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

Intuitive and Flexible User Interface for Creating Whole Body Motions of Biped Humanoid Robots

Shin'ichiro Nakaoka, Shuuji Kajita and Kazuhito Yokoi

Abstract— **This paper proposes a novel user interface for creating whole body motions of biped humanoid robots just by giving key poses. Although such an interface is popular for CG character animation, there have not been any practical systems that can appropriately handle the kinematic and dynamic conditions required for moving actual biped robots stably without falling down. In our interface, every time a key pose is created, modified or removed, the system immediately processes automatic trajectory adjustment of the feet and the waist so that the key poses and the interpolated motion can always meet the conditions. This enables a user to edit variety of motions in an intuitive and flexible way without paying attention to the conditions. We implemented the interface and confirmed that characteristic whole body motions of a realistic humanoid robot HRP-4C were easily created with it and the actual HRP-4C successfully performed them.**

I. Introduction

Humanoid robots as human-like figures are essentially expected to do various motions as humans do, and recently really human-like robots such as *HRP-4C* [1] (Fig.1), which have a potential to do various whole body motions, have been developed. Needs for creating various motions of them are hence increasing. In order to meet the needs, this study proposes a novel user interface that enables a user who is not necessarily a specialist of robots to create various whole body motions of humanoid robots at will. Our interface design and implementation algorithms make a key-pose [1] based simple edit style available even for biped-type humanoid robots so that the user can easily create motions including leg motions as if he or she were creating CG character animations.

Although our goal is to achieve a CG-like edit-style, CG tools in themselves are not applicable to creating motions of robots which must be moved in the real world. In fact many studies in the CG field deal with physical laws in creating animation of human-like figures, but there is a significant gap between obtaining physically plausible animations and moving actual robots stably because the animation can be accomplished just by rendering each frame but a robot must be controlled against errors in the physical model and low-level control. The gap becomes especially larger for biped-type robots because the physical interactions between the feet and ground significantly affect on the result of whole body trajectory and the errors in it easily cause falling down. Our system sufficiently considers those factors in such a way that a user does not have to pay attention to them.

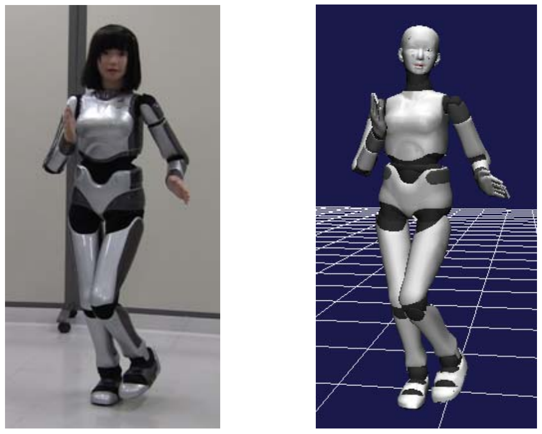
Fig. 1. Cybernetic Human HRP-4C. HRP-4C is a biped humanoid robot that models an average Japanese young female. It is 1580[mm] tall and weights 43[kg]. It has 44 DOF in its whole body. The left image is the actual robot and the right image is the 3D CG model used in the interface.

Using captured human motions to move humanoid robots is an interesting idea and there have been studies on it [2][3][4][5]. The problem in this approach is that the shape, structure, power and weight distribution of a robot body is generally different from those of the original human body. Adapting motions including leg motions into biped-type robots is especially difficult because of the reason described above. For this problem, Nakaoka et al. [5] proposed a method that recognizes lower body tasks including dynamic steps from given motion trajectories and reproduces the tasks stably on an actual robot. The method successfully enabled the biped humanoid robot *HRP-2* [6] to imitate traditional dance motions performed by human dancers [7].

However our interface does not intend the adaptation of existing motion data but it intends direct motion creation for a target robot because of the following reasons. First, a motion capture system and skillful human performers are not always available for a user. Secondly, essentially the adaptation approach does not guarantee that the adapted motion is exactly same as the original motion. Consequently there is a case where a motion designer want to directly create motions from scratch to obtain his/her desired motions in detail. That way is more appropriate to make full use of the characteristics and abilities of the target robot body and to accurately express the creator's original intentions. For this purpose, our interface is designed as an interactive motion editing tool for directly editing the final output. The

S. Nakaoka, S. Kajita and K. Yokoi are with Faculty of Intelligent Systems Research Institute, National Institute of Advanced Industrial Science and Technology (AIST), Tsukuba Central 2, 1-1-1 Umezono, Tsukuba, Ibaraki, Japan {s.nakaoka, s.kajita, k.yokoi}@aist.go.jp

[1]The "key pose" is almost the same concept as the "key frame" in the animation but it only focuses attention on the motion of an individual body.

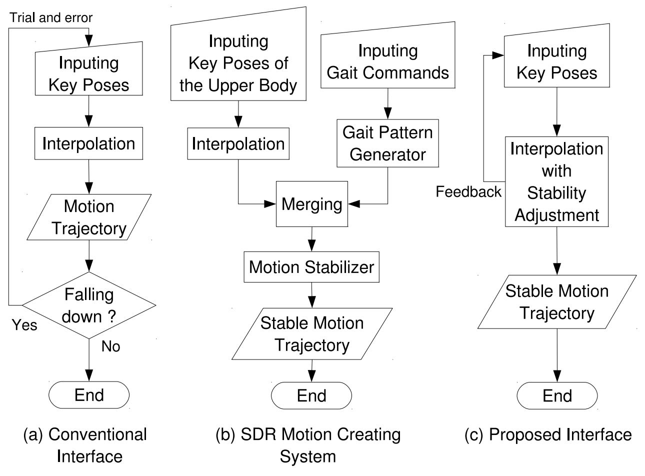

Fig. 2. Comparison of the edit processes.

concrete features and novelty of our interface are described in the following sections.

