Humanoid Dance Simulation Using Hybrid Model Predictive Control

Yuichi Tazaki†

***Abstract*— This paper proposes a method that realizes dynamic dancing motion of humanoid robots based on hybrid model predictive control. The proposed control method runs two types of model predictive controllers with different fidelity and time scale in parallel; one performs long-horizon prediction by making use of a closed-form solution of the centroidal dynamics, and the other performs short-horizon prediction based on the whole-body dynamics. A reference key-pose sequence of more than 100 key frames including stepping and fast upper-body movement was edited using Choreonoid and input to the controller. In closed-loop simulation of a torque-controlled 32-DoF humanoid robot, the controller was able to track the reference sequence by attenuating large angular momentum.**

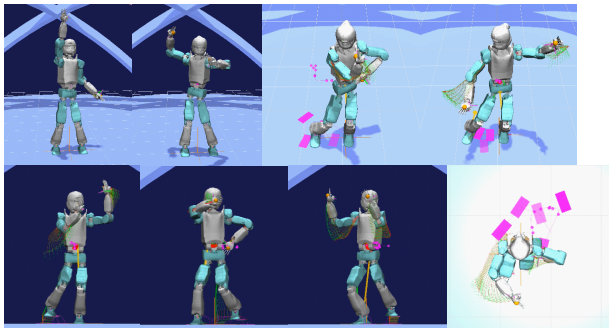


Fig. 1. Screenshots from dancing simulation

I. Introduction

Humanoid robots are built to imitate the appearance and physical ability of human being. This characteristic is not only useful for replacing human labor but also for replicating the artistic skills of human experts with robots. Dance is an example of such artistic skills that involve dynamic and large movement of the entire body. Recent advances in the performance of robotic hardware paved a way for realizing acrobatic dances with humanoid robots. On the other hand, establishing a design methodology of controllers that can realize stable dancing motion on a humanoid robot, even in simulation, is still a challenging and interesting topic to explore.

Various existing studies focused on motion retargeting, which is to convert reference motion provided by motion capture or by manual choreography to kinematically and dynamically feasible motion that can be executed by a robot. In previous studies, various retargeting techniques have been proposed. These include: balance compensation ([1, 2, 3, 4]), consideration of scale difference and self-collision ([5, 6]), and retargeting of human motion to a non-humanoid robot ([7]). Existing balance compensation techniques used for retargeting, however, are based on a simple strategy that is to satisfy the ZMP stability criterion by shifting the center-of-mass (CoM) horizontally. This technique has a number of limitations. First, certain margin from kinematic singularity must be ensured to make room for horizontal CoM adjustment (e.g., the knee joints must be bent all the time). Second, the method becomes infeasible when the downward CoM acceleration exceeds 1G. Third, modification of angular momentum is not considered.

More recently, whole-body model predictive control (MPC) enabled realization of wider variety of motion including singular postures, vertical CoM movement, and angular momentum variation. However, application of MPC has been mostly limited to loco-manipulation tasks, and its potential in realizing acrobatic dance has been explored little.

This paper proposes a motion retargeting and tracking technique for dancing motion including stepping and large angular momentum variation. The proposed method consists of two types of model predictive controllers executed in parallel. Centroidal (CD-)MPC makes long look-ahead prediction by making use of a closed-form solution of the centroidal dynamics, and generates desired motion that reduces angular momentum variation by modifying the base link angular velocity. Whole-body (WB-)MPC performs shorter-horizon prediction that tracks the output of CD-MPC. The developed technique is applied to a manually edited key pose sequence based on a dance performance of a CG animation character. In physics-based simulation, a closed-loop system consisting of the proposed controller and a 32-DoF torque-controlled humanoid robot model was constructed, and dance motion was successfully realized as shown in screenshot images (Fig. 1).

The rest of this paper is organized as follows. In Section II, software interface for dance motion editing and a robot model used in this study are described. In Section III, the centroidal dynamics of rigid-body systems is reviewed from the viewpoint of angular momentum compensation. In Section IV, a control system architecture of parallel MPC is presented and its detail is described. In Section V, simulation results are shown. Concluding remarks are given in Section VI.

II. Dance Motion Editing

Fig. 2 shows a screenshot of the GUI of Choreonoid [8] for dance motion editing. The GUI of Choreonoid is highly customizable. The one shown here is an example layout

†The author is with Faculty of Engineering, Department of Mechanical Engineering, University of Kobe, 1-1 Rokkodai, Nada-ku, Kobe, Japan. tazaki@mech.kobe-u.ac.jp

