Dynamics Filter: Towards Real-Time and Interactive Motion Generator for Human Figures

Katsu YAMANE[*1] and **Yoshihiko NAKAMURA**[*1*2]
(E-mail: [katz, nakamura]@ynl.t.u-tokyo.ac.jp)

[*1] Department of Mechano-Informatics, University of Tokyo
7-3-1 Hongo Bunkyo-ku Tokyo, 113-8656 JAPAN

[*2] CREST

Abstract

This paper describes the outline of our research related to motions of human figures. Our final goal is to develop a motion generating system which is capable of creating realistic motions of human figures interacting with the user in real time, using techniques based on computational methods for the dynamics and kinematics of kinematic chains. The application areas include humanoid robots and human characters in computer graphics. Towards this goal, we propose the concept of *dynamics filter* which generates physically natural motions from any reference motion allowing interactive inputs by the user. Dynamics filter has a potential ability to generate virtually infinite variety of motion from relatively small motion database or even motions captured from human in real time. We have also developed a prototype of real-time dynamics filter, which makes use of our efficient methods for computing the dynamics of human figures.

Introduction

Human figures are defined as kinematic models of human body with links connected by mechanical joints (figure 1). Motion generation of human figures is of great interest in robotics as well as in computer graphics. Humanoid robots, an embodiment of human figure in the real world, are expected to work in place of human in various fields such as welfare, plant maintenance, entertainment, and so on. In computer graphics, demands for contents of high quality with human characters are growing stronger along with the development and spread of multi-media technologies. The common keyword connecting these two issues is the motion of human figure. Motions of humanoid robots are neither simple nor predefined as in conventional industrial robots — we have to control the unstable body of robots to create almost infinite variation of motions. This situation is quite similar to human characters in computer animations, where we have to make various motions according to their role in virtual environments.

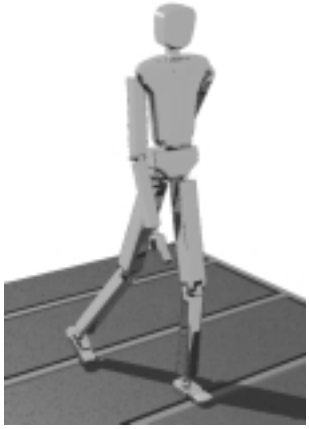

Figure 1: Human figure

In order to achieve this goal, we propose the concept of *dynamics filter*, a real-time and interactive motion generator for human figures. Dynamics filter is a concept quite different from conventional motion generation methods for humanoid robots or computer animations in that it allows complicated motion such as human motion captured data and also interactive inputs from the user. The basic function of the filter is to convert a physically inconsistent motion into a consistent one, to which following features are added:

1. Take human motions, either captured or drawn, as reference instead of creating from scratch, in order to generate human-like motions with expressive and emotional looking
2. Regard physical feasibility, or consistency, as minimum requirement for generated motions, in order to make the motion look natural and facilitate the control of humanoid robots
3. Execute the computation for conversion in real time in order to allow interactive inputs or adapt to dynamically changing environments

In this paper, we propose the concept of dynamics filter and introduce our first implementation. The implementation is based on our previous research on dy-

namics of structure-varying kinematic chains (Yamane & Nakamura 1999b), which is also an interesting problem regarding dynamics of human figures. Therefore, before describing the implementation of dynamics filter in detail, we first provide a summary of the methods for computing the dynamics of human figures, followed by the details on our dynamics filter. The implementation presented here, although still a prototype and needs more improvements, proves the usability of the concept. We also provide some examples of applying the dynamics filter to various motions. In the last section, we introduce our ongoing projects along with the conclusions of this paper.

The Concept of Dynamics Filter

Dynamics filter is the concept of motion generator for human figures which takes an input motion with physical inconsistency and converts it into a physically consistent one considering the dynamics. The input motion may be provided by motion capturing, interpolation of hand-drawn keyframes, numerical functions, or any kinematic synthesis of them. Thus provided, the input motion is commonly physically inconsistent with the dynamic model of the human figure. The dynamics filter takes care of the physical consistency considering the mass properties of the human figure, environments, and so on, and converts the input motion into a physically feasible one. The feasibility will help us not only to control a humanoid robot with simple controller but to make motions of human characters look natural.

Many methods have been proposed for generating motions for humanoid robots (Park & Rhee 1998; Huang *et al.* 1999; DasGupta & Nakamura 1999) and animations (Popovic & Witkin 1999; Ko & Badler 1996; Panne 1997). However, most of them have problems from the point of view of interactivity:

1. Poor flexibility — Each method is only applicable to limited motions, walking in most cases, which means that we need to prepare many motion generators corresponding to different motions.
2. Off-line computation — The whole sequence of motion is required before generating motion. This would be a fatal disadvantage in interactive motion generation because we cannot modify the reference motion once the computation starts.
3. Long computation time — Another problem is that they require long computation time for generating single sequence of motion.

