation sequence of pitch, yaw and roll. When the robot raises its arm horizontally, the DOF of the shoulder joint decreases and the robot cannot freely change the direction of the arm from that pose Figure 4.1 shows this situation. On the other hand, a human can move the shoulder joint in all directions regardless of its pose.

One approach to avoid the problems of singular points is to increase the number of joints. However, it is not easy to attach the new joint mechanism and actuator because of the lack of space within the body.

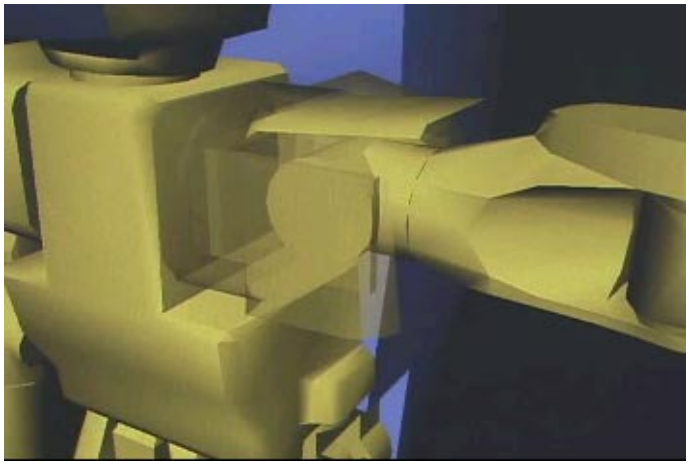

Figure 4.1: Pose in singular point. The robot cannot move the arm back and forth.

There is another approach which employs a new mechanism. Okada et al. proposed 'Cybernetic Shoulder' which has no singular point on simple mechanism [24]. This mechanism is innovative. However, although a robot with the cybernetic shoulder has been developed, it has only a body from the chest up which is fixed on a stand. Currently, there is no robot which has the whole body with the cybernetic shoulder. Such a robot may be difficult to develop, because this mechanism requires much space in the chest.

4.1.2 Constraints in Capacity of Actuators

Capacity of actuators becomes a constraint. Actuators of HRP-1S are the most powerful ones currently available. They are harmonic-gear motors, which realize powerful torque within a compact space, because of big reduction ratio and compact mechanism.

After Honda employed this type of motors in their product, they were widely used as actuators of biped humanoid robots.

However, although the actuator has sufficient torque, its angular velocity is not so very high because of the high reduction ratio. There is a case that the actuator cannot follow the original speed of motion. Dance motion often requires

quicker motions than walking or everyday tasks. Particularly, one of our targets, 'Jongara-Bushi', includes motions that are too quick to follow. A faster actuator is required to perform dances which include quick motion in the same speed.

4.2 Generation of Arm Motion

In generation of arm motions, initial values of joint angle trajectories are calculated directly by inverse kinematics of captured markers. Then the initial values are modified to adapt constraints of a robot by joint angle filters. This process manages to keep key poses and their timing as much as possible.

4.2.1 Inverse Kinematics from Captured Makers

To determine a pose of a robot, joint angles are required. Joint angles of arms are calculated by inverse kinematics of captured marker positions. In general, 'inverse kinematics' means to determine a sequence of several joint angles from a position of one target point, usually the end of the object. On the other hand, the inverse kinematics here is slightly different from the generic one, because we have abundant information about joint positions, which is acquired as marker positions. Joint angles can almost directly be calculated from position correlations among connected markers.

Concrete process in the marker layout of Vicon (Fig. 2.6) to HRP-1S joints (Fig. 2.8) is as follows. The calculation is independently carried out for each frame.

First, from the markers RSHO, LSHO, STRN, T10, chest frame is acquired:

$$a_x = \frac{v_{LSHO} - v_{RSHO}}{|v_{LSHO} - v_{RSHO}|} \tag{4.1}$$

$$a_z = \frac{a_x \times (v_{T10} - v_{STRN})}{|a_x \times (v_{T10} - v_{STRN}|} \tag{4.2}$$

$$a_y = \frac{a_z \times a_x}{|a_z \times a_x|} \tag{4.3}$$

$$F_0 = (a_x, a_y, a_z) \tag{4.4}$$

where v_x is position vector of marker x, F_i is coordinate frame matrix of joint i from the arm origin 0. Note that 'frame' here means the coordinate system which

represents orientation of each body part. It is different from the 'frame' which represents discrete time index of a motion sequence.

The following calculation process is an iteration in which angles related to the current frame are calculated and the frame is rotated by the acquired angles. In other words, joint angles are calculated by forward kinematics of frames.

$$
\begin{aligned}
v_0 &= F_0^{-1} \cdot (v_{RELB} - v_{RSHO}) && (4.5)\\
a_0 &= arctan2(-v_{0z}, -v_{0y}) && (4.6)\\
F_1 &= R(F_{0x}, a_0) \cdot F_0 && (4.7)\\
a_1 &= \arcsin(\frac{v_x}{|v|}) && (4.8)\\
F_2 &= R(F_{1z}, a_1) \cdot F_1 && (4.9)\\
v_2 &= F_2^{-1} \cdot \left\{ \frac{(v_{RWRA} + v_{RWRB})}{2} - v_{RELB} \right\} && (4.10)\\
a_2 &= arctan2(v_{2x}, v_{2z}) && (4.11)\\
F_3 &= R(F_{2y}, -a_2) \cdot F_2 && (4.12)\\
a_3 &= -\arccos(-\frac{v_{2y}}{|v_2|}) && (4.13)\\
F_4 &= R(F_{3x}, a_3) \cdot F_3 && (4.14)\\
v_4 &= v_{RWRA} - v_{RWRB} && (4.15)\\
a_4 &= arctan2(v_4 \cdot F_{4z}, v_4 \cdot F_{4x}) && (4.16)\\
F_5 &= R(F_{4y}, -a_4) \cdot F_4 && (4.17)\\
v_5 &= v_{RFIN} - \frac{v_{RWRA} + v_{RWRB}}{2} && (4.18)\\
a_5 &= arctan2(-v_5 \cdot F_{5z}, -v_5 \cdot F_{5y}) && (4.19)
\end{aligned}
$$

where a_i is angle of joint i, which is from the shoulder origin. $R(v, r)$ represents rotation matrix which rotates r around the axis v. Though the above process assumes the right arm, the same applies to the left arm.

This process is applied to all the 'index frames' in order to acquire motion data for robot joints.

4.2.2 Filters of Joint Angle Trajectory

In a joint angle sequence acquired by the above process, angles or angular velocities may become an impossible value over the constraints of the robot. In addition to the problem of each joint trajectory, the motion may imply poses which are in the neighborhood of singular points. At poses near the singular point, valid moving patterns are limited and movement may be locked. Such motion data cannot be performed by the actual robot. This problem can be considered as a problem of angular velocity because a non-continuous curve of velocity around a locked frame can be considered to be a high gradient curve on a discrete system.

Therefore, first of all, values must be restricted within the possible range by applying some filters.