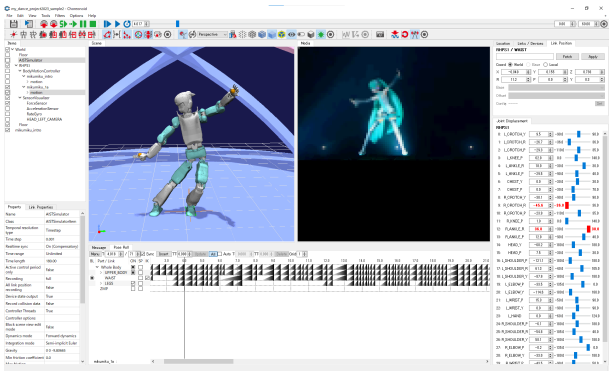
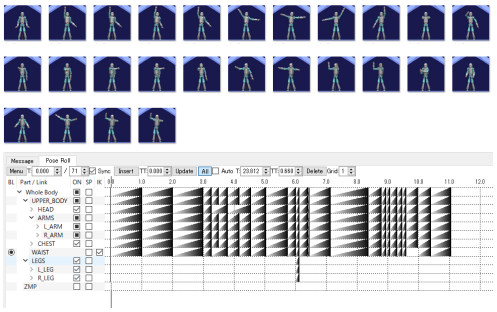


Fig. 2. GUI of Choreonoid



(a) Part 1. Total duration: 11s, number of key frames: 26.

(b) Part 2. Total duration: 24s, number of key frames: 81.

Fig. 3. Edited key-pose sequence

specialized for choreography, and consists of three main views: the scene view (upper left), the media view (upper right), and the pose-roll view (bottom). An original material of dance motion (e.g., CG animation, a video clip of human dance) is displayed in the media view, and the user edits key poses of the humanoid robot in the scene view. The pose-roll view enables insertion and deletion of key frames as well as adjustment of timing of existing key frames.

In this study, we used a video clip of the dance performance of a virtual character Hatsune Miku [1]. The original dance performance is more than 4 minutes long. Two parts (labeled Part 1 and Part 2) from the first 1 minute of the performance was used to create choreography of the robot as shown in Figs. 3(a)(b).

Some poses were difficult to reproduce precisely because of kinematic limitations. For example, large arm swinging around the sagittal axis was difficult because of shoulder singularity. Tilting of the hip around the sagittal axis was impossible because the robot model did not have the roll axis in the torso. Standing on tiptoe was also difficult because the robot model did not have toe joints. The original motion also included complex stepping such as turning of the support foot and jumping. In [9], classification of foot touch states that appear in dance was conducted, and dance motion including sliding of non-support foot was realized with a real humanoid robot. In this study, however, based on the consideration that our main focus is compensation of large angular momentum, these kinds of stepping were replaced by simple stepping in which each foot makes surface contact with the floor without sliding. We consider that reproducing complex stepping is a subject of future study.

The edited choreography data was exported in the YAML format to be loaded by the control program. Each key frame consists of the following information:

- Time stamp
- Base link pose
- Joint angles of upper-body and arm joints
- Left and right foot pose
- Contact state of left and right foot (flat contact or float)

III. Review of Centroidal Dynamics from the Viewpoint of Motion Retargeting

A. Centroidal Dynamics and Contact Stability

The centroidal dynamics equation [10] is expressed as follows.

$$\begin{aligned} \ddot{\boldsymbol{p}} &= \frac{1}{m} \sum_{l \in 1:n} \boldsymbol{f}_l \\ \dot{\boldsymbol{L}} &= \sum_{l} [\boldsymbol{\eta}_i + (\boldsymbol{p}_l - \boldsymbol{p}) \times \boldsymbol{f}_l] \end{aligned} \tag{1}$$

Here, m is the total mass, $\boldsymbol{p}$ is the center-of-mass (CoM), and $\boldsymbol{L}$ is the total angular momentum with respect to the CoM. Moreover, $\boldsymbol{f}_l$ and $\boldsymbol{\eta}_l$ are the linear and rotational component of the contact wrench applied to the l-th contact. In this study, we consider contact between either foot and the flat and level ground only, and we assume that each foot makes surface contact with the entire area of its sole.

The relationship between the total angular momentum and the whole-body configuration of the robot is expressed as follows.

$$\boldsymbol{L} = I(\boldsymbol{\theta})\boldsymbol{\omega} + \boldsymbol{q}\hat{\boldsymbol{L}}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}). \tag{2}$$

Here, $\boldsymbol{q}$ is a unit quaternion expressing the rotation of the base link and $\boldsymbol{\omega}$ is the angular velocity of the base link. Moreover I is the composite rigid-body inertia with respect to the CoM, which is dependent on the joint angle vector