II. Related Work

Several software tools for manually creating motions of humanoid robots from scratch have been developed [8], but most of them are based on a low-level interface for editing a set of joint angles that describes a key pose and for doing straightforward interpolation between key poses in the joint-angle space. Since it does not take care of the kinematic and dynamic consistencies between the body and the floor, created motions usually result in awkward motions or falling down when they are performed by actual robots on the floor. A user must manually do much trial and error so that the created motions can be stably performed (Fig.2-(a)). Moreover, it is not guaranteed that the trial and error finally result in the consistent motion. Such a motion might fortunately be able to be performed by a light-weight robot with relatively large soles, but it is almost impossible to move life-sized robots with relatively small soles such as HRP-4C.

There are few studies on the manual edit interface that takes care of the kinematic and dynamic consistencies between the body and the floor. Kuroki et al. [9] proposed the *SDR Motion Creating System* with such an interface, which was developed for creating motions of a small biped humanoid robot *SDR-4X (QRIO)* [10].

In their system, upper body motions and lower body motions are separately edited first by using different interfaces. The interface for the upper body is like conventional key pose interfaces. On the other hand, the interface for the lower body is based on their *gait pattern generator*, which generates well-formed trajectories of the feet from a given sequence of *gait commands*. A gait command consists of parameters including the landing position / orientation and height / speed of a step. A user directly edits gait commands on the specific interface. After an upper body motion and a gait pattern are edited, they can be merged into the whole body motion and the *motion stabilizer* can also be applied to make the merged motion dynamically consistent one. The above process is shown as Fig.2-(b).

Although their system has not been available publicly, it seems sufficiently practical because when QRIOs were being actively promoted, they had demonstrated various motions that were probably created with their system. However its separative edit style is not necessarily the best solution. In fact it is technically reasonable for the implementation because each separated process can concentrate on its own problem but it would not be intuitive for users because a user cannot directly see the resulting whole body poses when the user is editing upper body motion or gait commands. Especially editing motions where the combinations of upper body motion and lower body motion are important would not be so efficient. In addition, the method based on the gait pattern generator would not flexible because the generator only refers to a limited set of parameters, which would limit the editable lower body motions.

In contrast to their system, our goal is to make a “unified” interface that realizes more intuitive and flexible motion creation for users. The edit process in our proposed interface is roughly illustrated as Fig.2-(c), which achieves both the simplicity of interface (a) and the robustness of interface (b). The details of it are described in the following sections.

III. Design of the Proposed Interface

A. Overview

The most significant feature of our proposed user interface is that the functions for editing stable whole body motions including leg motions are integrated into the operation of “giving key poses”. Realizing this kind of edit style for biped humanoid robots is a novel achievement.

Figure 3 illustrates the overview of the proposed interface with a sample motion being edited on it. First we briefly review the basics of the key-pose editing. The lower images of Fig.3 are key poses. As shown in these images, a key pose is a particular pose at a particular time. The purpose of key poses is to compose the whole motion trajectories with a relatively small number of poses. In this example, about seven second characteristic motion is composed just by giving these eight key poses. The motion is generated so that each key pose appears at the time of it. Editing motions is processed by creating, modifying or removing key poses by using a rich set of GUI components. Various methods for inputting a pose in itself can be provided. As one of the effective methods, there are inverse kinematics methods which allows multi-part constraints with arbitrary weights [11] [12].

Note that a key pose can be associated with an arbitrary set of body parts. In this example, each key pose is associated with a whole body part except for facial parts. Thus facial expressions can be independently edited in this case. Or key poses for arms and for legs can be independently edited if needed.

Our goal is to make a motion being edited in the above edit style directly be a stable motion in terms of the kinematic and dynamic stability of the contact between the body and floor.