Dynamics filter is expected to provide a solution for these problems. The procedure for generating motion by dynamics filter is illustrated in figure 2. First, several properties, such as the motion (walk / run / sit ...), the model (mass / link length ...), character (male / female, adult / child, ...) and emotion (happy / angry / sad ...) are selected and combined kinematically. Next, the combined reference motion is input to the dynamics filter, which outputs a physically consistent

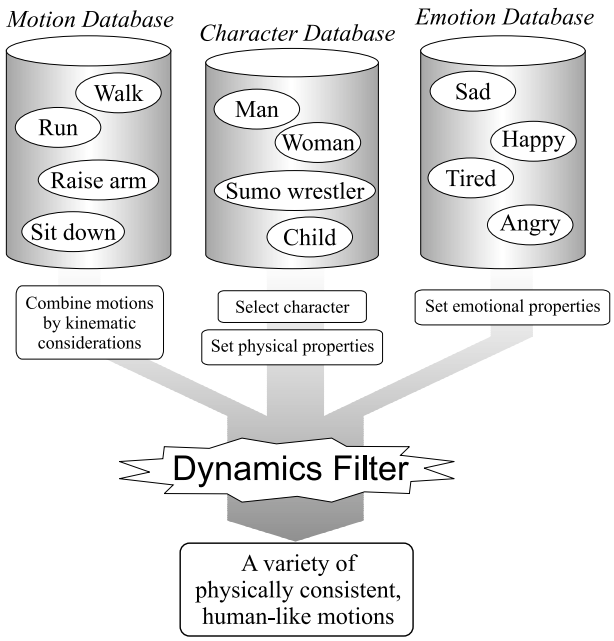

Figure 2: Motion generation through dynamics filter

motion close to the reference. Users may take some trial-and-error processes between the dynamics filter to meet their taste, especially in creating animation. In interactive systems, the reference motion may change during the computation according to user inputs.

Implementation of the filter may be off-line or on-line. Off-line filter, which requires the whole sequence of the input motion prior to the filtering, will generate motions with high quality and stability thanks to the consideration of global status. This type of dynamics filter would be good for creating artistic films in computer graphics, or motion library for humanoid robots. Some previous researches have already realized this type of dynamics filter (DasGupta & Nakamura 1999; Popovic & Witkin 1999; Ko & Badler 1996; Panne 1997).

On-line version of dynamics filter is eventually more difficult, but interesting from the viewpoint of interactive motion generation. Only the previous and current status is provided to on-line dynamic filter, making it extremely difficult to generate globally stable motion, but allowing interactive modification of input motion. This feature is essential for humanoid robots moving in dynamically changing environments and human characters in some applications such as games.

We provide an on-line implementation of dynamics filter in this paper. It is based on the equation of motion of closed and structure-varying kinematic chains developed for dynamics simulation of human figures (Yamane & Nakamura 1999b). In the next two sections we first provide the basic techniques for computing the dynamics of human figures, then proceed to the implementation of dynamics filter.

Dynamics Computation of Human Figures

Previous Works and Requirements

Since human figure is a form of kinematic chains, we can apply algorithms developed in multibody dynamics and robotics (Haug 1989; de Jalon & Bayo 1993; Featherstone 1987; Chang & Khatib 1999) to human figures as well (McMillan & Orin 1998; Bruneau & Ouezdou 1998; Perrin, Chevallereau, & Verdier 1997). However, human figures have quite different properties compared to conventional robot manipulators from the point of view of dynamics computation as listed below:

1. Large degree of freedom (DOF) — Human figures usally contain many DOF's, from twenty of the simplest model to more than forty, even when fingers are ignored.
2. Complicated closed kinematic chains — Human figures often form complicated closed kinematic chains by holding links in the environment, own body, or other figures, for which most dynamics algorithms require large computational load.
3. Structure changes — On catch or release of links by hands, human figures change their link structure dynamically during the motion.
4. Collisions and contacts — Human figures frequently collide with the environments, other human, or even itself during motion.
5. Under actuation — Since human figures have no fixed link, they are always under actuated, i.e. the DOF of motion is larger than the number of actuators. Therefore, we need to consider the physical consistency of motion in generating motions for human figures.
6. Requirement for interactivity — When future applications of human figures are considered, interactivity would be the most important feature. Human figures are expected to move in dynamically changing environment interacting with human, opposite to conventional manipulators which are usually operated in static environments where humans are kept out.

Taking these aspects into account, we decided to take a new approach towards the dynamics computation of human figures. Our objectives are:

1. To compute the dynamics of complicated kinematic chains efficiently, as fast as real time if possible
2. To handle open and closed kinematic chains seamlessly to enable on-line computation of structure-varying kinematic chains
3. To compute the dynamics of collisions and contacts efficiently

This section describes our methods for computing the dynamics of phenomena observed in motions of human figures (Yamane & Nakamura 1999b; 1999a).