[1] Hatsune Miku Magical Mirai 2017, https://www.youtube.com/watch?v=RU-OAZas1Ps

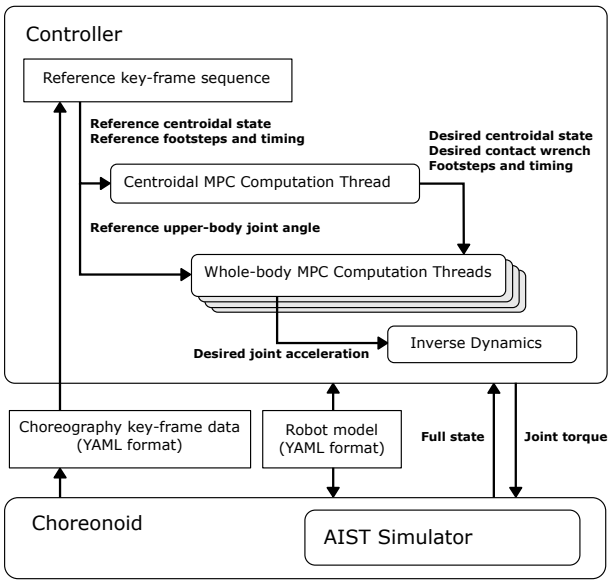


Fig. 4. Control system architecture

$\boldsymbol{\theta}$. Furthermore, $\hat{\boldsymbol{L}}$ is the *local* angular momentum of the system; namely, it expresses the total angular momentum expressed of the movement of the links expressed in the local coordinate frame of the base link. To distinguish between $\boldsymbol{L}$ and $\hat{\boldsymbol{L}}$, in this study we call $\boldsymbol{L}$ the *global* angular momentum. Equation (2) is essentially the same as Eqs.(7)-(9) of [11] and Eq.(13) of [10], but it expresses the decomposition of $\boldsymbol{L}$ more explicitly.

Most existing balance compensation techniques for motion retargeting are based on the linear inverted pendulum mode (LIPM), and they focused on modifying the CoM trajectory so that the resulting ZMP trajectory will stay inside the support region. In [6], a governing equation that includes not only the CoM but also the angular momentum was used. However, the angular momentum was not subject to modification. From (2) one can observe that the global angular momentum consists of the contribution of the base link angular velocity and that of the local angular momentum. We consider that the essence of dance motion is mostly embedded in whole-body movement relative to the base link and therefore modifying the angular velocity of the base link itself has little impact on the appearance of dance. On the other hand, reducing the magnitude of the global angular momentum and its time derivative is beneficial from the view point of balance control because movement with small angular momentum variation is generally easier to track. Based on this reasoning, we propose a model predictive controller that preserves the local angular momentum of the reference motion while reducing the global angular momentum by modifying the base link angular velocity.

IV. Control System Design

A. Overview

The overall control system architecture is shown in Fig. 4. The reference key pose sequence edited offline is loaded and stored in the controller's memory. A reference whole-body state of an arbitrary time instant is obtained by linear interpolation of key poses. Reference centroidal states (i.e., CoM position, velocity, and angular momentum) are calculated from the reference whole-body state. Reference footsteps and timing are directly obtained from key poses. The CD-MPC thread outputs optimized centroidal states together with modified footsteps and timing, and this is reflected to the cost function of WB-MPC. The desired joint angle and velocity of the upper-body and arms are directly provided from the interpolated key-pose sequence to WB-MPC without being filtered through CD-MPC. Desired joint torque is computed by means of inverse dynamics based on desired joint acceleration and contact wrench output by WB-MPC. Full state of the robot is retrieved from the simulator and input to WB-MPC to form a feedback control loop. On the other hand, the current state of the robot is not fed back to CD-MPC. In this sense, CD-MPC works as a feedforward dynamics filter.

B. Centroidal MPC

CD-MPC realizes long-horizon prediction by utilizing a closed-form solution of the centroidal dynamics (see [12] for detail). To begin with, the stiffness-based parametrization of contact force is defined as follows.

$$\boldsymbol{f}_l = m\lambda_l^2(\boldsymbol{p} - (\boldsymbol{p}_l + \boldsymbol{r}_l)) \tag{3}$$

Here, $\lambda_l \geq 0$ is the stiffness and it expresses the strength of the contact force. Moreover, $\boldsymbol{r}_l$ shifts the direction of the contact force; the contact force points from $\boldsymbol{p}_l$ towards $\boldsymbol{p}-\boldsymbol{r}_l$. By substituting (3) into (1) and applying the zero-order hold to λ_l, $\boldsymbol{r}_l$, and $\boldsymbol{\eta}_l$, the centroidal dynamics can be analytically integrated over a time interval between each consecutive key frames. In this manner, a discrete-time prediction model for $\boldsymbol{p}$, $\boldsymbol{v}$, and $\boldsymbol{L}$ can be obtained. A prediction model of $\boldsymbol{q}$, the base link rotation, is obtained by subdividing each time interval into smaller subintervals and applying Euler's method to the base link angular velocity given by $\boldsymbol{\omega} = I^{-1}(\boldsymbol{L} - \boldsymbol{q}\hat{\boldsymbol{L}})$. Here, the composite inertia I and the local angular momentum $\hat{\boldsymbol{L}}$ are calculated from interpolated key-pose sequence and provided to CD-MPC as references.