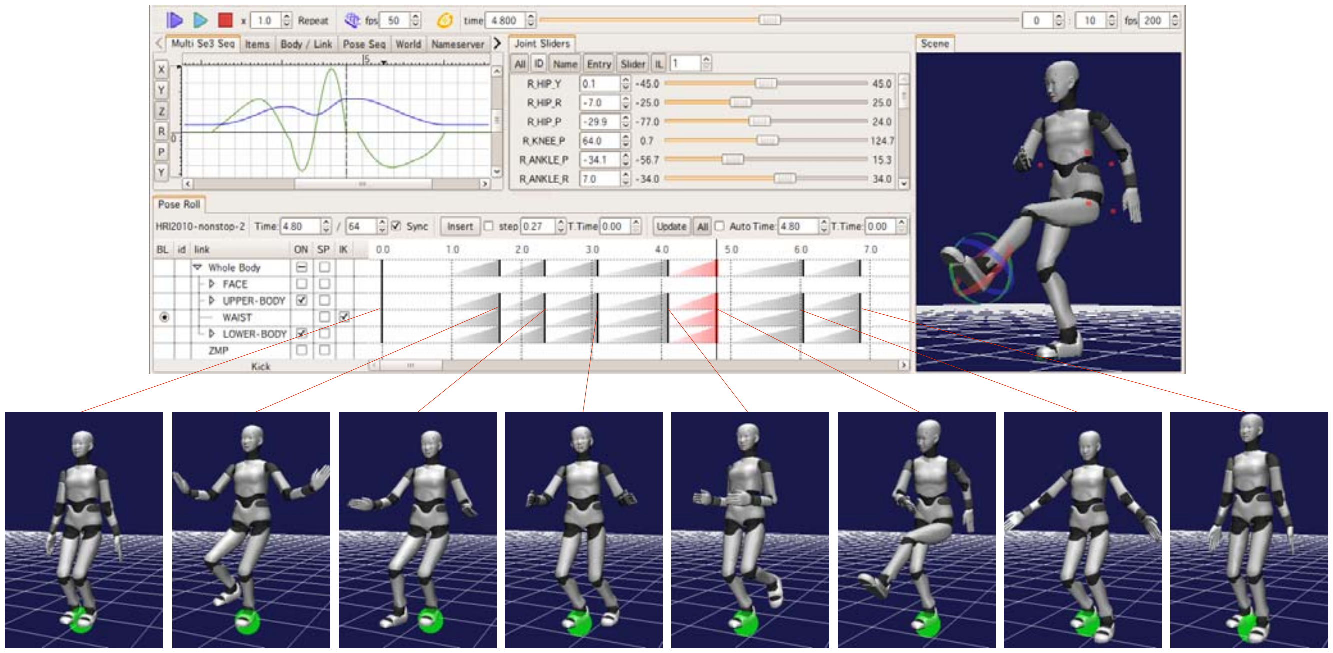

Fig. 3. Our implementation of the proposed interface and key poses being edited on it. The right pane on the tool is the scene view, which visualizes the models of robots and environmental objects with 3D computer graphics. The scene view also allows a user to directly drag each body part to modify the pose. The tool also provides other various way of editing key poses such as joint angle sliders in the upper middle pane. The left bottom pane visualizes a key pose sequence on the time-line. Rows of this view correspond to body parts and the rows can be managed as tree structure. Each triangle corresponds to key poses for each body part. The left vertex of a triangle corresponds to the beginning of the transition and the right edge corresponds to the time of a key pose where the robot completely makes the pose. The lower images are key poses. Just by giving poses like these ones, stable whole body motion including leg motions can be obtained and the motion can be immediately previewed by using the play button or time slider in the upper area. The spheres in the images show ZMPs at key poses, which are automatically determined by the system.

In fact, accepting completely arbitrary key poses or using straightforward interpolation of key poses cannot achieve our goal. Instead, we propose the interface that behaves as follows to achieve the goal:

1) Some factors of a key pose are guided by the interface so that they can satisfy the conditions of the stability:
   a) The horizontal waist position is guided to be dynamically balanced one in the motion.
   b) A foot position can be guided so that it can make appropriate contact to the floor.
2) The trajectories between key poses are generated so that they can satisfy the conditions of the stability:
   a) In combination with 1a, the horizontal waist trajectory is generated so that the global body motion can be dynamically balanced.
   b) Foot trajectories are generated so that it can make a stable contact to the floor
3) The above functions continuously work during the edit process. The guides are synchronized with user's edit operations and the latest trajectories can be confirmed with the motion preview at any time.

This design achieves an intuitive edit style where a user can concentrate on the cycle between editing key poses and checking the appearance of the resulting motion, even though many adjustment processes essentially have to be applied to the generation of the stable motion trajectories from the key poses initially given by a user. Since the types of factors which may be modified by guides are limited to necessary conditions, the flexibility of the key pose editing is not lost.

In the following subsections, we describe the details of the above behaviors and how to implement them is described in Section IV.

B. *Waist Trajectory Adjustment*

The dynamic balance of global body motion are satisfied by the function called *waist trajectory adjustment*, which corresponds to 1a and 2a in Section III-A.

Figure 4 illustrates how the waist trajectory adjustment works. A user can set an arbitrary pose to a key pose but the pose cannot necessarily be a dynamically balanced one as a part of the motion from the precedent key poses to the subsequent ones. In other words, a pose originally given by a user may be unnatural one from the viewpoint of dynamics. In fact, whatever the key poses are, the motion will be dynamically unbalanced one if the motion is determined by a straightforward interpolation from the key poses. By applying the waist trajectory adjustment, both the key poses and the motion interpolated from them are adjusted to be the dynamically balanced ones. The adjustment only modifies the horizontal waist position of the key poses and the interpolated ones, and it preserves the poses of the initial and final key poses.

It is important that the adjustment is automatically and

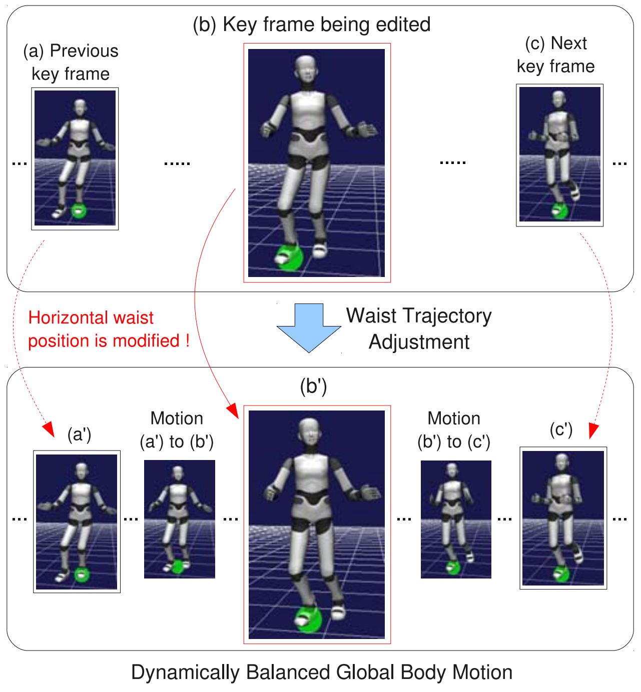
(b) Key frame being edited
(a) Previous key frame
(c) Next key frame
Horizontal waist position is modified !
Waist Trajectory Adjustment
(b')
(a')
Motion (a') to (b')
Motion (b') to (c')
(c')
Dynamically Balanced Global Body Motion