Closed Kinematic Chains

Dynamics of closed kinematic chains requires consideration of reaction forces in closed loops. Lagrange multipliers are often applied to compute the reaction force (Haug 1989; de Jalon & Bayo 1993). However, computational cost of Lagrange multipliers is too large to be applied to real-time or interactive simulation.

Alternative approach is to apply the principle of virtual work (Nakamura & Ghodoussi 1989; Codourey & Burdet 1997), which are mostly used to control robots with closed kinematic chains such as planar five-bar linkage or parallel mechanisms (Yamane *et al.* 1998). This method has the advantage of high computational efficiency which enables real-time control of manipulators. The problem is, however, that no general algorithm is known for computing the Jacobian matrix essential for applying the principle of virtual work, for which we provided a solution (Yamane & Nakamura 1999a). Our method can compute the Jacobian matrices used in the principle of virtual work for any closed kinematic chains.

The basic equation of motion of human figures is described as:

$$\begin{pmatrix} \boldsymbol{A} & -\boldsymbol{H}_C^T & -\boldsymbol{H}_J^T \\ \boldsymbol{H}_C & \boldsymbol{O} & \boldsymbol{O} \end{pmatrix} \begin{pmatrix} \ddot{\boldsymbol{\theta}}_G \\ \boldsymbol{\tau}_C \\ \boldsymbol{\tau}_J \end{pmatrix} = \begin{pmatrix} -\boldsymbol{b} \\ -\dot{\boldsymbol{H}}_C \dot{\boldsymbol{\theta}}_G \end{pmatrix} \tag{1}$$

which can be summarized as:

$$\boldsymbol{W}\boldsymbol{x} = \boldsymbol{u} \tag{2}$$

where

$\boldsymbol{A}$	:	inertial tensor
$\boldsymbol{b}$	:	centrifugal, Coriollis and gravity forces
$\boldsymbol{\theta}_G$	:	the generalized coordinates
$\boldsymbol{\tau}_C$	:	constraint forces at connected joints
$\boldsymbol{\tau}_J$	:	joint torques
$\boldsymbol{H}_C$	=	$\partial\boldsymbol{\theta}_C/\partial\boldsymbol{\theta}_G$
$\boldsymbol{H}_J$	=	$\partial\boldsymbol{\theta}_J/\partial\boldsymbol{\theta}_G$
$\boldsymbol{\theta}_J$	:	joint angles

and $\boldsymbol{\theta}_C$ is the variable that represents the constraint condition by $\ddot{\boldsymbol{\theta}}_C = \boldsymbol{O}$. If joint torques $\boldsymbol{\tau}_J$ were known, we can compute the generalized acceleration $\ddot{\boldsymbol{\theta}}_G$ and constraint forces $\boldsymbol{\tau}_C$ by equation (1).

Figure 3 shows an example of dynamics simulation of closed kinematic chain, where a human figure is trying to raise his body with his arms. The human figure has 28DOF and each rope consists of spherical and rotational joints. The figure holds the ends of ropes by spherical joints, forming a closed kinematic chain. Forward dynamics computation for this 48DOF system, including the six DOF of the base body, takes approximately 32msec on DEC Alpha 21264 500MHz processor.

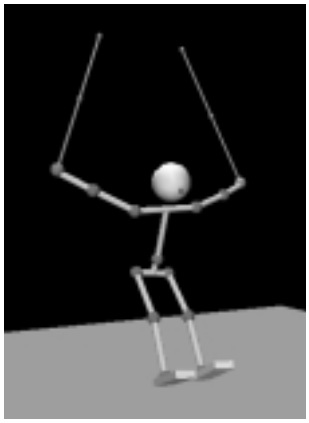

Figure 3: Dynamics simulation of closed kinematic chain

Structure-Varying Kinematic Chains

Most conventional methods for dynamics computation of kinematic chains assume that the link connectivity of the system does not change during the simulation. However, this is obviously not the case in human motions. In fact, we catch and release objects in the environment, other human, or ourselves. Walking, the most common motion of human, can be said to have three different kinematic chains. We call such systems *structure-varying* kinematic chains, and regard as one of key issues in dynamics of human figures. To handle such situations in conventional methods, we would need to prepare all the possible structures in advance and switch among them during the simulation, which is impossible in interactive applications.

In order to solve this problem, we developed two techniques regarding the description of link connectivity (Yamane & Nakamura 1999b). The first is the method of description itself, where we use three *pointers* ("parent", "child" and "brother") per link to indicate its neighboring links as illustrated in figure 4. However, these pointers are not capable of describing closed kinematic chains since parent-child relationship makes infinite loop. To describe closed kinematic chains, we virtually cut a joint in a loop and introduce an additional link called *virtual link* to describe the connection at the virtually cut joint. Another pointer, *real*, is added to the three pointers, to hold the relationship of real and virtual links. The concept is illustrated in figure 5.