We employ a filter proposed by Pollard et al. [25]. Their filter can limit angular velocity within a specified range, and can also control angular acceleration. The basic idea of this method is that a new angle trajectory is created by the following the original trajectory under the limit of angular velocity. The process is similar to the PD control method.

First, a new trajectory is created by the following equations:

$$\dot{\theta}_i = \theta_i - \theta_{i-1}, \tag{4.20}$$

$$\ddot{\theta}'_{i+1} = 2\sqrt{K_s}(\dot{\theta}_i - \dot{\theta}'_i) + K_s(\theta_i - \theta'_i) \tag{4.21}$$

$$\dot{\theta}'_{i+1} = max(\dot{\theta}_L, min(\dot{\theta}_U, \dot{\theta}'_i + \ddot{\theta}'_{i+1})) \tag{4.22}$$

$$\theta'_{i+1} = \theta'_i + \dot{\theta}'_{i+1} \tag{4.23}$$

where θ_i is the original joint angle, θ'_i is new joint angle, i is a joint number, $\dot{\theta}_L$ and $\dot{\theta}_U$ are the lower and upper velocity limits. K_s is the parameter which controls stiffness of motion.

The result becomes a trajectory similar to the original one within the limit. However, it must delay from the original one all through the motion, because the trajectory cannot follow the original motion in areas over the limit. (Fig.4.2-b).

Then another trajectory is created by the inverse process of the above equations from end to start as follows:

$$\dot{\theta}_i = \theta_i - \theta_{i-1} \tag{4.24}$$

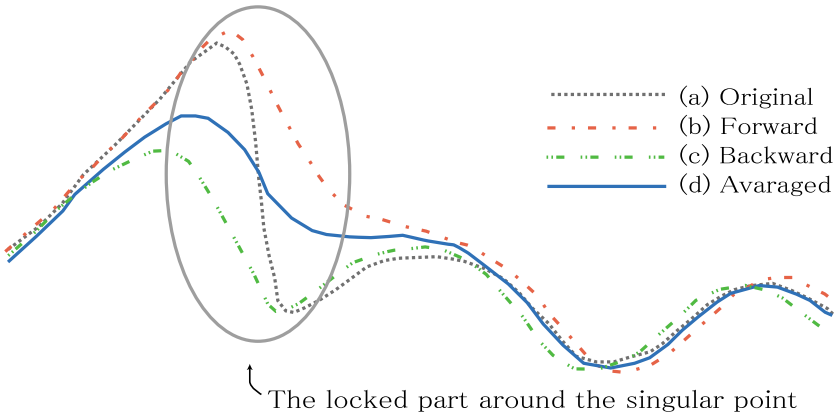

Figure 4.2: Filtering joint angle trajectory

$$\ddot{\theta}_{B,i-1} = 2\sqrt{K_s}(\dot{\theta}_i - \dot{\theta}_{B,i} + K_s(\theta_i - \theta_{B,i}) \quad (4.25)$$

$$\dot{\theta}_{B,i-1} = max(\dot{\theta}_L, min(\dot{\theta}_U, \dot{\theta}_{B,i} + \ddot{\theta}_{B,i-1})) \quad (4.26)$$

$$\theta_{B,i-1} = \theta_{B,i} + \dot{\theta}_{B,i-1} \quad (4.27)$$

The result becomes a trajectory of the future instance compared with the original one (Fig.4.2-c).

Finally, both trajectories are averaged to obtain a trajectory whose shape is overlapped by the original one as follows: (Fig.4.2-d).

$$\theta_{V,i} = \frac{\theta_{F,i} + \theta_{B,i}}{2} \quad (4.28)$$

In this final trajectory, the rhythm of curve approximately matches the original one , and values are within the limit, at the same time. The locked parts around singular points in the initial motion are also changed into a smooth curve. Figure 4.2 shows the initial motion which must lock on thactual robot, and the motion after the filter which can be performed on the actual robot.

4.2.3 Expression of Key Poses

As described in section 3.3, an arm takes a key pose at the frame where movement of a hand stops for a moment. Since key poses are essential elements of dance, they

should be expressed as obviously as possible. Particularly, stopping expression and its timing are important because they constitute the rhythm of the dance. If the timing is out of rhythm, the performance becomes a series of awkward motions.

However, in a motion after applying the filter, expression of key poses tends to be obscure. In most cases, the stopping of a hand is achieved by synchronized stops of all the joints of the arm. Although the filter preserves the approximate rhythm of each joint trajectory, stop timing may slightly differ from the original trajectory. Furthermore, the differences of the joints are not synchronized because the filter is independently applied to each joint. This inconsistency of stop timing leads to deformation or obscurity of key pose expressions, and the whole performance becomes an awkward one.

For a better performance, to clarification of key pose expressions is important. Although to make shapes of key pose exactly the same as the original shape under the constraints is difficult, synchronizing stop timings of all the joints with the original timing is possible. We take this approach to achieve a better performance.

Figure 4.2 shows joint angle trajectories of the original motion and the filtered one. Joint movement stops at extreme points, where angular velocity is zero, and some extreme points correspond to a frame of key pose. To clarify stop expressions, these extreme points of all the joint have to be arranged at the key pose frames in the original motion. Note that the arrangement must carried out within the possible range of the constraints. (Fig.4.3).

This process is as follows.

1. For each key pose frame in the original motion, find the nearest extreme point in the modified joint graph.

2. For points which are out of the key pose frames,

   - Slide the extreme point horizontally to the key pose frame.
   - If the new gradient is beyond the limit of angular velocity, slide the extreme point vertically to be under the limit.
   - stretch the trajectory to pass the new extreme point.

This process restores the rhythm of the original motion. A dance performance becomes more brilliant.

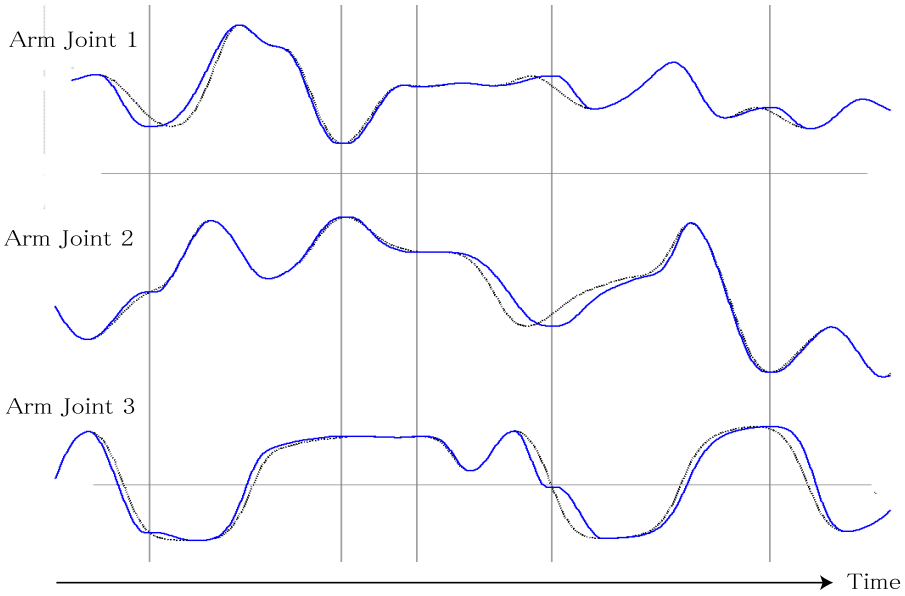

Figure 4.3: Arrangement of the extreme points nearby the segment boundaries (vertical lines). Dotted lines are the original trajectories and solid lines are the arranged ones.