Fig. 4. Waist trajectory adjustment, which is automatically processed immediately after every time key poses are modified. As a result, horizontal waist positions of key poses are slightly modified and a dynamically balanced motion is obtained.

immediately done every time a user has finished an edit operation which affects the dynamic balance. Such edit operations include the creation, modification, elimination and time shift of key poses. This is a very characteristic feature compared with the existing interfaces and the feature accelerates the efficiency of the motion editing. In the case of Fig.4, just after a user has finished an operation of creating or modifying key pose (b), the user obtains the pose (b'), which is guided by the interface and is actually registered to the key pose. In this time, the precedent and subsequent key poses are also slightly modified depending on the original modification to key pose (b).

Note that "dynamically balanced motion" in the above description concretely means that the trajectory of Zero Moment Point (ZMP) calculated from body motion trajectories is always inside the foot support area. Actually the amount of the adjustment is determined from the desired ZMP trajectory the system automatically generates from foot states of key poses. In Fig.3 and Fig.4, the centers of spheres around feet correspond to desired ZMPs. Usually a user does not have to consider ZMP (it can be hidden from the interface), but when a user want to control the horizontal waist position, desired ZMPs can be manually specified by setting the ZMP attribute to key poses. Especially for a period when a robot is standing with the both feet, the support area can be sufficiently large for control the waist position.

C. Floor Fitting

A user can basically set an arbitrary foot position and attitude. However, the position may not be appropriate for

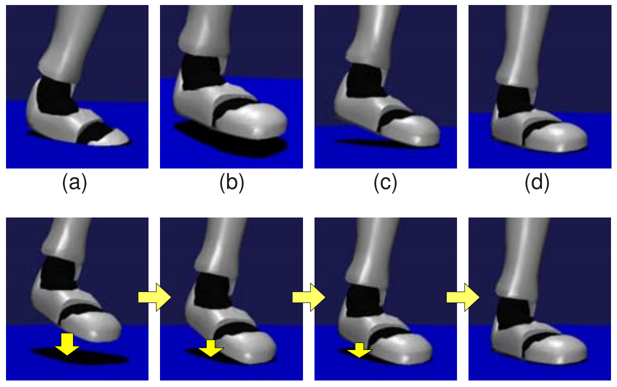

Fig. 5. Floor fitting.

making a stable motion. For example, in Fig.5, position (a) is fundamentally impossible because the foot is penetrating the floor. Positions (b) and (c) is not appropriate if the feet are assumed to be a support foot because (b) is floating on the floor and (c) does not make a support area, which is required for the condition of "dynamically balanced motion" as described in Section III-B. In summary, a foot must not penetrate the floor like (a) and a support foot must make a flat contact to the floor like (d). This condition is called the *foot contact condition*. Since results of manual pose editing easily breaks the condition, some assistance by the interface should be provided.

Floor fitting is a function that helps a user make a key pose that satisfies the foot contact condition. The function corresponds to item 1b in Section III-A. The floor fitting simply works as shown by the lower images of Fig.5. When a user moves a foot to the floor, the foot actually moves while preventing the penetration to the floor, and finally the movement stops when the sole completely fit to the floor. In this way, the function is integrated into the usual pose editing and it continuously works, so that it can be used intuitively and flexibly. The final position can be used for a support foot, and a halfway position can also be used for setting a swing foot position.

D. Foot Trajectory Adjustment

Even if the foot positions of all the key poses satisfy the foot contact condition, the trajectories interpolated from the key poses may include positions which do not satisfy the condition. In order to make the interpolated foot trajectories satisfy the condition, the interpolation must consider the following factors:

1) Blending of the joint-space interpolation and the Cartesian-space interpolation
2) Velocities at key poses

First, for a key pose where a foot touches the floor, the foot trajectory around the key pose should basically be done in the Cartesian-space because it is difficult for the joint-space interpolation to keep the condition of the position and attitude in the Cartesian-space. The joint-space interpolation generally produces the positions such as Fig.5 (a)-(c). However, the Cartesian-space interpolation is not necessarily the best method for the period where a foot does not touches the floor because the joint-space interpolation may be more reasonable because it does not have to worry about the singular points and the resulting trajectory is usually smoother for each joint actuator. Consequently the interpolations of the two spaces should be appropriately blended considering the foot state of the contact and user's intentions.

Secondly, the velocity of a foot should be zero at a key pose where the foot is assumed to be a support foot. Otherwise, the trajectory around the key pose does not keep the foot contact condition. On the other hand, for a swing foot, the velocity at a key pose should not necessarily be zero because it may be just a pass point of the nonstop movement. Imposing these velocity settings for all the key poses on a user is not practical. Hence the system must appropriately set velocities for the key poses where a user does not specify them.

In addition, the velocity of a foot just before the landing should be also considered. Because an actual robot cannot completely follow given motion trajectories, the velocity in the reference motion affects the smoothness of the actual landing. In order not to produce a big landing impact, foot trajectories should be generated so that each velocity of a foot just before the landing should be sufficiently reduced.

By considering the above factors, foot trajectories can be ones which are stably performed by an actual robot. The proposed interface automatically generates trajectories considering all the above factors just from the given key poses. This function is called *foot trajectory adjustment*, which corresponds to item 2b of Section III-A. As same as the waist trajectory adjustment, the adjustment works every time a user does an edit operation to key poses and the resulting motion can be immediately previewed.