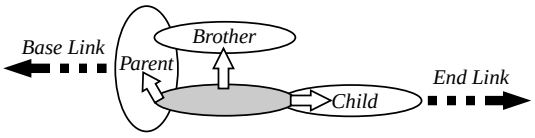

Figure 4: Three pointers to describe link connectivity

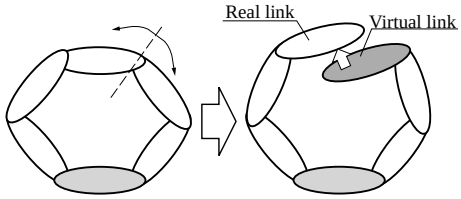

Figure 5: Virtual link to describe closed kinematic chains

The second is the maintenance of pointers, i.e. how to modify the connectivity data upon structure changes. Thanks to the introduction of virtual links, the procedure is quite simple as follows:

1. Link connection — If two links are connected by a new joint, we only need to create a new virtual link whose parent is one of the connected links, and set its real pointer to the other.
2. Joint cut — If a joint is cut, then we simply delete the virtual link associated with the cut joint.

By these simple procedures, link connectivity data is automatically changed according to the change of kinematic chain.

Each human figure is described as an independent open chain at the initial state. If a link in the figure is connected to another one, a virtual link is generated at the connection point following the procedure described above. Therefore, we do not need to reconstruct or reorder the link connectivity of human figures even if link connections or joint cuts occur.

The dynamics compuation part is programmed to generate the equation of motion automatically based on the link connectivity. Thus, arbitrary structure changes are allowed and simulated without any manual tasks by the user. In dynamics filter also, this description enables the filter to use the same strategy throughout the motion, without worrying about the change of structure.

Figure 6 is an example of interactive application making use of these techniques. When the user clicks the mouse button, the monkey releases his hand one by one and finally falls down.

Collisions and Contacts

Contacts can be seen as a sort of structure changes, because additional constraints appear at the contact point. The differences from joint connection are:

1. Transition of constraint — Constraints change discontinuously according to the contact state. In point contact, for example, only the relative position is constrained, but if the contact moves into face contact state, the relative orientation is also constrained.
2. Unilaterality — Constraint condition changes depending on the contact force due to its unilateral conditions. As for the direction normal to the contact face, there are only two possibilities: maintain

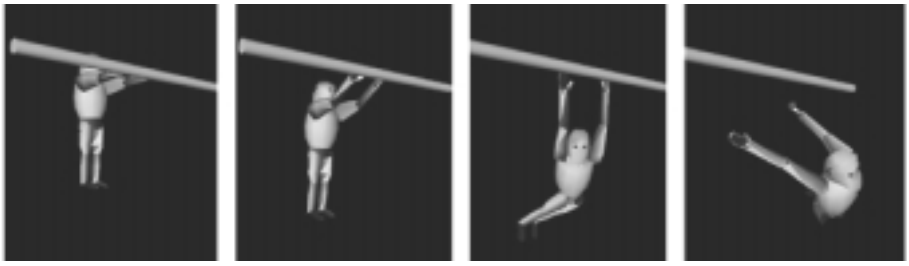

Figure 6: Dynamics simulation of structure-varying kinematic chain

contact with compressive force, or get apart without force. We need to watch the constraint force to determine the constraint.

Collision and contact models are categorized into two types. Visco-elastic body model, also known as penalty model, places virtual spring and damper at the contact point by which contact forces are exerted(Marhefka & Orin 1996; Kraus, Kunmar, & Dupont 1998). Rigid body model, on the other hand, finds the contact force that satisfies the unilateral conditions through some optimization processes (Pfeiffer & Glocker 1996). These methods focus on precise analysis of collisions and contacts for applications where effects of these phenomena are critical to the performance of the tasks such as manipulation by fingers. In human figures, however, preciseness of the contact model is not very important for the motion, because most contacts can be considered as stick contact and bouncing is rarely observed.

Our method is based on rigid body model in the sense that we do not consider deformation of links. However, to avoid complicated optimization process and reduce the computaion, we apply a simplified computation with iterative procedure.

The method is divided into two parts. The first part, which is activated if $\dot{\boldsymbol{\theta}}_C \neq \boldsymbol{O}$, i.e. if collision occurs, computes the discontinuous change of joint velocities using conservation of linear and angular momentum. This approach has the advantage of numerical stability because large impact forces are not handled. The second part, computes the constraint forces to maintain the constraint condition $\ddot{\boldsymbol{\theta}}_C = \boldsymbol{O}$ using equation (1). However, the computed constraint forces may not meet the unilateral conditions of contact forces. Therefore, we need to check the consistency of constraint forces and re-compute with different constraints if necessary.