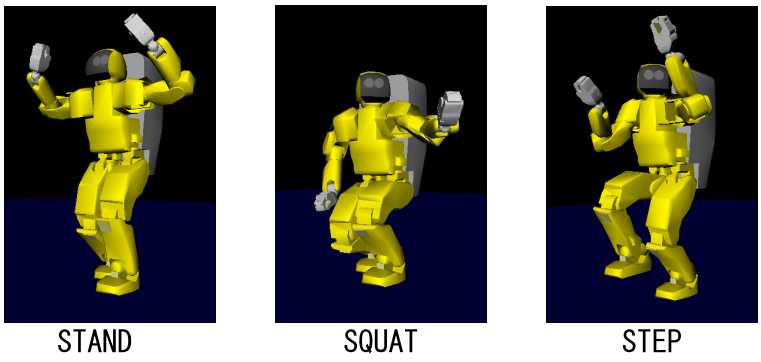

Figure 4.4: Poses generated from leg primitives

4.3 Generation of Leg Motion

Leg motion for a robot is generated by using a leg primitive sequence which is acquired from the original dance motion through the process for recognizing leg motion.

The primitive motions for legs (described in section 3.4) are designed to contain information which is required to recreate leg motion for a robot. Therefore, in contrast with arm motion, leg motion is generated only from the information of recognized primitives, in terms of the information of the original dance. Raw marker positions are not used. Figure 4.4 shows an example of generated poses of each kind of leg primitives.

In generating leg motion, joint angle trajectories are not directly generated. First, position and orientation of a foot are generated. Then, joint angles are calculated from the foot values by inverse kinematics of the leg.

4.3.1 Generation of Foot Trajectory

For each primitive in the acquired sequence, a foot trajectory which represents an action of the primitive is generated. To be precise, the values of the foot trajectory are the position of the ankle joint and the orientation of the sole. These values are expressed on the waist coordinate, because the inverse kinematics of leg require such values.

For the STEP primitive, the trajectories are basically created by interpolating

initial, medium, and final states of position and orientation. The initial state is the state just before entering the new primitive motion. The medium state and the final state are determined by the parameters of the STEP primitive. The parameters of the primitive are values of a swing foot and the waist, and they are relative values from a support foot. Since values required by the inverse kinematics are relative ones from the waist, coordinates must be converted.

A trajectory is calculated as follows.

For a support leg, the initial position vector and orientation matrix on the waist coordinate are known. They are expressed as ${}^{w}v_{su,initial}$, ${}^{w}M_{su,initial}$, where ${}^{B}v_{A,i}$ is a vector of a foot A based on B at frame i, ${}^{B}M_{A,i}$ is a rotation matrix which represents orientation of a foot A based on B at frame i. The final position is calculated to be the center of the feet as follows:

$$ {}^{w}v_{su,final} = -{}^{su}M^{-1}_{w,final}\frac{{}^{su}v_{sw,final}}{2} \tag{4.29} $$

From these values, a trajectory of a support leg is calculated by interpolation as follows:

$$ {}^{su}v_{su,i} = IV^{final}_{i=initial}({}^{w}v_{su,initial}, {}^{w}v_{su,final}, i) \tag{4.30} $$
$$ {}^{w}M_{su,i} = IR^{final}_{i=initial}({}^{w}M_{su,initial}, {}^{su}M^{-1}_{w,final}, i) \tag{4.31} $$

where $IV^{b}_{i=a}(A, B, i)$ is a vector at frame i in the interpolation from value A at frame a to value B at frame b, IR is the similar interpolation function of rotation matrix by euler angle. We use third polynominal interpolation where velocity is zero at end points.

For a swinging leg, required values are calculated in similar way:

$$ {}^{w}v_{sw,mid} = {}^{w}v_{su,mid} + {}^{w}M_{su,mid}{}^{su}v_{sw,mid} \tag{4.32} $$
$$ {}^{w}v_{sw,final} = -{}^{w}v_{su,final} \tag{4.33} $$

Then, a trajectory is calculated as follows. In a swinging leg, interpolation uses not only the final state but also the medium state:

$$
{}^{w}v_{su,i} = \begin{cases} IV^{mid}_{i=initial}({}^{w}v_{sw,initial}, {}^{w}v_{sw,mid}) \\ IV^{final}_{i=mid}({}^{w}v_{sw,mid}, {}^{w}v_{sw,final}) \end{cases} \tag{4.34}
$$

$$
{}^{w}M_{sw,i} = \begin{cases} IR^{mid}_{i=initial}({}^{w}M_{sw,initial}, {}^{w}M_{su,mid}{}^{su}M_{sw,mid}) \\ IR^{final}_{i=mid}({}^{w}M_{sw,mid}, {}^{w}M_{su,final}{}^{su}M_{sw,final}) \end{cases} \tag{4.35}
$$

There is the case that medium state and final state are modified according to the robot constraints. The modification is caused by contact condition, the constraint of movable range, and self collision.

A sole of a robot must be lie flat against the floor when a foot comes in contact with the floor. Otherwise, a foot cannot support the body stabley. On the other hand, a human can freely contact a foot with the floor. Therefore, final foot orientation in acquired STEP primitives must be constrained to be horizontal, and final position must be level with the floor.

Movable range of robot legs is usually narrower than that of human legs. If the state of a primitive is beyond that range, it must be restricted within the range.

Since the robot has fat legs, self collision of legs easily occurs. Or, modification for movable range may cause a collision. The motion which includes self collision cannot be performed. Therefore, whether collision occurs must be checked in the generation process. If collision occurs, the position of the collision and related links is examined, and the trajectory is modified to cause no collision.

For the SQUAT primitive and STAND primitive, foot trajectories are generated by a similar process, state interpolation which considers the constraints.

4.3.2 Inverse Kinematics of Leg

Since leg motion is initially generated as a trajectory of one point, the end of a leg, inverse kinematics is required in order to calculate joint angles of legs. In contrast with arm motion, inverse kinematics here corresponds to the usual one, that is, to determine a sequence of several joint angles from position and orientation of one target point.

In general, for the objects which consist of many joints, inverse kinematics is difficult to solve directly. In that case, inverse kinematics is solved by difference calculation with the Jacobian matrix. If the current pose is given, differences of joint angles from that pose are calculated from difference of the target point. Joint

angle trajectories are acquired by moving the target point in small steps from the initial pose. This method is useful because it can be mechanically applied to generic cases, and many robot systems use this method. However, this method cannot deal with the poses around singular points. For example, when a robot assume the pose in which the knee joint is fully expanded to linear, the knee joint falls into singular point.