E. Target and Limitations

As you can see in the above descriptions, our interface intends to generate motion data which are used as reference trajectories for controlling a robot. Thus the target robots of our interface are those who are driven by the position-based control and should have a ZMP-based feedback stabilizer [13].

In addition, although the interface is sufficiently flexible for creating variety of whole body motions, current implementation in fact has the following limitations about the editable motions:

1) The robot should always touch the floor with a valid support area
2) The motion should be performed on the flat and level floor
3) The motion should not include slips between the body and the floor

To be accurate, the motions that do not satisfy the above conditions can be edited on the interface, but the interface cannot guarantee that those motions can be as-is realized by the actual robot. In fact it is not practical for creating those motions in the current system.

TABLE I
ATTRIBUTES OF A KEY POSE

Overall Attributes

t	Time of the key pose
h	Maximum transition time to this key pose. (∞ by default).
S	A set containing the indices of the links that are processed as valid samples of the interpolation. (all the links by default).

Attributes of each valid link ($i \in S$)

θ_i	Joint angle of the i-th link
$\boldsymbol{p}_i, \boldsymbol{r}_i$	The position ($\boldsymbol{p}_i$) and rotation ($\boldsymbol{r}_i$) of the i-th link in the world coordinate. $\boldsymbol{r}_i$ is a 3d vector of the axis-angle representation. The attributes are only valid for the IK-links.
$is\_Cartesian_i$	Whether the Cartesian-space interpolation is applied to i-th link or not. This attribute is only valid for the IK-links. The value is always true for the waist link.
is_sp_i	Whether the interpolation treats the i-th link at t as a stationary point or not
$is\_touching_i$	Whether the i-th link is touching the floor or not. This attribute is only set by the system.

Attributes for the waist trajectory adjustment

$\boldsymbol{z}$	ZMP in the world coordinate
is_zmp_valid	Whether ZMP is specified or not
is_static	Whether the waist trajectory adjustment treats the key pose as the statically balanced one or not

It should be mentioned that the conditions other than those described above are not automatically handled in the current system. For example, self-collisions and the overruns of the angular velocity limits of joints are not automatically avoided. Instead the interface just notices the emergence of them and they must be removed by manual modifications. It is not necessarily better to automatically remove these faults because there are a number of modification variations for removing those faults and part of them would not be able to satisfy the user's original intention. Instead, the direct edit style of our interface would enable a user to efficiently do manual modifications for removing these faults.

IV. IMPLEMENTATION

A. Key Poses and Interpolation

A key pose has the attributes shown in Table I. In the key pose editing, basically t, θ_i, $\boldsymbol{p}_i$, $\boldsymbol{r}_i$ $(i \in S)$ are specified using the GUI by a user. These values determines the time and pose of a key pose. Just after a key pose is created or modified, each $is\_touching_i$ attribute of it is automatically set to true by the system if the corresponding foot is making a contact with the floor. The remaining attributes are specified by a user only when the user want to control the details. Otherwise the system automatically sets those attributes or default values are used.

The key pose has the concept of *IK-links*. An IK-link is a link whose position and attitude in the Cartesian space are registered and interpolated in addition to the joint angle. Basically the waist link and the foot links are IK-links, and the hand links can also be IK-links.

The interpolation of the whole body pose is processed as follows. First the positions and attitudes of the IK-links are

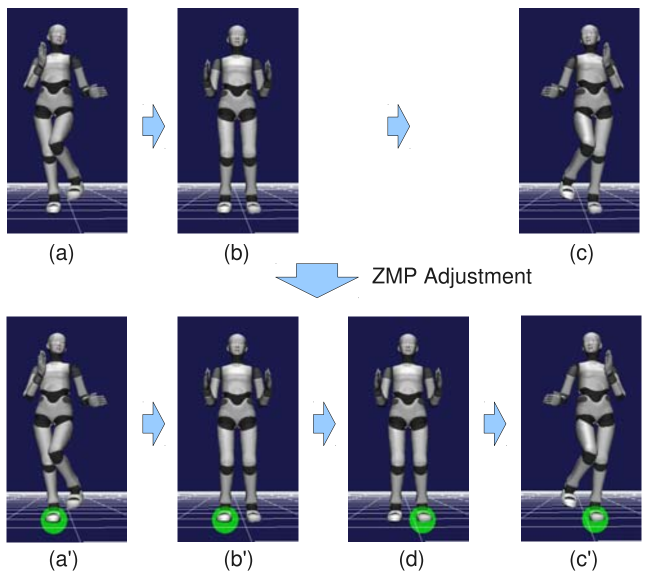

Fig. 6. Setting key ZMPs for the waist trajectory adjustment.

interpolated. For a joint between IK-Links, the joint angle is obtained by blending the normally interpolated value and the value calculated by the inverse kinematics (IK) between the IK-Links. The blending ratio is determined by a polynomial function for the key pose segment where the result at the key pose with $is\_Cartesian_i$ on becomes the value calculated by IK and the result at the key pose with $is\_Cartesian_i$ on becomes the normally interpolated value. For the other joints, the normally interpolated joint angle is employed. In our implementation, analytical IK solutions are used for IK between IK-Links.

Low-level interpolations are separately processed for each i-th element of joint angles, link positions and link attitudes. Each interpolation is basically done by the cubic spline. In addition to the first and last samples, the samples at key poses with is_sp_i on become the end points with zero-velocity.