The outline of the method is summarized as follows:

1. Compute the discontinuous change of joint velocities
2. Starting from full constraint, relative position and orientation constrained in case of face contact, compute the constraint force to maintain the constraint by the equation of motion (1)
3. Check the feasibility of the computed contact force.
4. If infeasible constraint forces were found, modify the constraint and return to 1.
5. If all constraint forces were feasible, accept the constraint.

This method is more efficient than usual rigid body methods for two reasons: (1) the number of iteration is up to 4, and (2) computation for each iteration step is very small.

Dynamics Filter Implementation

Basic Equations

The developed filter consists of two parts — feedback control and optimization based on equation (1).

The control part computes the desired joint accelerations considering the reference motion, current state, and stability. First initial desired acceleration of generalized coordinates $\ddot{\boldsymbol{\theta}}_G^{d0}$ is computed by simple feedback of joint angle and velocity:

$$\ddot{\boldsymbol{\theta}}_G^{d0} = \ddot{\boldsymbol{\theta}}_G^{ref} + \boldsymbol{K}_D(\dot{\boldsymbol{\theta}}_G^{ref} - \dot{\boldsymbol{\theta}}_G) + \boldsymbol{K}_P(\boldsymbol{\theta}_G^{ref} - \boldsymbol{\theta}_G) \quad (3)$$

where $\boldsymbol{\theta}_G^{ref}$ is the generalized coordinates in reference motion, and $\boldsymbol{K}_D$ and $\boldsymbol{K}_P$ are constant gain matrices.

Then, in order to consider the global stability, feedback of position and orientation of a specified point $\boldsymbol{P}$ in the upper body are included as follows. The desired acceleration of $\boldsymbol{P}$, $\ddot{\boldsymbol{\theta}}_P^d$, is computed by a similar feedback law as:

$$\ddot{\boldsymbol{r}}_P^d = \ddot{\boldsymbol{r}}_P^{ref} + \boldsymbol{K}_{DP}(\dot{\boldsymbol{r}}_P^{ref} - \dot{\boldsymbol{r}}_P) + \boldsymbol{K}_{PP}(\boldsymbol{r}_P^{ref} - \boldsymbol{r}_P) \quad (4)$$

where $\boldsymbol{r}_P^{ref}$ is the position and orientation of $\boldsymbol{P}$ in the reference motion which can be obtained by forward kinematics computation, $\boldsymbol{K}_{DP}$ and $\boldsymbol{K}_{PP}$ are constant gain matrices, and $\boldsymbol{r}_P$ is current position and orientation of $\boldsymbol{P}$. The initial desired acceleration of the generalized coordinates $\ddot{\boldsymbol{\theta}}_G^{d0}$ is modified into $\ddot{\boldsymbol{\theta}}_G^{d}$ so that the desired acceleration of $\boldsymbol{P}$, $\ddot{\boldsymbol{r}}_P^d$ is realized :

$$\ddot{\boldsymbol{\theta}}_G^d = \ddot{\boldsymbol{\theta}}_G^{d0} + \Delta\ddot{\boldsymbol{\theta}}_G^d \quad (5)$$

$$\Delta\ddot{\boldsymbol{\theta}}_G^d = \boldsymbol{J}_P^{\sharp}(\ddot{\boldsymbol{r}}_P^d - \ddot{\boldsymbol{r}}_P^{d0}) \quad (6)$$

where $\ddot{\boldsymbol{r}}_P^{d0} \triangleq \boldsymbol{J}_P\ddot{\boldsymbol{\theta}}_G^{d0} + \dot{\boldsymbol{J}}_P\dot{\boldsymbol{\theta}}_G$, $\boldsymbol{J}_P$ is the Jacobian matrix of $\boldsymbol{P}$ with respect to the generalized coordinates, and $\boldsymbol{J}_P^{\sharp}$ is the weighted pseudo inverse of $\boldsymbol{J}_P$.

The optimization part computes the feasible accelerations close to the desired based on equation (1). Solutions of equation (1) represents all the feasible combination of $\ddot{\boldsymbol{\theta}}_G, \boldsymbol{\tau}_C$ and $\boldsymbol{\tau}_J$. The task of the optimization part is to find the optimal solution of equation (1) to realize the desired acceleration. The optimized accelerations are integrated to derive the joint angle data. Details of the process are described below.