The state, control input, and output (used to define the cost function) of CD-MPC are defined as follows.

$$\boldsymbol{x}_k^{\mathrm{cd}} = \begin{bmatrix} \boldsymbol{p}_k \\ \boldsymbol{v}_k \\ \boldsymbol{q}_k \\ \boldsymbol{L}_k \\ t_k \\ \{\boldsymbol{p}_{l,k}\} \\ \{\boldsymbol{q}_{l,k}\} \end{bmatrix}, \quad \boldsymbol{u}_k^{\mathrm{cd}} = \begin{bmatrix} \tau_k \\ \{\boldsymbol{v}_{l,k}\} \\ \{\boldsymbol{\omega}_{l,k}\} \\ \{\lambda_{l,k}\} \\ \{\boldsymbol{r}_{l,k}\} \\ \{\hat{\boldsymbol{\eta}}_{l,k}\} \end{bmatrix}, \quad \boldsymbol{y}_k^{\mathrm{cd}} = \boldsymbol{x}_k^{\mathrm{cd}} \tag{4}$$

Note that each time step corresponds to a single time interval between consecutive key frames. The variables $\boldsymbol{p}_{l,k}$, $\boldsymbol{q}_{l,k}$, $\boldsymbol{v}_{l,k}$, and $\boldsymbol{\omega}_{l,k}$ denote the position, orientation, velocity, and angular velocity of the l-th foot, respectively. The variable t_k denotes the time instant of the beginning of the k-th key frame, while τ_k denotes its duration. CD-MPC formulated

above is capable of footstep and timing adaptation because it includes the footstep poses and timing as decision variables. However, the effectiveness of this feature in dance motion retargeting is not investigated in depth in the present study.

The cost function of CD-MPC is defined as follows:

$$
\begin{aligned}
J^{\mathrm{cd}} &= L^{\mathrm{cd}}_{N^{\mathrm{cd}}} + \sum_{k=0}^{N^{\mathrm{cd}}-1} L^{\mathrm{cd}}_{k}, \\
L^{\mathrm{cd}}_{k} &= \frac{1}{2}\|W^{\mathrm{cd,y}}(\boldsymbol{y}^{\mathrm{cd}}_{k} - \boldsymbol{y}^{\mathrm{cd,ref}}_{k})\|^2 \\
&\quad + \frac{1}{2}\|W^{\mathrm{cd,u}}(\boldsymbol{u}^{\mathrm{cd}}_{k} - \boldsymbol{u}^{\mathrm{cd,ref}}_{k})\|^2
\end{aligned}
\tag{5}
$$

where N^{cd} is the number of prediction steps (counted by number of key frames), $\boldsymbol{y}^{\mathrm{cd,ref}}_{k}$ and $\boldsymbol{u}^{\mathrm{cd,ref}}_{k}$ are desired values, and $W^{\mathrm{cd,x}}$ and $W^{\mathrm{cd,u}}$ are diagonal weight matrices. The desired value of the angular momentum is set as zero based on the discussion given in Section III. Desired timing (t_k and τ_k) are determined from the time stamps of key poses. The desired value of $\lambda_{l,k}$ is dependent on the contact state of the l-th contact during the k-th frame. If it is in contact, the desired stiffness is set as a value needed for statically supporting the weight of the robot, and otherwise it is set as zero. The desired value of $\boldsymbol{r}_{l,k}$ and $\boldsymbol{\eta}_{l,k}$ are both set as zero. For other variables, desired values are calculated from interpolated key poses. The setting of weights is summarized in Table I(b).

C. Whole-body MPC

Whole-body MPC is based on the centroidal dynamics + full kinematics formulation. By differentiating (2), we obtain

$$
\dot{\boldsymbol{L}} = I\dot{\boldsymbol{\omega}} + \dot{I}\boldsymbol{\omega} + \boldsymbol{q}\dot{\hat{\boldsymbol{L}}} + \boldsymbol{\omega} \times (\boldsymbol{q}\hat{\boldsymbol{L}}).
$$

Rearranging this gives

$$
\begin{aligned}
\dot{\boldsymbol{\omega}} = I^{-1}\Big(&- (\dot{I}\boldsymbol{\omega} + \boldsymbol{\omega} \times (\boldsymbol{q}\hat{\boldsymbol{L}}) + \boldsymbol{q}\dot{\hat{\boldsymbol{L}}}) \\
&+ \sum_{l \in 1:n} (\boldsymbol{\eta}_l + (\boldsymbol{p}_l - \boldsymbol{p}) \times \boldsymbol{f}_l) \Big)
\end{aligned}
\tag{6}
$$

In this manner, angular velocity of the base link can be explicitly included in the state variable.