In dance performance, using the full capacity of the body is frequently required. It is desirable that inverse kinematics can deal with poses around singular points.

Fortunately, the leg structure of HRP-1S is not too complicated to solve inverse kinematics directly. Joint angles are calculated as follows:

$$
\begin{aligned}
a_0 &= yaw && (4.36)\\
o &= \begin{pmatrix} -\sin(yaw) \\ \cos(yaw) \\ 0 \end{pmatrix} && (4.37)\\
k &= R_z(-a_0) \cdot R\left(o, \frac{\pi}{2} - \arcsin(\frac{|v|}{L})\right) \cdot \frac{L}{2|v|} v && (4.38)\\
a_1 &= arctan2(k_y, -k_z) && (4.39)\\
a_2 &= \arcsin\left(\frac{-2k_x}{L}\right) && (4.40)\\
a_3 &= \pi - 2\arcsin\left(\frac{|v|}{L}\right) && (4.41)\\
a_4 &= pitch - (a_2 + a_3) && (4.42)\\
a_5 &= roll - a_1 && (4.43)
\end{aligned}
$$

where a_i is angle of a leg joint i, a number is from the waist. v is a position of a foot, $pitch, roll, yaw$ is orientation of a foot, L is length of a leg.

4.4 Balance Control

In the motion generated through the above generation process, the robot has the ability to assume poses in the motion sequence. However, if the robot tries to perform the motion on the floor being supported by its legs, the robot will be unable to keep its balance and it will fall down.

This section describes how to control the robot motion to maintain balance so that a robot can actually perform the dance. Such controls for dynamic balance should be adapted both in off-line motion generation and online, real-time control. This thesis focuses on the former issue. The latter is solved by the control system embedded in HRP-1S.

4.4.1 Dynamic Force Balance

Because the above generation process does not consider dynamics, the generated motion does not always satisfy dynamic consistency in interaction with the floor.

Strictly speaking, in appearance, the motion is generated under the assumption that all areas of a foot sole comes in contact with the floor when supporting the body. However, in dynamics, the motion does not necessarily satisfy that assumption. Hence actual behavior comes from the assumption that the sole of a supporting foot slants against the floor. At that time, the desired motion is not achieved. If the robot still tries to follow the subsequent motion even in that time, the robot will fall down.

In order to increase stability, one valid approach is to reduce amount of the moments generated by movement of the upper body.

The Pollard method which is mentioned in Section 4.2.2 is also useful for this purpose because it can control angular acceleration by parameter Ks in equation 4.21 and 4.25. The higher Ks is, the bigger angular acceleration becomes. If higher Ks is applied to arm motion, response to the original motion becomes better, but the moment of the upper body becomes bigger. On the contrary, when Ks is smaller, response is worse but moment of the upper body is smaller so that dynamics stability is better.

When arms move in wide arcs with high acceleration, big moment is generated so that the robot becomes unstable. In this case, more stable motion can be acquired by reducing Ks.

This method is useful to increase stability. However, this cannot become a fundamental solution for balance control because stability conflicts with response of motion, and after all, this method does not guarantee dynamic consistency at all.

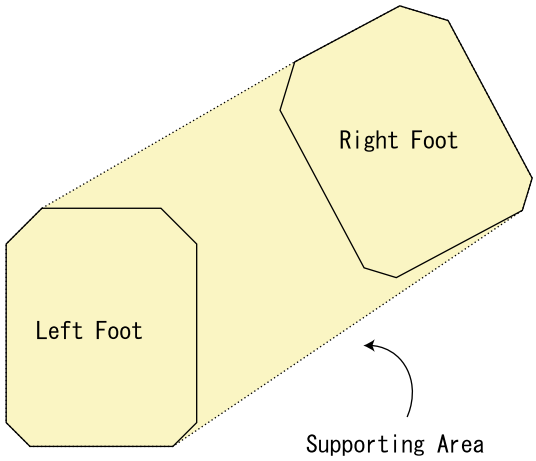

Figure 4.5: Supporting area when both feet support the body

4.4.2 Zero Moment Point

Recall that the motion is generated under the assumption that all areas of a foot sole come in contact with the floor when it is supporting the body. In other words, a sole does not rotate during that time. In terms of dynamics, this assumption is satisfied when the point at which the moment to the robot body is zero exists in the area of the sole surface. During this time, the sole does not rotate. The point is called the 'zero moment point (ZMP)' and the area is called the 'supporting area'. If a robot is supported by both feet, the supporting area corresponds to the convex area which consists of both soles as shown in Fig. 4.5. The concept of ZMP was proposed by Vukobratovic [31].

Given the physical model of a robot, a trajectory of ZMP can be calculated from motion data for the robot, under the assumption that the supporting area is infinite. If ZMP moves out of an actual supporting area, the motion is impossible to perform because the actual motion must imply rotation of the supporting sole at that time. In this way, dynamic consistency of the motion can be checked from the calculated ZMP trajectory.

A simple model of interaction between a robot and the floor is represented in Fig. 4.6.In this model, the truck on a table corresponds to the center of mass of a robot, and a stand of the table corresponds to a sole of the robot. Moment at

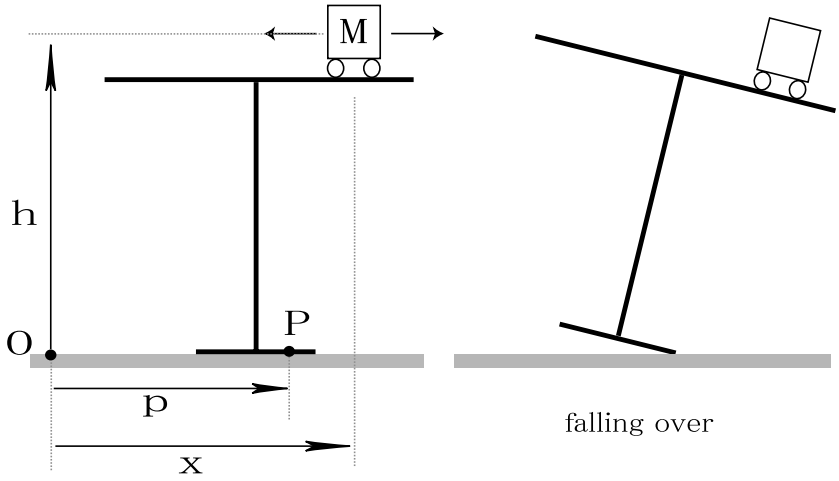

Figure 4.6: Simple Model for ZMP calculation

a point on the stand is calculated as follows:

$$\tau_P = Mg(x-p) - M\ddot{x}h \tag{4.44}$$

where τ_P is moment at a point P, M is mass of a robot, x is horizontal position of center of mass, p is position of the point P, h is height of center of mass,.

When the table maintains balance, the stand does not rotate. That is, the point at which τ_P is zero exists on the stand. In this case, the following equations are derived:

$$Mg(x-p) - M\ddot{x}h = 0 \tag{4.45}$$

$$p = x - \frac{h}{g}\ddot{x} \tag{4.46}$$

ZMP corresponds the position p .