First, in preparation for the optimization, we derive the weighted least-square solution of equation (2) and the null space of $\boldsymbol{W}$ regardless of the desired acceleration:

$$\boldsymbol{x} = \boldsymbol{W}^\sharp \boldsymbol{u} + (\boldsymbol{E} - \boldsymbol{W}^\sharp \boldsymbol{W})\boldsymbol{y} \tag{7}$$

where $\boldsymbol{W}^\sharp$ is the pseudo inverse of $\boldsymbol{W}$, $\boldsymbol{y}$ an arbitrary vector, and $\boldsymbol{E}$ the identity matrix of the appropriate size. Picking up the upper rows of equation (7) corresponding to the generalized accelerations, we get:

$$\ddot{\boldsymbol{\theta}}_G = \ddot{\boldsymbol{\theta}}_G^0 + \boldsymbol{V}_G \boldsymbol{y} \tag{8}$$

where $\ddot{\boldsymbol{\theta}}_G^0$ is the generalized acceleration in the least-square solution.

Next, we determine the arbitrary vector $\boldsymbol{y}$ to minimize the acceleration error by

$$\boldsymbol{y} = \boldsymbol{V}_G^*(\ddot{\boldsymbol{\theta}}_G^d - \ddot{\boldsymbol{\theta}}_G) \tag{9}$$

where $\boldsymbol{V}_G^*$ is the singularity-robust inverse(Nakamura & Hanafusa 1986) of $\boldsymbol{V}$.

Finally, substituting the calculated $\boldsymbol{y}$ into equation (7), we get the optimized solution of $\boldsymbol{x}$. Since $\boldsymbol{x}$ includes the generalized acceleration, joint torques and constraint forces all in one, we can say that optimization part plays three roles at the same time: (1) computation of optimized motion, (2) computation of joint torques to realize the computed acceleration, and (3) dynamics simulation of the result.

Applications

In the following examples, the additional control point $\boldsymbol{P}$ was taken at the neck and its position and orientation were computed off-line, although it is easy to compute them on-line.

Filtering Raw Motion Captured Data Figure 7 compares the captured (above) and filtered (below) walking motions. Although small latency is observed in the filtered motion, the result is satisfactory.

This method is applicable to any motion as shown in figure 8, which means that we do not need to prepare different filters for different motions.

Filtering into Different Environment We also tried to apply the same walking motion as in figure 7 to a different environment, in this case a slope, and see what happens. The result is shown in figure 9. No modification on captured data was made except for the position of the body, which was modified to maintain the same height from the ground.

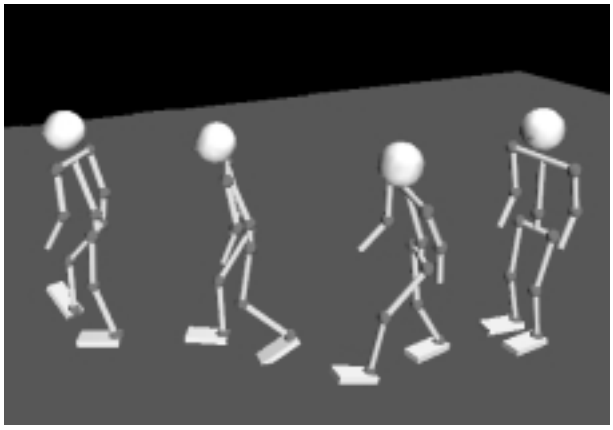

Figure 10: Motion generated from kinematically combined motion

Filtering Kinematically Synthesized Motion The dynamics filter accepts not only raw captured data but also kinematic combination of captured motions. The human figure in figure 10 makes a turn of 30 degrees, by simply giving motion obtained by smoothly connecting two walking motions heading different directions as reference. In the reference motion, only the orientation of the body is interpolated and no dynamic effects such as centrifugal force are considered. This result shows that we can make a human figure walk along any path by indicating the direction using some input device.

Interactive Motion Generation Note that the optimization applied here is strictly local to each frame, which means that this filter has a potential ability to realize real-time and interactive motion generator. Although we cannot call it as "real-time dynamics filter" because the computation takes 70 to 80 msec per frame with Alpha 21264 500MHz processor, we tried to interact with the figure filter as in figure 11, where the figure is controlled to keep standing by the dynamics filter, and pushed in various directions by the user.

Conclusions

This paper presented our research issues towards motion generator for human figures. The conclusions of this paper are summarized as follows:

1. The concept of dynamics filter was proposed. Dynamics filter is a motion generator that creates a physically consisntent motion from any reference motion, allowing interactive inputs from the user.
2. Basic dynamics computation methods used in the implementation of dynamics filter were presented. The methods can handle various phenomena observed in motions of human figures, such as:
   (a) General closed kinematic chains
   (b) Structure-varying kinematic chains
   (c) Collisions and contacts