The state, control input, and output of WB-MPC are defined as follows.

$$
\boldsymbol{x}^{\mathrm{wb}}_{\kappa} = \begin{bmatrix} \boldsymbol{p}_{\kappa} \\ \boldsymbol{v}_{\kappa} \\ \boldsymbol{q}_{\kappa} \\ \boldsymbol{\omega}_{\kappa} \\ \boldsymbol{\theta}_{\kappa} \\ \dot{\boldsymbol{\theta}}_{\kappa} \end{bmatrix}, \quad \boldsymbol{u}^{\mathrm{wb}}_{\kappa} = \begin{bmatrix} \ddot{\boldsymbol{\theta}}_{\kappa} \\ \{\boldsymbol{f}_{l,\kappa}\} \\ \{\boldsymbol{\eta}_{l,\kappa}\} \end{bmatrix}, \quad \boldsymbol{y}^{\mathrm{wb}}_{k} = \begin{bmatrix} \boldsymbol{x}^{\mathrm{wb}}_{\kappa} \\ \{\boldsymbol{p}_{l,\kappa}\} \\ \{\boldsymbol{q}_{l,\kappa}\} \\ \{\boldsymbol{v}_{l,\kappa}\} \\ \{\boldsymbol{\omega}_{l,\kappa}\} \end{bmatrix}
\tag{7}
$$

The time index of WB-MPC is denoted by κ to avoid confusion with CD-MPC. A discrete-time prediction model for WB-MPC is derived by applying the Euler method with a fixed time step Δt. The cost function of WB-MPC is defined as follows:

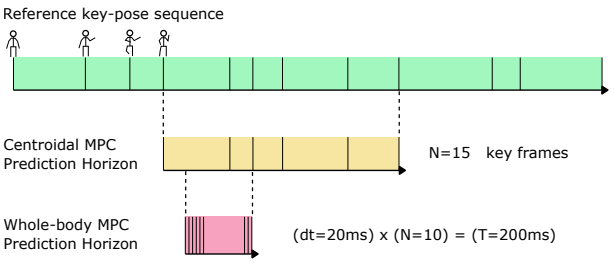


Fig. 5. Relationship of MPC threads

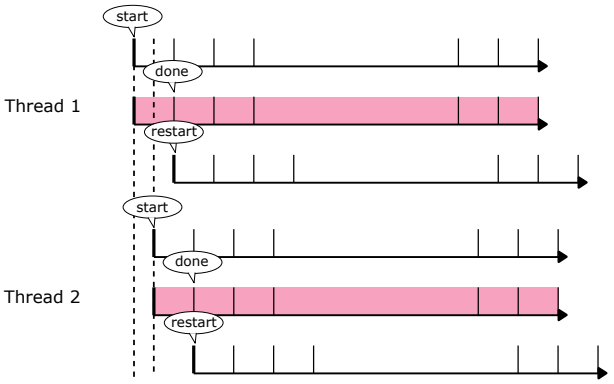


Fig. 6. Illustration of a group of multiple optimization threads triggered at shifted timing

$$
\begin{aligned}
J^{\mathrm{wb}} &= L^{\mathrm{wb}}_{N^{\mathrm{wb}}} + \sum_{\kappa=0}^{N^{\mathrm{wb}}-1} L^{\mathrm{wb}}_{\kappa}, \\
L^{\mathrm{wb}}_{\kappa} &= \frac{1}{2}\|W^{\mathrm{wb,y}}(\boldsymbol{y}^{\mathrm{wb}}_{\kappa} - \boldsymbol{y}^{\mathrm{wb,ref}}_{\kappa})\|^2 \\
&\quad + \frac{1}{2}\|W^{\mathrm{wb,u}}(\boldsymbol{u}^{\mathrm{wb}}_{\kappa} - \boldsymbol{u}^{\mathrm{wb,ref}}_{\kappa})\|^2
\end{aligned}
\tag{8}
$$

where N^{wb} is the number of prediction steps, $\boldsymbol{y}^{\mathrm{wb,ref}}_{k}$ and $\boldsymbol{u}^{\mathrm{wb,ref}}_{k}$ are desired values, and $W^{\mathrm{wb,y}}$ and $W^{\mathrm{wb,u}}$ are diagonal weight matrices. The desired values of the centroidal states ($\boldsymbol{p}_{\kappa}$, $\boldsymbol{v}_{\kappa}$, $\boldsymbol{q}_{\kappa}$, and $\boldsymbol{\omega}_{\kappa}$) are obtained by substituting the output of CD-MPC into the closed-form solution of the centroidal dynamics. The desired values of the foot states ($\{\boldsymbol{p}_{l,\kappa}\}$, $\{\boldsymbol{q}_{l,\kappa}\}$, $\{\boldsymbol{v}_{l,\kappa}\}$, and $\{\boldsymbol{\omega}_{l,\kappa}\}$) are obtained by interpolating the footsteps included in the output of CD-MPC using cycloid curves. Desired contact wrenches are calculated by substituting the output of CD-MPC to (3). Desired joint angle and velocity are directly taken from interpolated key poses, and desired joint acceleration is set as zero. The setting of weights is summarized in Table I(c).