In practice, the strict model is not as simple as Fig.4.6. The robot body consists of many joints and segments. Since the segments are a rigid body, their moment element cannot be ignored. Being considered these elements, ZMP is calculated by summing up force and moment which each segment acts:

$$x_{zmp} = \frac{\sum m_i z_i \ddot{x}_i - \sum\{m_i(\ddot{z}_i + g)x_i + (0,1,0)^T \mathbf{I}_i \dot{\omega}_i\}}{-\sum m_i(\ddot{z}_i + g)} \tag{4.47}$$

where x_{zmp} is x-axis position of ZMP, x_i, z_i are a position of a segment i, m_i is mass, I_i is inertia tensor ω_i is an angular velocity vector and g is gravitational constant. This expression is on x-axis. A siimilar expression is valid on y-axis, and the calculation can be separately performed on each axis.

Dynamic consistency of the motion can be checked from calculated ZMP trajectory. If calculated ZMP of the motion is always in the supporting area, the motion is a feasible one. To put it another way, the motion has to be generated or modified to have such a ZMP trajectory.

In this study, a desired ZMP trajectory which is always in the supporting area is prepared first. Then the motion is modified to realize the trajectory. This approach was proposed by Tak et al. [28].

4.4.3 Desired ZMP

It is fundamental that desired ZMP must be inside a supporting area. If the supporting area remains on one state, ZMP should remain a stable point in the center of the area or just below the ankle joint. However, supporting state changes with steps. A stability of motion depends on a ZMP trajectory with state transitions. For stable transition, the ZMP trajectory should be as smooth as possible. In this study, we applied the following criteria.

- In STEP state, ZMP must be located at the center of a supporting sole.
- In STAND state, ZMP moves from a previous position to the next supporting position by third order polynomial equation. Initial velocity and accelerations and final ones are kept at zero.
- If the period of STAND state is long, transition is separated into three steps: (1) from a previous position to center of supporting area, (2) stay there and (3) move to the next supporting position.
- If the period of STAND state is short, ZMP movement speeds up and robot motion becomes unstable. Adequate transition time is required for stable

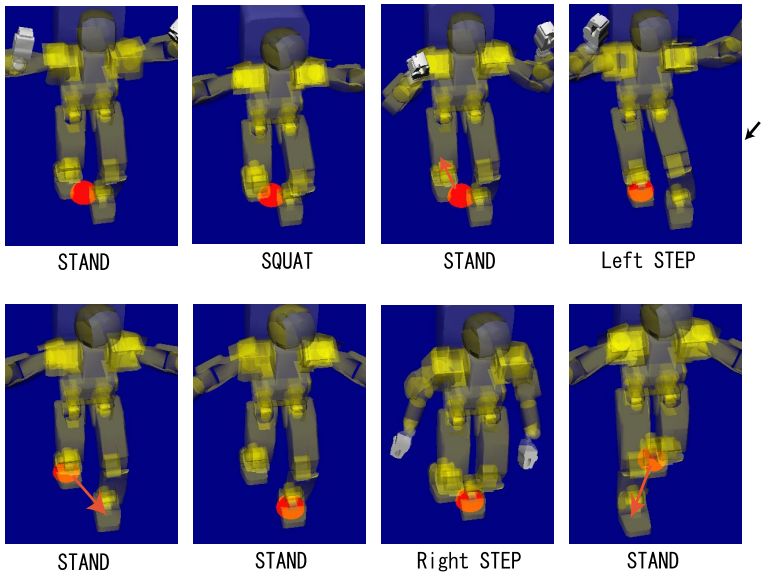

Figure 4.7: A motion with support state transition. A marker on the feet shows ZMP.

motion. In this case, ZMP movement is expanded so that it starts in the previous state and extends into the next state. Acceleration and deceleration of ZMP is done in those states.

From the above method, we generate a desired ZMP trajectory and modify the trajectory of the upper body position. Figure 4.7 shows a sequence of support state and ZMP trajectory.

4.4.4 How to realize desired ZMP

Given a desired ZMP trajectory, a motion must be modified to realize it. The problem is to solve the equation 4.47 under the situation that positions and angular velocities of each segment are unknown variables and the position of ZMP is given. However, this problem is impossible to solve directly, because the number of variables is too great and they are constrained by joint structure. In addition, better modification of which the difference from the original motion is smaller must be found among infinite solutions.

Nishiwaki et al. [23] [22] proposed a method to solve this problem simply.

In equation 4.47, in order to realize desired ZMP x'_p, consider that only x_i is modified to x'_i:

$$x'_p = \frac{\sum m_i z_i \ddot{x'}_i - \sum \{m_i(\ddot{z}_i + g)x'_i + (0,1,0)^T \mathbf{I}_i \dot{\omega}_i\}}{-\sum m_i(\ddot{z}_i + g)} \tag{4.48}$$

Supposing $x^e_p = x'_p - x_p$ and $x^e_i = x'_i - x_i$, the following equation is derived from 4.47 and 4.48.

$$x^e_p = \frac{\sum m_i z_i \ddot{x^e_i} - \sum m_i(\ddot{z}_i + g)x^e_i}{-\sum m_i(\ddot{z}_i + g)} \tag{4.49}$$

In this equation, the constraint that all the segments translate parallel in the same distance is assumed. That is, for all the segments i, $x^e_i = x^e$:

$$-\frac{\sum m_i z_i}{\sum m_i(\ddot{z}_i + g)} \ddot{x^e} + x^e = x^e_p \tag{4.50}$$

This represents the same situation as equation 4.46. In practice, modification can be approximately considered as upper body translation.

On a discrete system in Δt, the following equation is expressed:

$$\ddot{x^e}(t_i) = \frac{x^e(t_{i+1}) - 2x^e(t_i) + x^e(t_{i-1}))}{\Delta t^2} \tag{4.51}$$

From equation 4.50 and 4.51, the following equation is derived:

$$x^e_{zmp}(t_i) = \frac{-hx^e(t_{i+1}) + (2h + g\Delta t^2)x^e(t_i) - hx^e(t_{i-1})}{g\Delta t^2} \tag{4.52}$$

where x^e_{zmp} is the difference between the original ZMP and the desired ZMP, x^e is a difference between positions of the original segments and modified positions, t_i is time at frame i, h is height of center of mass, Δt is time per one frame.

This equation is expressed by information of 3 consecutive frames. These types of equations are solved as tridiagonal simultaneous linear equations [26]. $x^e(t_i)$ can be calculated mechanically.

This method cannot figure out a result which completely follows the desired ZMP trajectory in one calculation, because the constraint that all the segments translate parallel in the same distance is actually impossible. However, a result easily converges in interaction of calculation, because the difference between actual behavior and the constraints becomes smaller and smaller in interaction. Although a modification is limited to horizontal translation of the upper body, this method is effective in controlling balance.

Chapter 5

Experiments

We generated motions for humanoid robot HRP-1S by our method which we have described in this thesis, and tested the motions on the robot platform. In this chapter, we examine the validity of the generated robot motions.

