INTECH
open science | open minds

OPEN ACCESS
ARTICLE

International Journal of Advanced Robotic Systems

Inverse Kinematic Control of Humanoids Under Joint Constraints

Regular Paper

Inhyeok Kim[1,*] and Jun-Ho Oh[1]

1 Division of Mechanical Engineering, The School of Mechanical, Aerospace & Systems Engineering, KAIST, Daejeon, South Korea
* Corresponding author E-mail: inhyeok@kaist.ac.kr

Received 22 Aug 2012; Accepted 26 Nov 2012

DOI: 10.5772/55247

Abstract We propose an inverse kinematic control framework for a position controlled humanoid robot with bounded joint range, velocity, and acceleration limits. The proposed framework comprises two components, an inverse kinematics algorithm and a damping controller. The proposed IKTC (Inverse Kinematics with Task Corrections) algorithm is based on the second order task-priority method in order to ensure the velocity-continuity of the solution. When the minimum norm solution exceeds the joint bounds, the problem is treated as a quadratic optimization problem with box constraints; an optimal task correction that lets the solution satisfy the constraints is found. In order to express the three kinds of joint constraints as a second order box constraint, a novel method is also proposed. The joint stiffness of a position controlled humanoid robot necessitates a damping controller to attenuate jolts caused by repeated contacts. We design a damping controller by using an inverted pendulum model with a compliant joint that takes into account the compliance around the foot. By using ZMP [20] measurement, the proposed damping controller is applicable not only in SSP (Single Support Phase) but also in DSP (Double Support Phase). The validity of the proposed methods is shown by imitating a captured whole-body human motion with a position controlled humanoid robot.

Keywords Inverse Kinematics, Joint Constraint, Damping Controller, Humanoid Robot

1. Introduction

The ultimate purpose of humanoid robots is to provide convenience for humans in human living environments. In order for humanoids to assist or substitute for humans, reliable mobility in various environments and dexterous manipulability are required. Therefore, humanoids are designed to have structural similarities to humans, i.e., two arms and two legs. With this structural characteristic a humanoid robot is able to perform multiple tasks simultaneously. However, this design introduces many degrees of freedom (DOF) and makes dynamic problems very complicated. A position-controlled humanoid, in this sense, allows simpler kinematic approaches.

The task-priority method [13, 18], which uses task Jacobian and its null space projection operators, is very suitable to handle the kinematic problems of a humanoid robot in multiple tasks. But simple inversion of the Jacobian does not take into account unilateral constraints such as the joint range, velocity, and acceleration limits. For this reason, several methods have introduced tasks to cope with unilateral constraints. The potential field is a task that pushes joints away from their limit positions [9]. An objective function that has a minimum value at the centre of the joint range is used in the gradient projection method [11]. However, the priorities of these tasks are always lower than those of the primary tasks, and thus satisfaction of the constraint is not guaranteed. In addition, it is difficult to select appropriate parameters that determine the strength of the repulsion or attraction. A method using the weighted least-norm solution was proposed in [3] to overcome the shortcomings of the gradient projection method; however, joint velocity and acceleration limits were not taken into account.

Clamping the joint velocity and acceleration to the limit values is the simplest way to satisfy the constraints but may result in unpredictable task errors. By using the infinity-norm, the lowest possible joint velocity solution was obtained in [5]. However, it is not guaranteed that the lowest velocity solution will satisfy the velocity limit. In [2, 1], the task-space trajectory was slowed down when the joint velocity or acceleration limits were encountered.

A typical joint position controller uses a high-gain PD control and thus discontinuous position commands should be avoided to protect joints from being damaged. For this reason, not only joint range but also joint velocity and acceleration limits should be considered simultaneously. A unified framework in which the three kinds of joint constraints are explicitly taken into account was proposed in [6]. The method has the same starting point as ours in shaping the joint velocity bounds. However, our inverse kinematics algorithm is defined at the acceleration level and is different from their method since they dealt with a redundant manipulator, whereas we are interested in a humanoid robot, which may not be redundant due to multiple tasks. From the aspect of multiple tasks, a general framework was proposed in [8] for a system with prioritized equality and/or inequality tasks. However, the constraints considered in this paper always have the highest priority and this allows a more efficient algorithm to be developed.