D. Parallel Execution of MPC Threads

Integrating multiple predictive controllers based on models with different fidelity and time scale is a major technical challenge. In [13], switching of full and simple models within the prediction horizon was considered, but in this case the two models shared the same time scale. In this study, we take a strategy in which multiple MPC threads are

TABLE I
PARAMETER SETTING

	CD-MPC	WB-MPC
prediction steps	15	10
time step	1 key frame	20ms
update period	1 key frame	4ms
number of iterations	20	2
number of threads	1	5
state dimension	25	76
input dimension	25	50

(a) MPC parameters

CoM position	10
CoM velocity	10
Base rotation	10
Angular momentum	10
Time	1
Foot position	10
Foot rotation	1
Duration	1
Foot velocity	1
Foot angular velocity	1
Foot stiffness	1
Foot CMP offset	100
Foot moment	100

(b) CD-MPC weights

CoM position	5
CoM velocity	5
Base rotation	5
Base angular velocity	5
Angular momentum	5
Foot position	5
Foot rotation	5
Foot velocity	5
Foot angular velocity	5
Foot force	200
Foot moment	200
Leg joint angle	0
Leg joint velocity	0
Other joint angle	10
Other joint velocity	10
Joint acceleration	100

(c) WB-MPC weights

executed in parallel with loose synchronization, as illustrated in Fig. 5. Optimization of the CD-MPC thread is triggered at every key frame switching, while optimization of each WB-MPC thread is triggered at a fixed time preriod. The most recent optimization result of CD-MPC is reflected to the cost function of WB-MPC.

Another point of consideration is computation delay. Each optimization cycle of WB-MPC takes more than 10ms, which is much longer than a typical control period of a humanoid robot. To improve the update rate, we propose a simple technique which runs multiple computation threads with shifted timing, as illustrated in Fig. 6. As illustrated in the figure, n_{p} threads are run in shifted timing. Consider that the optimization computation of the i-th thread is triggered at time t. This optimization will be completed by $t+\Delta t_{\mathrm{delay}}$. The $(i+1)$-th thread will be triggered at $t+\Delta t_{\mathrm{update}}$, where $\Delta t_{\mathrm{update}} = \Delta t_{\mathrm{delay}}/n_{\mathrm{p}}$. In this manner, the update period is shortened to $\Delta t_{\mathrm{update}}$ while the latency is still determined by $\Delta t_{\mathrm{delay}}$.

E. Parameter Setting

The parameter setting of MPC is summarized in Tables I(a)-(c). Decision variables are normalized based on their physical dimensions so that their values range within the same order of magnitude. The weights shown in the table are applied to those normalized variables. In this study, good combination of weights was searched by manual tuning. It was found important to assign greater weights to control input variables (i.e., joint acceleration and contact wrenches) to avoid oscillatory behavior. In WB-MPC, the joint angle and velocity weights of the leg joints were set as 0 because these joints were guided by desired foot pose and velocity.

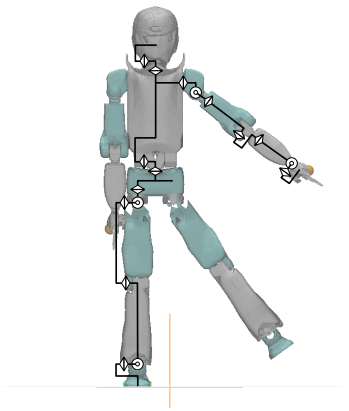


Fig. 7. Kinematic diagram of RHP-S1

V. DANCE SIMULATION RESULTS

A. Robot Model

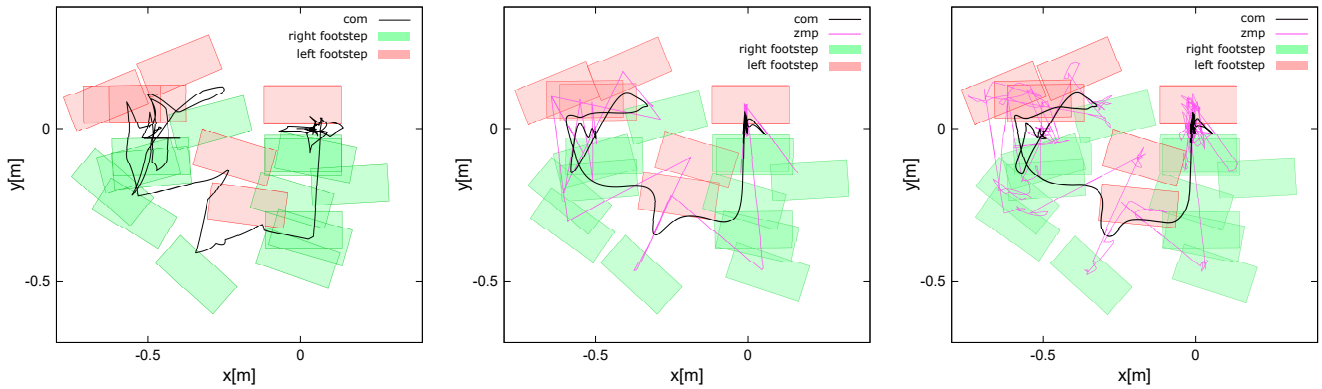
A model of RHP-S1 "Friends"[2] (Kawasaki Heavy Industry, [14]) was used in this study. The kinematic diagram of the robot is shown in Fig. 7. The model has 32 DoF (4 DoF in torso and neck, 8 DoF in each arm, 6 DoF in each leg). The kinematic and inertial parameters were matched to the real robot. Each arm and each leg of the robot weighs approximately 13% and 14% of the total mass, respectively. This means that this robot has particularly high mass proportion of arms compared to human body, whose arm mass is approximately $1/3$ of leg mass. This makes dancing motion involving rapid arm swinging especially challenging.