5.1 Appearance of Generated Robot Performance

Figure 5.1 shows a captured dance motion and a robot motion for HRP-1S in Jongara-Bushi. The robot motion is automatically generated from the captured motion by our method. For comparison, two motions are simultaneously performed on the same stage.

It seems that the result can be regarded as a good imitation of the human performance. Although the filter changes appearance of arm motion, difference of hand trajectories do not stand out so much. The most remarkable difference is trajectories of a whole body. Since movable range of the robot legs is restricted, the robot cannot always follow a rapid turn or a wide step in the original motion. This cannot be avoided.

A method to evaluate similarity of performances is necessary. It is difficult problem. Simple comparison of trajectories of some body parts such as hands makes no sense because body type is different between a robot and a human, and this also applies among skilled human dancers. We currently have no proper method, so we cannot evaluate appearance of the performance.

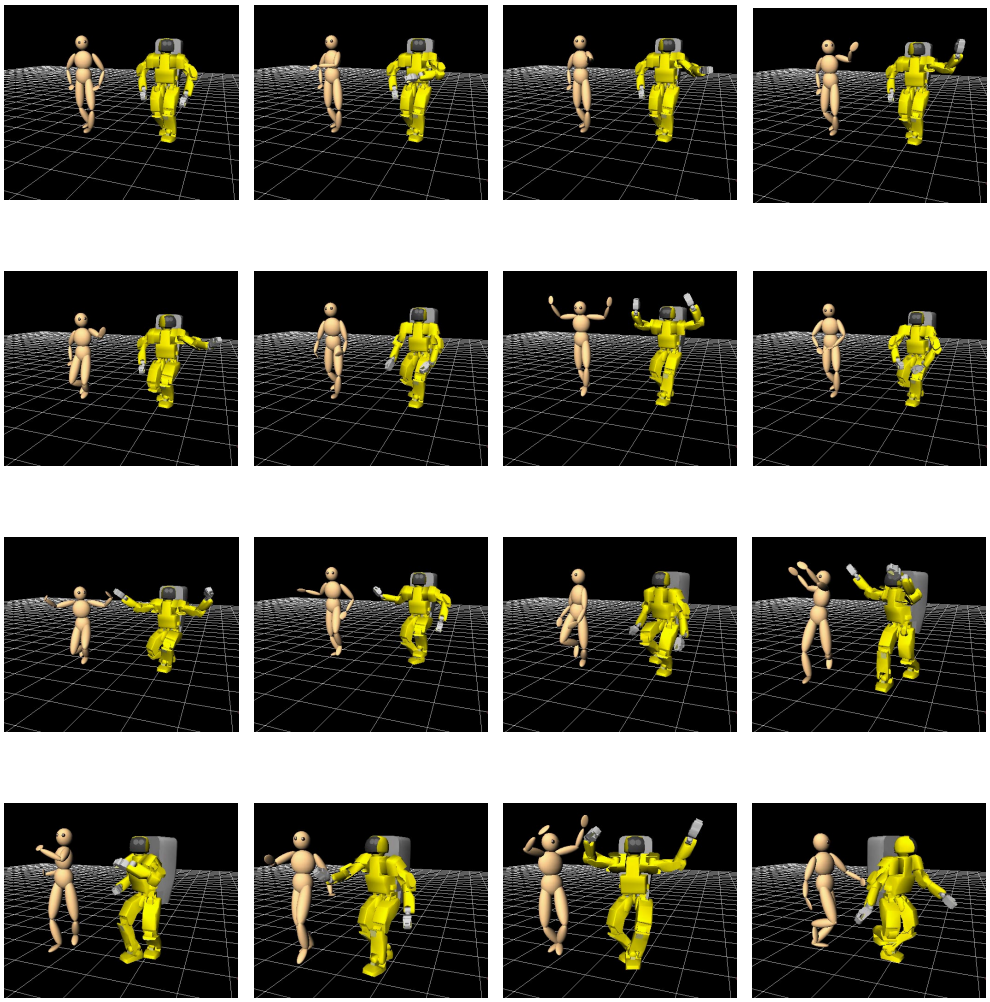

Figure 5.1: Comparison between original motion and generated robot motion

5.2 Feasibility in Dynamics

Compared with appearance, validity of dynamics is clearly evaluated by whether a robot succeeds in a performance. First, a motion is tested on the dynamics simulator. If the robot can stably perform a dance from beginning to end in simulation, an actual performance by a robot is performed as an experiment.

5.2.1 ZMP in Generated Motions

Before simulation, ZMP behavior should be checked. Robot Motion Composer has a facility to show ZMP with animation of a desired performance. In a motion without balance control, ZMP movement becomes noisy and it cannot remain in the supporting area. When balance control is applied to the motion, ZMP movement becomes quite stable. However, ZMP cannot completely follow the desired one and it cannot always be in the supporting area. The precise reason for this is not clear, but we suppose that ZMP in an initial motion before balance control is too noisy to fix completely. The balance control method seems to require smoothness of an initial ZMP trajectory. Initial leg motion is mechanically generated by third polynomial interpolation. This process does not consider dynamics at all. It seems to affect balance control.

5.2.2 Simulation of Upper Body Motion

As a first step, we examine a performance without leg steps. Three motions through different generation processes are tested. First is the motion without balance control. The econd is also without balance control, but is applied small k_s in equation 4.21 and 4.25 against unstable part of the first motion. As described in Section 4.4.1, k_s can control stability of upper body motion. The third motion is applied to all the generation process, including balance control. The tests are done in OpenHRP in HG control mode, without feedback control. The setting of that condition attempts to certify validity of only the generation process.

In the first one, the robot fell over at the first swinging action of its arms. The second one could sustain the performance until the largest swing of arms at the latter part. Stiffness parameter k_s has some function for balance control, but small k_s makes a motion lazy. It should be used with light k_s value just to help the main balance control. In the third one, the robot could stably perform a dance

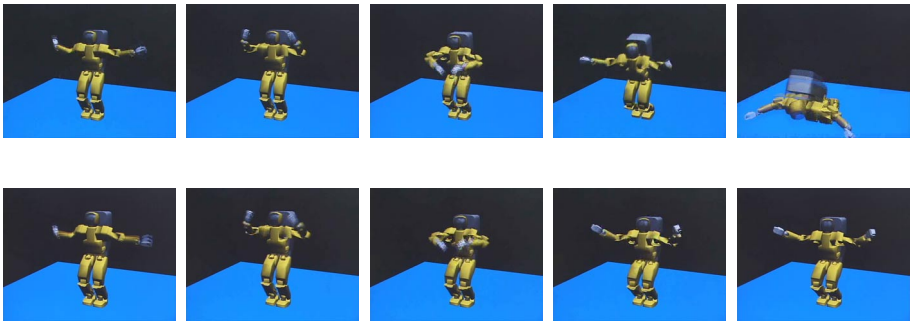

Figure 5.2: Simulation results of upper body performances. Upper sequence shows a motion without balance control. Lower sequence shows a motion with balance control.

from beginning to end. Figure 5.2 shows the results of the first one and the third one.

5.2.3 Performance by Real HRP-1S

Figure 5.3 shows an actual performance by real HRP-1S. The robot could perform upper body motions, including squats, from beginning to end.