The high-gain position controllers of a humanoid robot introduce stiffness of joints and make the system vulnerable to impact forces. Even a small impact on the foot may cause a jolt to the system since a humanoid robot is not fixed on the ground. A damping controller can be used to reduce the jolt but normally high control bandwidth is required. For this reason, compliant materials are used, as shown in Figure 9, to absorb impacts on the contact surface; these materials function as a low-pass filter, lowering the required control bandwidth. Thus the damping controller should take into account the effect of these compliant materials.

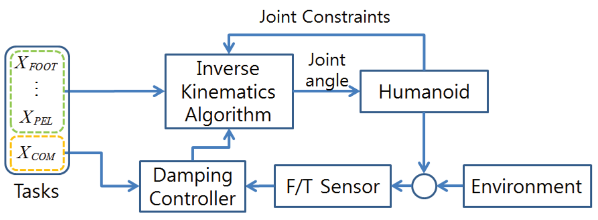


Figure 1. The proposed inverse kinematic control scheme.

In this paper, we propose an inverse kinematic control framework for a position controlled humanoid robot that has constraints of joint range, velocity, and acceleration limits. The proposed framework is comprised of two components, an inverse kinematics algorithm and a damping controller, as shown in Figure 1.

The proposed inverse kinematics algorithm, IKTC, is based on the second order task-priority method. When the minimum norm solution exceeds the joint bounds, an optimal task correction that would let the solution reside within the bounds is found by using quadratic programming. While the general method proposed in [8] has to solve a cascade of quadratic programs, IKTC solves a quadratic program once and thus can increase computational efficiency. In order to express the three kinds of constraints as a second order box constraint, a novel method is also proposed.

The proposed damping controller uses an inverted pendulum model with a compliant joint (IPCJ), as in [10]. While the damping controller proposed in [10] had the purpose of attenuation of the vibration in SSP (single support phase) only, our damping controller uses ZMP measurement and is applicable not only in SSP but also in DSP (double support phase).

The proposed methods are validated by imitating a captured human motion with a position controlled humanoid robot. Since there are kinematic and dynamic differences between humans and humanoids, the imitation of a captured human motion will be a good challenge to verify the proposed methods.

This paper is organized as follows. In Section 2, the background of the proposed methods is given. Section 3 presents the inverse kinematics algorithm and Section 4 presents the damping controller. The experiments are presented in Section 5 and conclusions are drawn in Section 6.

2. Background

2.1 Task-Priority based Inverse Kinematics

As mentioned earlier, a humanoid robot has a lot of DOF and thus is capable of performing multiple tasks simultaneously. However, a solution does not exist that is able to achieve all the tasks when there is a conflict. The least-squares solution can be used to minimize the second-norm of task errors when there is no exact solution, though this may not satisfy any of the tasks exactly.

The task-priority strategy is devised to handle tasks according to their order of priority [7,12,13]. By executing a task of lower priority in the null space of a higher priority task, conflicts between tasks are dealt with. A general framework of the methodology is proposed in [18] and is briefly reviewed in the following.

Consider a kinematic equation for the *i*-th task

$$x_i = f_i(q) \tag{1}$$

where $x_i \in \mathbb{R}^{m_i}$ is the *i*-th task vector, $q \in \mathbb{R}^n$ is the joint position vector, and *i* is the order of priority. The second-order differential kinematic equation for the *i*-th task can then be expressed as

$$J_i \ddot{q} = h_i \tag{2}$$

where $J_i = \partial f_i / \partial q \in \mathbb{R}^{m_i \times n}$ is the Jacobian matrix of the *i*-th task and h_i is defined as

$$h_i \triangleq \ddot{x}_i - \dot{J}_i \dot{q} \,. \tag{3}$$

The null space projection operator of the *i*-th task N_i is given by

$$N_i = I - J_i^+ J_i \tag{4}$$

where the superscript + denotes the pseudo-inverse of a matrix and I is an $n \times n$ identity matrix.

If we let the (*i*-1)th task have priority over the *i*-th task, and assume that the sum of all tasks' dimensions does not exceed the system's DOF, i.e., $\sum_i m_i \le n$, then the joint acceleration that achieves higher *i* tasks (from the first to the *i*-th task) is given by

$$\ddot{q}_i = \ddot{q}_{i-1} + \bar{J}_i^+ \left(h_i - J_i \ddot{q}_{i-1} \right), \quad \left(\ddot{q}_1 = J_1^+ h_1 \right) \tag{5}$$

with

$$\bar{J}_i = J_i N_{i-1}^A \,, \tag{6}$$

$$N_i^A = N_{i-1}^A - \bar{J}_i^+ \bar{J}_i \,, \quad \left(N_0^A = I \right). \tag{7}$$

As shown in (5), $\ddot{q}_i$ can be divided into two components, the solution of previous (*i*-1) tasks, $\ddot{q}_{i-1}$, and the solution that achieves the *i*-th task in the null space of the previous tasks by using the inverse of the task-consistent Jacobian $\bar{J}_i$ given in (6). The null space projection operator given in (7) projects any joint space vector onto the null space of previous tasks and therefore any components that would disturb the previous tasks are excluded by the task-consistent Jacobian.