B. Simulation Setting

Choreonoid was used for both dance motion editing and dynamical simulation. Note that Body Motion Controller (a built-in balance compensator of Choreonoid) was not used, and raw key pose sequence was directly input to a user-defined controller that realizes MPC in the simulation loop of Choreonoid. The AIST simulator item was used for physics simulation. The time step of physics simulation was set as 0.25ms, whereas the control cycle was set as 1ms. A custom C++ trajectory optimization library named `dymp` [3] was developed and used to implement MPC. The main features of this library are analytical gradient computation of the centroidal and whole-body models, and implementation of differential dynamic programming (DDP) allowing infeasible starting points.

[2]Project Kaleido (in Japanese), `https://p-kaleido.com`

[3]dymp: DYnamics-based Motion Planner, `https://github.com/ytazz/dymp`



(a) Key Pose Interpolation (b) Centroidal MPC output (c) Wholebody MPC output

Fig. 8. Trajectories and footsteps output in different stages of computation

C. Results

Some screenshot images of simulation are shown in Fig. 1. See the attached video for better visualization of simulation. The knee joints of the robot had to be almost fully stretched throughout the tested dance motion. This made it difficult to apply conventional balance compensation techniques based on CoM shifting. Instead, the model-predictive controller mainly made use of tilting of the upper body and adjustment of arm trajectory to maintain the centroidal state inside the stabilizable region.

Figs. 8(a)-(c) show footsteps, CoM and ZMP trajectory generated in different stages. The CoM trajectory output by CD-MPC was much smoother than the one obtained by linear interpolation of key poses and it could be tracked by WB-MPC without much error. The ZMP output of WB-MPC also tracked that of CD-MPC although it showed certain degree of fluctuation for maintaining balance.

Figs. 9(a)-(c) show time series of the global and local angular momentum. Angular momentum obtained by linear interpolation of key poses showed large amplitude and discontinuity. In contrast, CD-MPC was able to output angular momentum with much smaller amplitude by canceling the local angular momentum with the base link angular velocity. The output of WB-MPC was also expected to show similar angular momentum profile but the result showed relatively greater amplitude. One possible reason of this error is that CD-MPC utilizes (2) based on the assumption that $\boldsymbol{\omega}$ and $\dot{\boldsymbol{\theta}}$ are independent. Because of kinematic constraint, the leg joint velocity is actually dependent on the base link angular velocity when the foot is in contact with the floor. Discrepancy between Fig. 9(b) and (c) could be reduced by taking this dependency into account. Nevertheless, the amplitude of the global angular momentum was smaller than the local angular momentum as we can observe in Fig. 9(c), which indicates that cancelation with the base link rotation was effective to a certain extent.

D. Computation Time

The average and maximum computation time of MPC threads are summarized in Table II. The computation time of CD-MPC was negligible considering its low update rate (one update per key frame). The computation time of WB-MPC could be made smaller than the specified recomputation period (20ms) by limiting the number of iterations to 2. The overall simulation was slower than real time, however, because dynamical simulation and MPC were run on the same computer. The reason why WB-MPC required much longer computation time per iteration than CD-MPC is because it has much greater state and input dimensions (see Table I(a)). The computation cost of whole-body MPC could be reduced further by exploiting its structure (e.g., sparsity of state equations and costs). We consider it as a subject of future study.