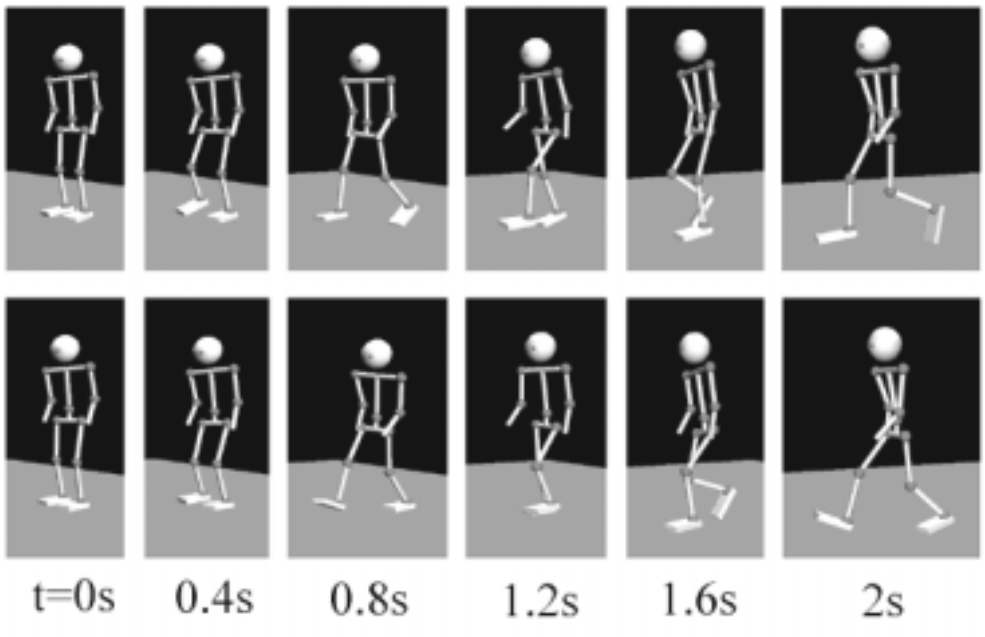

Figure 7: Captured (above) and filtered (below) walking motions

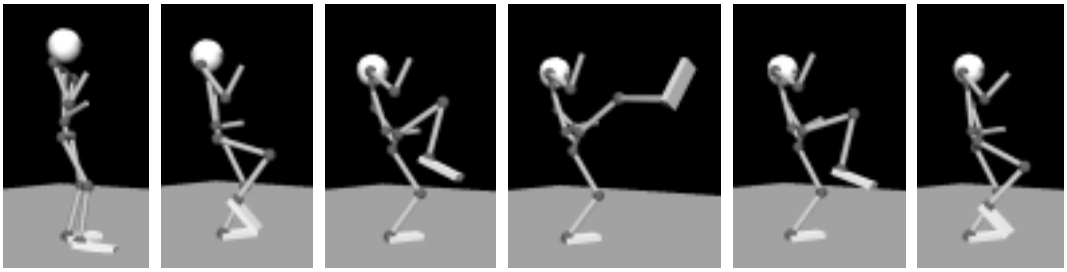

Figure 8: Karate kick generated by the dynamics filter

3. An implementation of on-line dynamics filter was also introduced and proved the potential ability of dynamics filter by examples of motions generated from motion captured data.

Currently we have several projects related to this issue:

1. Improvement of dynamics filter in its ability and computation time, towards real-time interactive system
2. Development of behavior capture system, which not only captures the motion of human but also his 3D shape, interaction with the environments, and *mind* via various sensors
3. Controlling a real humanoid robot using the behavior capture system as input device to make the robot imitate the motion of human

Acknowledgements This research was supported by Humanoid Robotics Project, NEDO, Japan, and CREST Program of the Japan Science and Technology Corporation.

References

Bruneau, O., and Ouezdou, F. 1998. Dynamic Walk Simulation of Various Bipeds via Ankle Trajectory. In *Proceedings of the 1998 IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, 58–63.

Chang, K., and Khatib, O. 1999. Efficient Algorithm for Extended Operational Space Inertia Matrix. In *Proceedings of IEEE International Conference on Robotics and Automation*, 350–355.

Codourey, A., and Burdet, E. 1997. A Body-Oriented Method for Finding a Linear Form of the Dynamic Equation of Fully Parallel Robots. In *Proceedings of IEEE International Conference on Robotics and Automation*, 1612–1618.

DasGupta, A., and Nakamura, Y. 1999. Making Feasible Walking Motion of Humanoid Robots from Human Motion