2.2 Damped Least-Squares Inverse

When a task Jacobian loses its rank, the task cannot be executed and it is called a *kinematic singularity.* Another singularity called an *algorithmic singularity* occurs when a task-consistent Jacobian $\bar{J}_i$ is rank-deficient. In other words, the *i*-th task is incompatible with the higher (*i*-1) tasks. With these singularities, the inverse of $\bar{J}_i$ becomes ill-conditioned and the solution (5) tends to infinity.

To handle the singularities, we use the damped least-squares method proposed in [19]. The damped least-squares (DLS) inverse is given by

$$J^{\lambda+} = \left(J^T J + \lambda^2 I \right)^{-1} J^T \tag{8}$$

or, equivalently,

$$J^{\lambda+} = J^T \left(J J^T + \lambda^2 I \right)^{-1} \tag{9}$$

where $\lambda \in \mathbb{R}$ is the damping factor.

The DLS inverse (8) provides the solution of

$$\min_{\ddot{q}} \frac{1}{2} \left\| J\ddot{q} - h \right\|^2 + \lambda^2 \left\| \ddot{q} \right\|^2 . \tag{10}$$

Therefore, the solution norm is bounded to a certain value and singularity-robustness is obtained. However, the DLS inverse is not so sophisticated that the solution satisfies the given joint constraints. Thus we use the DLS inverse only for regularization of the solution in the neighbour of the singularity.

2.3 Closed-Loop Inverse Kinematics (CLIK)

In the task-priority strategy, lower priority tasks are sacrificed to achieve higher priority tasks when there is a conflict between the tasks. The error due to the sacrifice cannot be compensated for by the *open-loop* solution (5) even when there is no conflict afterwards. Besides, numerical integration of the differential inverse kinematic solution (5) would introduce long-term drift of the reconstructed profile from the desired profile. In order to deal with this problem, we adopt a second-order closed-loop inverse kinematics (CLIK) scheme [17].

For the generic *i*-th task, the CLIK scheme can be implemented as

$$\ddot{x}_i = \ddot{x}_{i,d} + k_v \left(\dot{x}_{i,d} - \dot{x}_i \right) + k_p \left(x_{i,d} - x_i \right) \tag{11}$$

where the subscript *d* denotes the desired value; k_v and k_p are the control gains for velocity and position error, respectively. If we let e_i denote the error between the desired value and the actual value of the *i*-th task, (11) can be rewritten as

$$\ddot{e}_i + k_v \dot{e}_i + k_p e_i = 0 \tag{12}$$

where $e_i = x_{i,d} - x_i$. Eq. (12) is the error dynamics of the *i*-th task; the convergence of the error can be shaped by selecting appropriate k_v and k_p.

2.4 Consideration of Linear Inequality Constraints Using Quadratic Programming

In the presence of joint limits, an inverse kinematics problem can be treated as a quadratic optimization problem with linear inequality constraints such as

$$\begin{aligned} &\min \frac{1}{2}\|J\ddot{q} - h\|^2 \\ &\text{subject to } b_l \le \ddot{q} \le b_u \end{aligned} \tag{13}$$

where $b_u \in \mathbb{R}^n$ and $b_l \in \mathbb{R}^n$ are the upper and lower bounds of the joint acceleration, respectively. The optimal solution of (13) can be found by using a typical quadratic programming (QP) algorithm [15].

The optimization algorithm proposed in [8] uses a sequence of QP in order to handle prioritized tasks. From the highest to the lowest priority task, QP is carried out during each task to solve an optimization problem given in the form of (13); the solution of the current priority task is imposed as a linear equality constraint on the next priority task. This algorithm is generous such that not only prioritized linear equality tasks but also prioritized linear inequality tasks can be taken into account. However, since here the constraints of the joint range, velocity, and acceleration limits always have the highest priority, a more efficient algorithm could possibly be developed.

3. Inverse Kinematics Algorithm

3.1 Joint Range, Velocity, and Acceleration Limits

Since we deal with inverse kinematics problems at the acceleration level, the joint range, velocity, and acceleration limits should be taken into account in the form of a second-order inequality constraint, as shown in (13).