B. Waist Trajectory Adjustment

First of all, the desired ZMP trajectory is required for the waist trajectory adjustment. The trajectory is interpolated from key ZMPs, each of which is specified by attribute $\boldsymbol{z}$ of a key pose with *is_zmp_valid* on. Necessary key ZMPs are automatically set by the system. By checking attribute $is\_touching_i$ of the foot links, the support state at a key pose can be found. From the state change sequence, the system can find the lacking key ZMPs which are needed for the stable ZMP transition and the system adds them. As shown in Fig.6 (a) to (c), there is a case where just adding key ZMPs at the existing key poses cannot achieve the stable ZMP transition. In this case, an auxiliary key pose for the swing foot with an appropriate key ZMP is implicitly inserted in the interpolation system as shown in Fig.6-(d).

After the desired ZMP trajectory is prepared, the dynamic balance adjustment is processed using the method proposed by Nishiwaki et al. [14]. The method calculates the horizontal CM trajectory that is consistent with the given ZMP trajectory by solving a system of tridiagonal matrix that is derived from the discrete ZMP equation. In fact, the modification of the CM trajectory is approximated by the modification of the waist trajectory and the method is iteratively applied to converge the solution. In our implementation, the modification of the waist trajectory is directly applied to the interpolation of the waist link. This implementation can avoid the generation of the modified trajectories for all the links, so that the adjustment process becomes more simple and efficient.

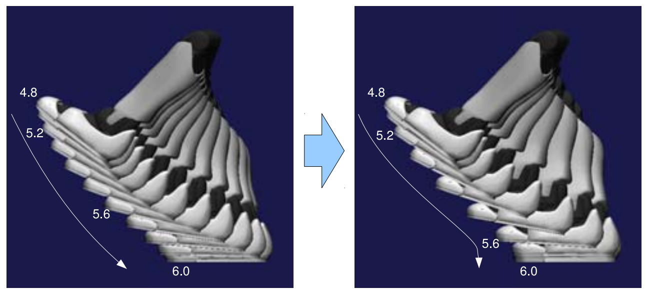

Fig. 7. Example of the smooth landing adjustment. The number attached to foot positions are times [s] of the positions.

As shown in Section V, the implementation of the adjustment process is sufficiently efficient for usual cases, but immediate adjustment will be difficult for a fairly long motion sequence if the adjustment is applied to the entire motion sequence. The attribute *is_static* can be used for avoiding this problem. If a key pose with *is_static* on is inserted, the tridiagonal matrix system for the balance adjustment is separated at that point. This allows the partial update and it can be processed in a shorter time. This can be a practical solution because long motion sequence generally includes statically balanced instants.

C. Foot Trajectory Adjustment

The foot trajectory adjustment is achieved by appropriately setting relevant attributes of key poses and inserting auxiliary key poses.

As for the attribute setting, the system has to make $is\_Cartesian_i$ and is_sp_i true for every key poses with $is\_touching_i$ on so that the floor contact condition can be kept in the interpolation.

For the landing impact relaxation, the system inserts auxiliary key poses for a swing foot at the timings just before the swing foot landings. Figure 7 shows an example of applying this process. The left image shows a normally interpolated trajectory of a swing foot and the right image shows the trajectory with an auxiliary key pose. The left trajectory spends more time to trajectory just before the landing to reduce the velocity just before the landing. In order to reduce both the vertical and horizontal impact, the angle of approach is also changed so that the horizontal impact as well as the vertical impact can be reduced.

Note that changing attributes and inserting auxiliary key poses are implicitly processed in the system, so that the foot

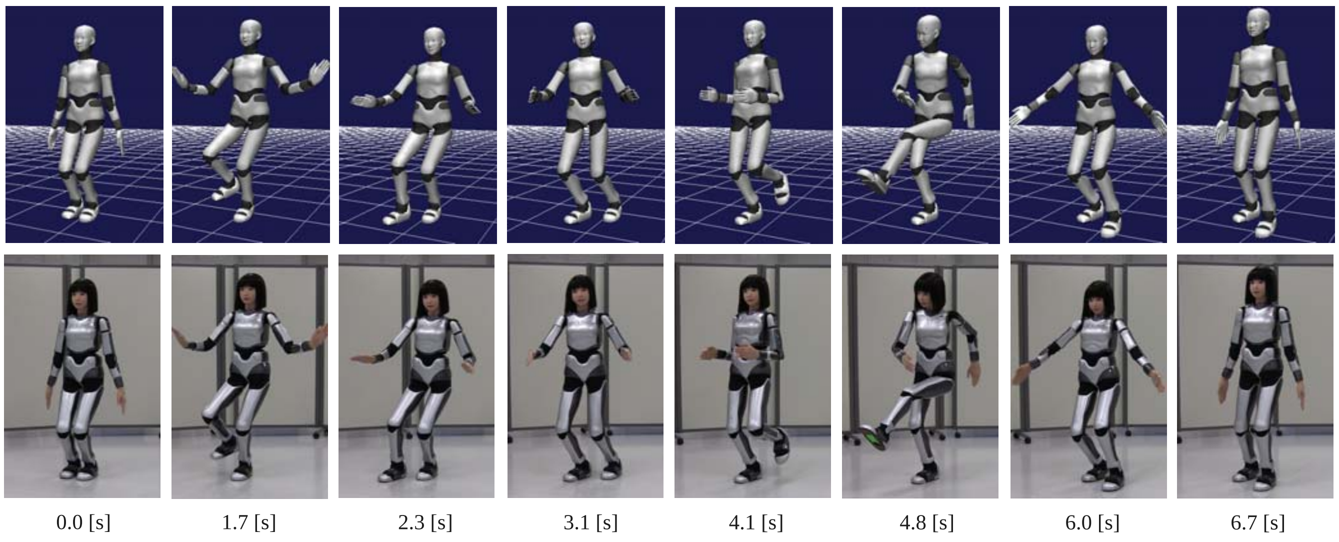

Fig. 8. Example motion for testing the proposed interface. Actual HRP-4C successfully performed the motion created with the interface.

trajectory adjustment does not affect the key pose editing by a user.