TABLE II
COMPUTATION TIME

	ave [ms]	max [ms]
CD-MPC (20 iterations)	12	14
WB-MPC (2 iterations)	14	16

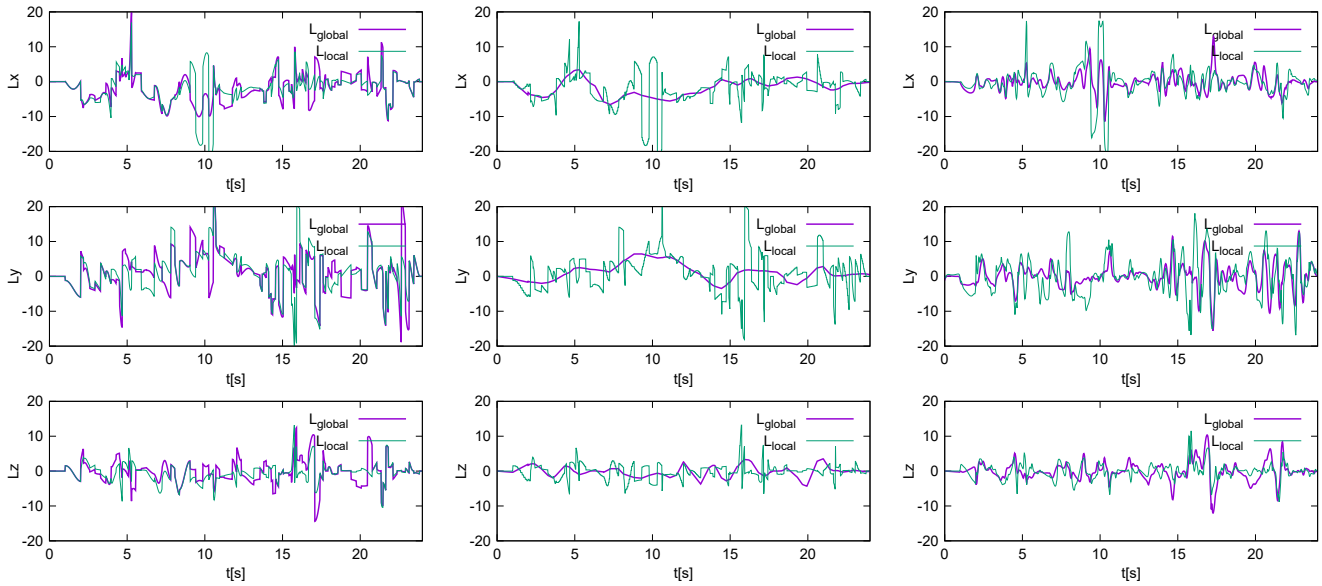
VI. Conclusion

This paper proposed a control method of dance motion based on a parallel model predictive control framework. Acrobatic dance motion was performed by a torque-controlled humanoid robot model in dynamical simulation. Future work includes realizing dance motion with a real humanoid robot. Major technical challenges toward sim2real would be to integrate whole-body state estimation into the control system loop and to respect joint velocity and torque limits.

References

[1] S. Nakaoka, A. Nakazawa, K. Yokoi, H. Hirukawa, and K. Ikeuchi. Generating whole body motions for a biped humanoid robot from captured human dances. In *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*, volume 3, 3905–3910 vol.3, 2003.

[2] S. Nakaoka, A. Nakazawa, K. Yokoi, and K. Ikeuchi. Leg motion primitives for a dancing humanoid robot. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, volume 1, 610–615 Vol.1, 2004.



(a) Key pose interpolation (b) Centroidal MPC output (c) Wholebody MPC output

Fig. 9. Angular momentum trajectories

[3] 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 *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3157–3162, 2005.

[4] K. Yamane and J. Hodgins. Simultaneous tracking and balancing of humanoid robots for imitating human motion capture data. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2510–2517, 2009.

[5] S. Nakaoka and T. Komura. Interaction mesh based motion adaptation for biped humanoid robots. In *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*, pages 625–631, 2012.

[6] T. Moulard, E. Yoshida, and S. Nakaoka. Optimization-based motion retargeting integrating spatial and dynamic constraints for humanoid. In *IEEE ISR 2013*, pages 1–6, 2013.

[7] K. Yamane and J. Hodgins. Control-aware mapping of human motion data with stepping for humanoid robots. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 726–733, 2010.

[8] S. Nakaoka. Choreonoid: extensible virtual robot environment built on an integrated gui framework. In *2012 IEEE/SICE International Symposium on System Integration (SII)*, pages 79–85, 2012.

[9] K. Kojima, S. Nozawa, K. Okada, and M. Inaba. Dance-like humanoid motion generation through foot touch states classification. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1788–1793, 2014.

[10] D. E. Orin, A. Goswami, and S. H. Lee. Centroidal dynamics of a humanoid robot. *Autonomous Robots*, 35(2-3):161–176, 2013.

[11] S.-H. Lee and A. Goswami. Reaction mass pendulum (rmp): an explicit model for centroidal angular momentum of humanoid robots. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 4667–4672, 2007.

[12] Y. Tazaki. Trajectory generation for legged robots based on a closed-form solution of centroidal dynamics. *IEEE Robotics and Automation Letters*, 9(11):9239–9246, 2024.

[13] H. Li, R. J. Frei, and P. M. Wensing. Model hierarchy predictive control of robotic systems. *IEEE Robotics and Automation Letters*, 6(2):3373–3380, 2021.

[14] Y. Kakiuchi, M. Kamon, N. Shimomura, S. Yukizaki, N. Takasugi, S. Nozawa, K. Okada, and M. Inaba. Development of life-sized humanoid robot platform with robustness for falling down, long time working and error occurrence. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 689–696, 2017.