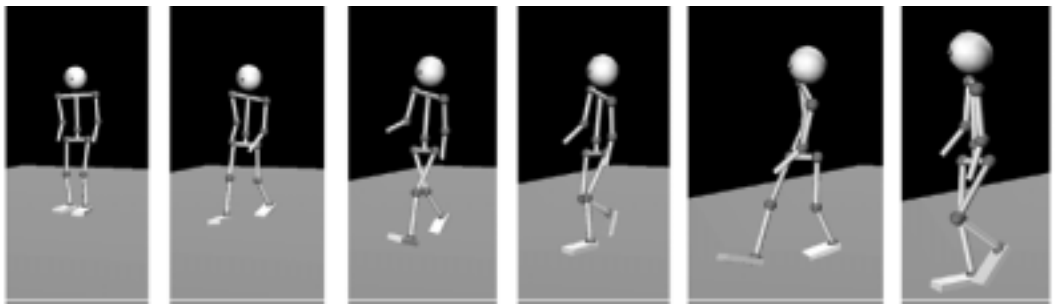

Figure 9: Walk on a slope

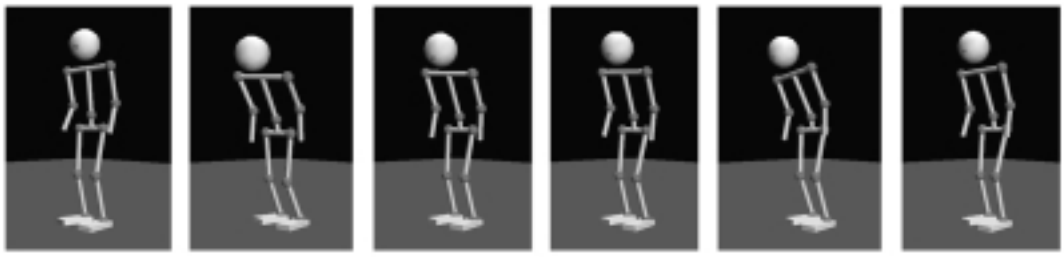

Figure 11: Example of interactive motion generation: push a standing figure

Captured Data. In *Proceedings of International Conference on Robotics and Automation*, 1044–1049.

de Jalon, J. G., and Bayo, E. 1993. *Kinematic and Dynamic Simulation of Multibody Systems – the Real-Time Challenge*. Springer – Verlag.

Featherstone, R. 1987. *Robot Dynamics Algorithm*. Kluwer.

Haug, E. 1989. *Computer Aided Kinematics and Dynamics of Mechanical Systems*. Allyn and Bacon Series in Engineering.

Huang, Q., and Kajita, S. *et al.* 1999. A High Stability, Smooth Walking Pattern for a Biped Robot. In *Proceedings of International Conference on Robotics and Automation*, 65–71.

Ko, H., and Badler, N. 1996. Animating Human Locomotion with Inverse Dynamics. *IEEE Transactions on Computer Graphics* 16(2):50–59.

Kraus, P.; Kunmar, V.; and Dupont, P. 1998. Analysis of Frictional Contact Models for Dynamic Simulation. In *Proceedings of the 1998 IEEE International Conference on Robotics and Automation*, 976–981.

Marhefka, D., and Orin, D. 1996. Simulation of Contact Using a Nonlinear Damping Model. In *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, 1662–1668.

McMillan, S., and Orin, D. 1998. Forward Dynamics of Multilegged Vehicles Using the Composite Rigid Body Method. In *Proceedings of IEEE International Conference on Robotics and Automation*, 464–470.

Nakamura, Y., and Ghodoussi, M. 1989. Dynamics Computation of Closed-Link Robot Mechanisms with Nonredundant and Redundant Actuators. *IEEE Transactions on Robotics and Automation* 5(3):294–302.

Nakamura, Y., and Hanafusa, H. 1986. Inverse Kinematics Solutions with Singularity Robustness for Robot Manipulator Control. *Journal of Dynamic Systems, Measurement, and Control* 108:163–171.

Panne, M. v. d. 1997. From Footprints to Animation. *Computer Graphics Forum* 16(4):211–223.

Park, J., and Rhee, Y. 1998. ZMP Trajectory Generation for Reduced Trunk Motions of Biped Robots. In *Proceedings of the 1998 IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, 90–95.

Perrin, B.; Chevallereau, C.; and Verdier, C. 1997. Calculation of the Direct Dynamics Model of Walking Robots: Comparison Between Two Methods. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 2, 1088–1093.

Pfeiffer, F., and Glocker, C. 1996. *Multibody Dynamics with Unilateral Contacts*. Wiley Series in Nonlinear Science.

Popovic, Z., and Witkin, A. 1999. Physically based motion transformation. In *Proceedings of SIGGRAPH'99*, 11–20.

Yamane, K., and Nakamura, Y. 1999a. Dynamics Computation of Closed Kinematic Chains for Motion Synthesis of Human Figures. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1108–1114.

Yamane, K., and Nakamura, Y. 1999b. Dynamics Computation of Structure-Varying Kinematic Chains for Motion Synthesis of Humanoid. In *Proceedings of IEEE International Conference on Robotics and Automation*, 714–721.

Yamane, K.; Okada, M.; Komine, N.; and Nakamura, Y. 1998. Parallel Dynamics Computation and H_∞ Acceleration Control of Parallel Manipulators for Acceleration Display. In *Proceedings of 1998 IEEE International Conference on Robotics and Automation*, 2301–2308.