D. Implementation Platform

We implemented the proposed user interface as an application software. The application is built on our *"Extensible C++ Application Development Environment (Excade)"*. Excade is a cross-platform framework for developing an GUI application with a rich set of visualization facilities. It is written in C++ to obtain the maximum performance. It uses *Gtkmm* as a GUI toolkit and *OpenSceneGraph* as a library for rendering 3D Graphics. These are popular cross-platform, open-source libraries. The application also uses the library of *OpenHRP3* [15]. OpenHRP3 is an open-source dynamics simulator for robots and its core program for the model computations is available as a C++ library. By using the above framework and libraries, we have implemented the proposed interface as an practical application software that supports both Linux and Windows.

V. Experiments

In this section, we show an example of motion editing and discuss the usefulness of the proposed interface.

We used HRP-4C [1] as a robot model for testing the interface. As shown in Fig.1, HRP-4C is a biped humanoid robot with realistic human-like appearance. Its height is 1.58[m] and weight is 43[kg]. Its total DOF is 43: 8 in the face, 3 in the neck, 8×2 in the arms (with 2-DOF hands), 3 in the torso, and 6×2 in the legs. HRP-4C has a potential to perform various expressive motions using these joints and human-like appearance. It should be noted that the sole size of HRP-4C is close to that of humans, which means the relative sole size is smallest among the existing humanoid robots. To control the robot, we used a ZMP-based feedback stabilizer developed by Kajita et al. [13]. The frame rate of the trajectory data given to the controller is 200 [fps], which can be directly output from the application.

As sample motions for testing the interface, we created two patterns of motions. One is a motion like exercise, which is partly shown in Fig.1 and Fig.6. We were able to easily create this motion just by putting poses (a), (b) and (c) in Fig.6 and the created motion was successfully performed by actual HRP-4C. The left image of Fig.1 shows a cut of that performance.

The other motion is shown in Fig.3 and Fig.8. In this motion, the robot does two steps while making characteristic poses using the whole body. This kind of complex motion is difficult to be generated by pattern generation programs such as walking pattern generators. All we had to do in editing this motion was just giving these eight key poses. We were able to smoothly edit the key poses using the interface described in Section III-A. The key ZMPs shown in Fig.3 were automatically determined by the system. As shown in the lower images of Fig.8, this motion was also successfully performed by HRP-4C.

To check the validity of the proposed interface from the viewpoint of making stable motions, we tested the dynamics simulations of the sample motion data which are generated with the trajectory adjustments disabled. As shown in Fig. 9, the robot falls down in a short time for those motions.

The simulations were done by using the dynamics engine of OpenHRP3 [16][15], which has been used for a number of simulations for testing the motions of the robots including HRP-2 and HRP-4C. The dynamics simulation function is also integrated into the developed application so that a user can easily test the dynamics of edited motions on the application.

We also measured a time consumed for doing the automatic adjustment process just after a modification of key poses by a user. The conditions of the measurement were as follows: CPU was Intel Core i7 965 (3.2GHz). OS is Ubuntu Linux 9.10 and the program was compiled by GCC 4.4.1. The frame rate of the motion data used in editing was 30 [fps], which is sufficient for the use of motion preview. The

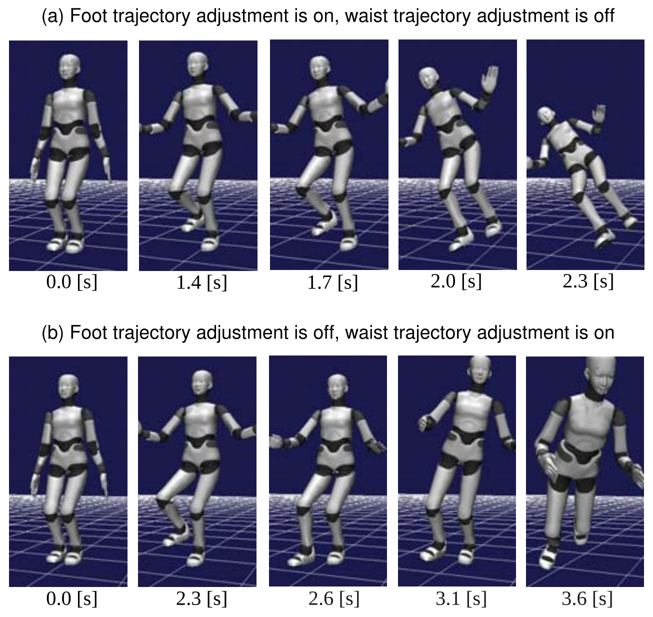
(a) Foot trajectory adjustment is on, waist trajectory adjustment is off
0.0 [s]
1.4 [s]
1.7 [s]
2.0 [s]
2.3 [s]
(b) Foot trajectory adjustment is off, waist trajectory adjustment is on
0.0 [s]
2.3 [s]
2.6 [s]
3.1 [s]
3.6 [s]

Fig. 9. The results of the dynamics simulations where the robot performed the sample motion data which are created without either the foot trajectory adjustment or the waist trajectory adjustment. In the both cases, the robot fell down in a short time.

number of the iterations of the low-level convergent process described in Section IV-B is three, with which the result sufficiently converges. In the above conditions, the motion of Fig.8 was used for this test and the average consumed time of 10 trials was 36.0 [ms]. This result indicates that the computation time of the adjustment process is sufficiently fast to be the on-line process. This will hold if the total time length of a edited motion is dozens of times longer. Even if the total time length is longer than that, inserting key poses with *is_static* attribute on can keep the processing time still practical as described in Section IV-B.