In the first experiments, motion became unstable due to impacts generated from actuators. This behavior was caused by a limit over angular velocity. When the actuator is driven by commands over the limit of angular velocity, the actuator still just follows the commands. However, when the commands turn around, the forward direction of the actuator immediately changes , and the rapid turn generates impacts. By restricting proper limitation for actual capacity, the impacts disappear and the performance becomes stable.

5.2.4 Performance with Leg Steps

As next step, we tried a performance that included leg steps. Performance tends to be unstable when the robot uses maximum ability for expression, such as a wide step. Restricting such wide steps, allows a robot to perform the dance without falling over on simulation. Figure 5.4 shows the simulation result.

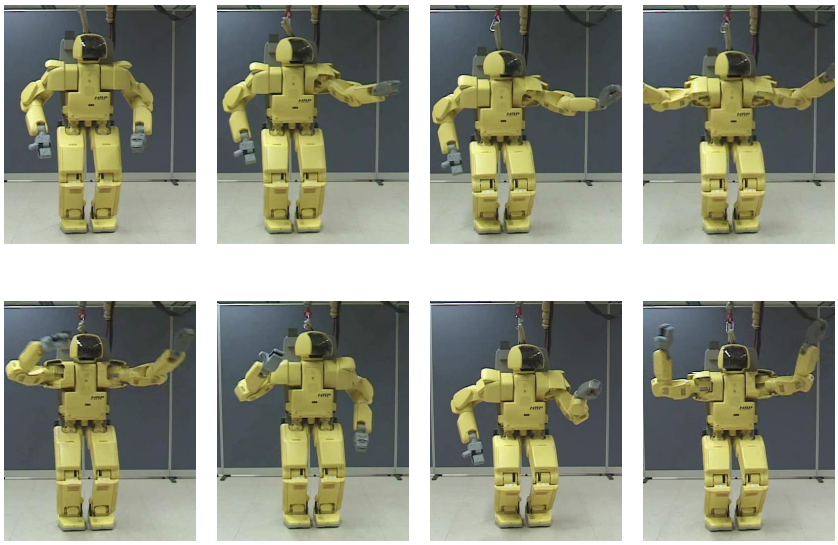

Figure 5.3: Performance of Jongara-bushi by HRP-1S

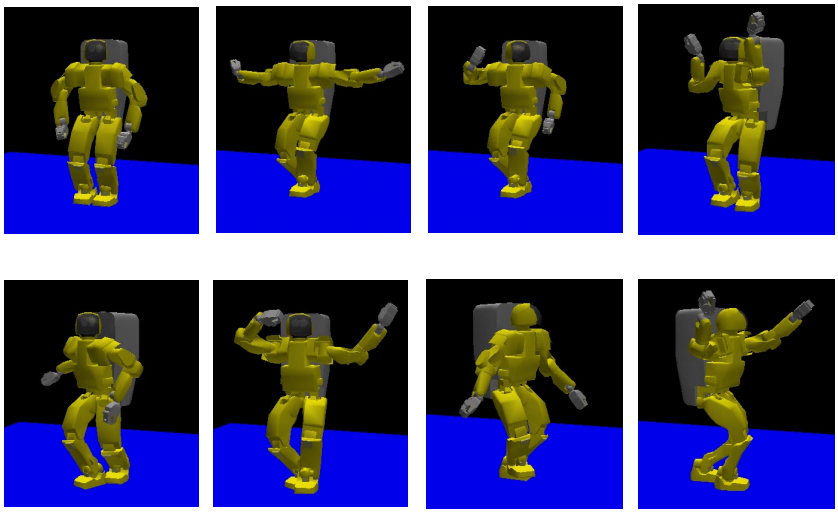

Figure 5.4: Simulation with leg steps

Chapter 6

Conclusion

In this thesis, we described a method to realize a dancing robot which has the ability to observe and imitate human dance performance.

Human dance motions are acquired by the motion capturing system. A sequence of primitive motions is extracted from acquired motion data. On the basis of the extracted primitives, a motion for the robot is generated. Imitation from observation is achieved on this framework.

Initial arm motion is generated by inverse kinematics of the joint positions and it is modified into a feasible motion which satisfies the mechanical structure and capacity of actuators. Leg motion is generated from the sequence of primitive motions, considering constraints of the robot. Then waist trajectory which satisfies the desired balance point between body and ground is calculated. This process enables the robot to keep its balance without falling down.

Through the process proposed in this thesis, captured human motion is automatically converted to a feasible robot motion.

Generated motions were tested on the HRP robot platform. As initial experiments, the upper body motion of Jongara-Bushi was successfully tested on OpenHRP dynamics simulator. Then the robot HRP-1S could actually perform the dance, maintaing balance by its legs. As next step, we succeeded a simulation of a motion which includes leg steps. Through these tests, the validity of this study has been certified.

6.0.5 Future Work

First of all, experiments of a whole body motion on the actual robot should be realized. Although performances of such motions have been realized on simulation, an actual performance seems to require more stable motions. The generation method of the initial leg motion should be improved to consider dynamics. The present balance control method itself has enough ability; additional light condition of dynamics in the generation process may help to acquire more stable motions.

There is still a significant problem, i.e., how to evaluate the skill of dance performance in a mathematical way ? Without a reliable method of evaluation, dance performances by a robot makes no sense in our purpose. It must be a necessary element of evaluation that a robot dancer can actually perform dances while keeping its balance, that is to say, a skill of dynamics. This skill has been achieved to some degree. In addition to this, the skill in appearance is defined, and we will try to achieve a skilled performance in both dynamics and appearance.

References

[1] Ken Endo, Takashi Maeno, and Hiroaki Kitano. Co-evolution of morphology and walking pattern of biped humanoid robot using evolutionary computation - consideration of characteristic of the servomotors -. *Proceedings of International Conference on Intelligent Robots and Systems*, 2002.

[2] R. Agrawal et al. Fast discovery of associattion rules,advances in knowledge discovery and data mining. *MIT Press*, 1996.

[3] Kiyoshi FUJIWARA, Fumio KANEHIRO, Shuji KAJITA, Kenji KANEKO, Kazuhito YOKOI, and Hirohisa HIRUKAWA. Ukemi: Falling motion control to minimize damage to biped humanoid robot. *Proceedings of International Conference on Intelligent Robots and Systems*, 2002.

[4] Motofumi HATTORI, Shinichiro KITADA, Satoshi TADOKORO, and Toshi TAKAMORI. The motion description in computer to make and edit body movement data. *Proceedings of PNC Annual Conference and Joint Meetings 2001 PNC/ECAI/IPSJ-SIGCH/EBTI*, December 2001.

[5] Kazuo Hirai, Masato Hirose, Yuji Haikawa, and Toru Takenaka. The development of honda humanoid robot. *Proceedings of International Conference on Robotics and Automation*, May 1998.

[6] Hirohisa Hirukawa, Fumio Kanehiro, Shuji Kajita, Kazuhito Yokoi, Kenji Kaneko, Kiyoshi Fujiwara, Kazuo Tanie, and Takashi Nagasaki. Dynamics simulation of humanoid robots based on a spring-damper model (in japanese), 2000.