Proposition 3.1

If the joint acceleration $\ddot{q}$ satisfies the inequality (14), the joint position, velocity, and acceleration are kept below their limits, respectively,

$$b_l \le \ddot{q} \le b_u \tag{14}$$

$$\begin{cases} b_u = \min\left(a_{max}, \dfrac{v_{max} - \dot{q}}{\Delta t}, \dfrac{\sqrt{2a_{max}(q_{max} - q)} - \dot{q}}{\Delta t} \right) \\ b_l = \max\left(-a_{max}, \dfrac{-v_{max} - \dot{q}}{\Delta t}, \dfrac{-\sqrt{2a_{max}(q - q_{min})} - \dot{q}}{\Delta t} \right) \end{cases}$$

where q_{min}, q_{max}, v_{max}, a_{max} are given values for the minimum and maximum joint position limits and the maximum velocity and acceleration limits, respectively.

Proof) Since the first and second terms in the min/max function are trivial constraints for the joint acceleration and velocity limits, respectively, the proof is omitted.

By the third term in the min/max function, the joint velocity $\dot{q}$ is constrained such that

$$-\sqrt{2a_{max}(q - q_{min})} \le \dot{q} \le \sqrt{2a_{max}(q_{max} - q)}, \tag{15}$$

and, as the result of this, the joint position is kept inside the position range $[q_{min}, q_{max}]$. As for the proof, we show that the joint position gets out of the range if (15) does not hold.

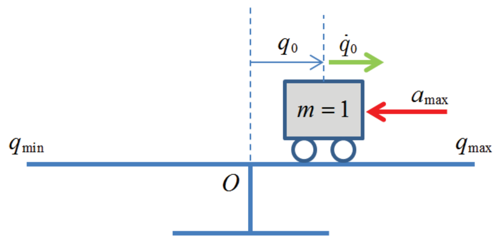


Figure 2. A cart-table model for the proof of Proposition 3.1.

Consider a cart of unit mass on a table, as shown in Figure 2. Let us assume that the initial velocity of the cart is

$$\dot{q}_0 = k\sqrt{2a_{max}(q_{max} - q_0)}, \quad (k > 1 : \text{const.}), \tag{16}$$

and that it decelerates maximally so as not to fall from the table. From the viewpoint of energy, the kinetic energy of the cart should be zero, at least on the edge of the table, in order not to fall. By applying the principle of the conservation of the energy, the cart has kinetic energy $KE_{@q\,max}$ at $q = q_{max}$ and $KE_{@q\,max}$ is given by

$$\begin{aligned} KE_{@q\,max} &= KE_{@q_0} + W_{dec} \\ &= \frac{1}{2}k^2 2a_{max}(q_{max} - q_0) - a_{max}(q_{max} - q_0) \\ &= (k^2 - 1)a_{max}(q_{max} - q_0) > 0 \quad (\because k > 1) \end{aligned} \tag{17}$$

where W_{dec} is the work done by an external force during deceleration. Since k was assumed to be $k > 1$, the kinetic energy $KE_{@q\,max}$ is greater than zero and therefore the cart falls from the table. In order for the cart not to fall from the table, k must be $|k|<1$ and the right inequality in (15) has been proved. The left inequality in (15) can be proved in the same way.

3.2 Inverse Kinematics with Task Correction (IKTC) Algorithm

In our inverse kinematics algorithm, some or all tasks are corrected such that the joint acceleration solution satisfies the inequality constraints (14) and the corrections are determined by solving a quadratic optimization problem.

Before proceeding, it should be mentioned that an over-constrained system is not considered in this paper. We are interested in a system that has DOF more than or equal to the task dimension. Thus,

$$n \ge \sum_{i=1}^{n_T} m_i \tag{18}$$

where n is the system DOF, m_i is the dimension of the *i*-th task, and n_T is the number of tasks.