VI. Summary

In this paper, we have proposed a novel user interface for creating whole body motions of biped humanoid robots. The interface enables a user to create the motions just by giving key poses as if the user were creating a CG character animation. The result of such a simple operation is guaranteed to be a feasible motion that satisfies the kinematic and dynamic consistency between the robot and the floor. The design and implementation of the proposed interface are described, and the application software that provides the interface has actually been developed. The motion edit using the interface has been tested for biped humanoid robot HRP-4C. The result shows the usefulness and practicality of the proposed interface.

We are planning to publicly release the developed application, which also provides the functions for creating facial motions of humanoid robot [17]. The application will enable researchers, developers and users of humanoid robots to create various motions for various purposes including entertainment use and human-robot interaction.

References

[1] K. Kaneko, F. Kanehiro, M. Morisawa, K. Miura, S. Nakaoka, and S. Kajita, "Cybernetic Human HRP-4C," in *IEEE-RAS International Conference on Humanoid Robots*, Paris, France, December 2009.

[2] M. Riley, A. Ude, and C. G. Atkeson, "Methods for motion generation and interaction with a humanoid robot: Case studies of dancing and catching," in *Proceedings of AAAI and CMU Workshop on Interactive Robotics and Entertainment 2000*, Pittsburgh, Pennsylvania, April 2000, pp. 35–42.

[3] N. S. Pollard, J. K.Hodgins, M. J. Riley, and C. G. Atkeson, "Adapting human motion for the control of a humanoid robot," in *Proceedings of the 2002 IEEE International Conference on Robotics and Automation*, Washington DC, April 2002, pp. 1390–1397.

[4] K. Yamane and Y. Nakamura, "Dynamics filter - concept and implementation of online motion generator for human figures," *IEEE Transaction on Robotics and Automation*, vol. 19, no. 3, pp. 421–432, 2003.

[5] S. Nakaoka, A. Nakazawa, F. Kanehiro, K. Kaneko, M. Morisawa, and K. Ikeuchi, "Task model of lower body motion for a biped humanoid robot to imitate human dances," in *Proceedings of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Edmonton, Canada, August 2005, pp. 2769–2774.

[6] K. Kaneko, F. Kanehiro, S. Kajita, H. Hirukawa, T. Kawasaki, M. Hirata, K. Akachi, and T. Isozumi, "Humanoid robot HRP-2," in *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, New Orleans, LA, April 2004, pp. 1083–1090.

[7] S. Nakaoka, A. Nakazawa, F. Kanehiro, K. Kaneko, M. Morisawa, H. Hirukawa, and K. Ikeuchi, "Learning from observation paradigm: Leg task models for enabling a biped humanoid robot to imitate human dances," *International Journal of Robotics Research*, vol. 26, no. 8, pp. 829–844, August 2007.

[8] G. Pierris and M. G. Lagoudakis, "An interactive tool for designing complex robot motion patterns," in *Proceedings of The 2009 IEEE International Conference on Robotics and Automation*, Kobe, Japan, May 2009, pp. 4013–4018.

[9] Y. Kuroki, B. Blank, T. Mikami, P. Mayeux, A. Miyamoto, R. Playter, K. Nagasaka, M. Raibert, M. Nagano, and J. Yamaguchi, "Motion creating system for a small biped entertainment robot," in *Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Las Vegas, Nevada, October 2003, pp. 1394–1399.

[10] Y. Kuroki, M. Fujita, T. Ishida, K. Nagasaka, and J. Yamaguchi, "A small biped entertainment robot exploring attractive applications," in *Proceedings of the 2003 IEEE International Conference on Robotics and Automation*, Taipei, Taiwan, September 2003, pp. 471–476.

[11] K. Yamane and Y. Nakamura, "Natural motion animation through constraining and deconstraining at will," *IEEE Transaction on Visualization and Computer Graphics*, vol. 9, no. 3, pp. 352–360, 2003.

[12] T. Sugihara, "Solvability-unconcerned inverse kinematics based on Levenberg-Marquardt method with robust damping," in *Proceedings of the 9th IEEE-RAS International Conference on Humanoid Robots*, Paris, France, December 2009, pp. 555–560.

[13] S. Kajita, M. Morisawa, K. Miura, S. Nakaoka, K. Harada, K. Kaneko, F. Kanehiro, and K. Yokoi, "Biped walking stabilization based on linear inverted pendulum tracking," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Taipei, Taiwan, October 2010.

[14] K. Nishiwaki, T. Sugihara, S. Kagami, M. Inaba, and H. Inoue, "Online mixture and connection of basic motions for humanoid walking control by footprint specification," in *Proceedings of the 2001 IEEE International Conference on Robotics and Automation*, Seoul, Korea, May 2001, pp. 4110–4115.

[15] OpenHRP3 official site. [Online]. Available: http://www.openrtp.jp/openhrp3/en/

[16] S. Nakaoka, S. Hattori, F. Kanehiro, S. Kajita, and H. Hirukawa, "Constraint-based dynamics simulator for humanoid robots with shock absorbing mechanisms," in *Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Diego, CA, USA, October 2007, pp. 3641–3647.

[17] S. Nakaoka, F. Kanehiro, K. Miura, M. Morisawa, K. Fujiwara, K. Kaneko, S. Kajita, and H. Hirukawa, "Creating facial motions of Cybernetic Human HRP-4C," in *IEEE-RAS International Conference on Humanoid Robots*, Paris, France, December 2009.