[7] Ann Hutchinson. Labanotation. 1977.

[8] K. Ikeuchi and T. Suehiro. Toward an assembly plan from observation, Part I: Task recognition with polyhedral objects. *IEEE Trans. Robot. Automat.*, Vol. 10, No. 3, pp. 368–385, Jun. 1994.

[9] Katsushi Ikeuchi, Yutaka Takase, Ryo Kurazume, Takeshi Ooishi, Ryusuke Sagawa, and Ko Nishino. Modeling cultural heritage through observation. *International Symposium on Artificial Intelligence, Robotics and Human Centered Technology for Nuclear Applications*, pp. 26–32, 2002.

[10] Tetsunari Inamura, Yoshihiko Nakamura, Hideaki Ezaki, and Iwaki Toshima. Imitation and primitive symbol acquisition of humanoids by the integrated mimesis loop. *Proceedings of International Conference on Robotics and Automation*, May 2001.

[11] Tetsunari Inamura, Iwaki Toshima, and Yoshihiko Nakamura. Acquisition and embodiment of motion elements in closed mimesis loop. *Proceedings of International Conference on Robotics and Automation*, 2002.

[12] 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. *Proceedings of the First IEEE-RAS International Conference on Humanoid Robots*, 2000.

[13] Satoshi KAGAMI, Koichi NISHIWAKI, James J. Kuffner, Tomomichi SUGIHARA, Masayuki INABA, and Hirochika INOUE. Design and implementation of humanoid h6 and its application to remote operation. *In Experimental Robotics VII, Lecture Notes in Control and Information Sciences 271*, pp. 41–50, 2001.

[14] Shuji Kajita. A realtime pattern generator for biped walking. *Proceedings of International Conference on Robotics and Automation*, 2002.

[15] Shuuji Kajita, Fumio Kanehiro, Kenji Kaneko, Kazuhito Yokoi, and Hirohisa Hirukawa. The 3d linear inverted pendulum mode: A simple modeling biped walking pattern generation. *Proceedings of International Conference on Intelligent Robots and Systems*, 2001.

[16] Shuuji Kajita, Osamu Matsumoto, and Muneharu Saigo. Real-time 3d walking pattern generation for a biped robot with telescopic legs. *Proceedings of International Conference on Robotics and Automation*, May 2001.

[17] Fumio Kanehiro. Virtual humanoid robot platform to develop controllers of real humanoid robots without porting. *Proceedings of International Conference on Intelligent Robots and Systems*, 2001.

[18] Fumio KANEHIRO, Kiyoshi FUJIWARA, Shuuji KAJITA, Kazuhito YOKOI, Kenji KANEKO, Hirohisa HIRUKAWA, Yoshihiko NAKAMURA, and Katsu YAMANE. Open architecture humanoid robotics platform. *Proceedings of International Conference on Robotics and Automation*, 2002.

[19] Kenji KANEKO, Fumio KANEHIRO, Shuuji KAJITA, Kazuhiko YOKOYAMA, Kazuhiko AKACHI, Toshikazu KAWASAKI, Shigehiko OTA, and Takakatsu ISOZUMI. Design of prototype humanoid robotics platform for hrp. *Proceedings of International Conference on Intelligent Robots and Systems*, 2002.

[20] Takashi NAGASAKI, Shuuji KAJITA, Kazuhito YOKOI, Kenji KANEKO, and Kazuo TANIE. Running pattern generation for a humanoid robot. *Proceedings of 19th Annual Conference of the Rbotics Society of Japan*, 2000.

[21] Minako Nakamura and Kozaburo Hachimura. Labanotation and new technology: Application of hypermedia to choreography and dance education, 2000.

[22] Koichi NISHIWAKI, Satoshi KAGAMI, Yasuo Kuniyoshi, Masayuki INABA, and Hirochika INOUE. Online generation of humanoid walking motion based on a fast generation method of motion pattern that follows desired zmp. *Proceedings of International Conference on Intelligent Robots and Systems*, 2002.

[23] Koichi NISHIWAKI, Tomonobu KITAGAWA, Tomomichi SUGIHARA, Satoshi KAGAMI, Masayuki INABA, and Hirochika INOUE. Fast generation method of dynamically stable trajectory of humanoid motion base on the characteristics of zmp (in japanese). *Proceedings of 19th Annual Conference of the Rbotics Society of Japan*, 2000.

[24] Masafumi OKADA, Yoshihiko NAKAMURA, and Shin ichiro HOSHINO.

[25] Nancy S. Pollard, Jessica K.Hodgins, Marcia J. Riley, and Christopher G. Atkeson. Adapting human motion for the control of a humanoid robot. *Proceedings of International Conference on Robotics and Automation*, 2002.

[26] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brain P. Flannery. Numerical recipes in c. 1988.

[27] Asako SOGA. Web3d dance composer: A web-based ballet performance simulation system. *Proceedings of 11th International Symposium on Electronic Art*, pp. 16–19, 2002.

[28] Seyoon Tak, Oh young Song, and Hyeong-Seok Ko. Motion balance filtering. *Proceedings of EUROGRAPHICS*, Vol. 19, No. 3, 2000.

[29] Jun Takamatsu, H. Tominaga, Koichi Ogawara, Hiroshi Kimura, and Katushi Ikeuchi. Symbolic representation of trajectories for skill generation. *Proceedings of International Conference on Robotics and Automation*, Vol. 4, pp. 4077–4082, 2000.

[30] Yukiharu Tamiya, Masayuki Inaba, and Hirochika Inoue. Realtime balance compensation for dynamic motion of full-body humanoid standing on one leg (in japanese). *Journal of Robot Society of Japan*, Vol. 17, No. 2, pp. 112–118, 1999.

[31] Miomir Vukobratovic, B.Borovac, D.Surla, and D.Stokic. Biped locomotion: Dynamics, stability, control and application. *volume 7 of Scientific Fundamentals of Robotics*, 1990.

[32] Katsu Yamane and Yoshihiko NAKAMURA. Dynamics filter - concept and implementation of on-line motion generator for human figures. *Proceedings of International Conference on Robotics and Automation*, 2000.

[33] Katsu YAMANE and Yoshihiko NAKAMURA. Efficient parallel dynamics computation of human figures. *Proceedings of International Conference on Robotics and Automation*, May 2002.

[34] Kazuhito Yokoi, Fumio Kanehiro, Kenji Kaneko, Kiyoshi Fujiwara, Shuji Kajita, and Hirohisa Hirukawa. A honda humanoid robot controlled by aist software. *Proceedings of the IEEE-RAS International Conference on Humanoid Robots*, 2001.

[35] Takashi Yukawa, Takaaki Kaiga, Kazuo Nagase, and Hideo Tamamoto. Human motion description system using buyo-fu (in japanese). *Information Processing Society of Japan*, 2000.

[36] 高西淳夫. 上体の運動によりモーメントを補償する 2 足歩行ロボット. 日本ロボット学会誌, Vol. 11, No. 3, pp. 348–353, 1993.