If we let the correction of the *i*-th task be $\Delta h_i \in \mathbb{R}^{m_i}$, then, by using (5), the joint acceleration solution $\ddot{q}_i^*$ that satisfies the inequality constraints is obtained by

$$\begin{aligned} \ddot{q}_1^* &= J_1^+\left(h_1 + \Delta h_1\right) \\ \ddot{q}_2^* &= \ddot{q}_1^* + \bar{J}_2^+\left(h_2 + \Delta h_2 - J_1\ddot{q}_1^*\right) \\ &\vdots \\ \ddot{q}_i^* &= \ddot{q}_{i-1}^* + \bar{J}_i^+\left(h_i + \Delta h_i - J_{i-1}\ddot{q}_{i-1}^*\right) \end{aligned} . \tag{19}$$

For given n_T tasks, (19) can be rewritten as

$$\begin{aligned} \ddot{q}_1^* &= \ddot{q}_1 + J_1^+\Delta h_1 \\ \ddot{q}_2^* &= \ddot{q}_2 + J_1^+\Delta h_1 + \bar{J}_2^+\left(\Delta h_2 - J_2 J_1^+\Delta h_1\right) \\ &\vdots \\ \ddot{q}_{n_T}^* &= \ddot{q}_{n_T} + G\left[\Delta h_1^T \quad \Delta h_2^T \quad \cdots \quad \Delta h_{n_T}^T\right]^T \end{aligned} \tag{20}$$

with

$$\begin{gathered} G = \left[g_1 \quad g_2 \quad \cdots \quad g_{n_T}\right] \\ \left(g_i : n \times m_i \text{ matrix for } i = 1,\dots,n_T\right) \end{gathered} \tag{21}$$

where $\ddot{q}_1,\dots,\ddot{q}_{n_T}$ are joint accelerations obtained by (5) and $g_1,\dots,g_{n_T}$ are determined by collecting the coefficient matrices of Δh_i's. Therefore, with the proper manipulation of Δh_i's, G can be obtained easily using (5); this process is described in Algorithm 1.

Remark 1. If there is neither an algorithmic singularity nor a kinematic singularity, then G has full column rank since the dimension of G is $n \times \sum_{i=1}^{n_T} m_i$ and $n \ge \sum_{i=1}^{n_T} m_i$. Furthermore, using DLS inverse $\bar{J}^{\lambda+}$ instead of $\bar{J}^+$ in Algorithm 1, it can be guaranteed that G always has full rank even when there are singularities.

As shown in (20), $\ddot{q}_{n_T}^*$ is composed of two components, the joint acceleration $\ddot{q}_{n_T}$ from original tasks and the corrective task term by which the inequality constraints are satisfied. Substituting $\ddot{q}_{n_T}^*$ into the inequality constraints (14) gives

$$b_l - \ddot{q}_{n_T} \le G\left[\Delta h_1^T \quad \Delta h_2^T \quad \cdots \quad \Delta h_{n_T}^T\right]^T \le b_u - \ddot{q}_{n_T} \quad (22),$$

Algorithm 1 : Obtaining G in (20)

```
1: for $i = 1$ to $n_T$ do
2:    $\gamma \leftarrow [0 \ \cdots \ 0]^T \in \mathbb{R}^{m_i}$
3:    for $j = 1$ to $m_i$ do
4:       Substitute 1 for the j-th element of $\gamma$ and 0 for the
         other elements.
5:       $\psi \leftarrow \bar{J}_i^+\gamma$
6:       for $k = i + 1$ to $n_T$ do
7:          $\psi \leftarrow \left(I_n - \bar{J}_k^+ J_k\right)\psi$
8:       end for
9:       Substitute $\psi$ for the j-th column vector of $g_i$.
10:   end for
11:   Substitute $g_i$ for the i-th submatrix of $G$ in (21).
12: end for
```

Any task corrections satisfying (22) can be chosen as the solution for $\ddot{q}_{n_T}^*$. However, since the task corrections are equal to the task errors, and there are priorities given to tasks, the task corrections should be as small as possible and should avoid higher priority tasks as much as possible. For these purposes, let us consider an optimization problem:

$$\begin{gathered} \min \frac{1}{2}\left\|\Delta h_i^a\right\|^2 \\ \text{subject to } b_l - \ddot{q}_{n_T} \le g_i^a \Delta h_i^a \le b_u - \ddot{q}_{n_T} \end{gathered} \tag{23}$$

where Δh_i^a and g_i^a are defined by

$$\Delta h_i^a \triangleq \begin{bmatrix} \Delta h_i \\ \vdots \\ \Delta h_{n_T} \end{bmatrix}, \quad g_i^a \triangleq \left[g_i \quad \cdots \quad g_{n_T}\right]. \tag{24}$$

Solving (23) begins with $\Delta h_{n_T}^a$ and $g_{n_T}^a$ in order to avoid higher priority task errors. However, a solution may not exist because $g_{n_T}^a$ has rank of m_{n_T} and $m_{n_T} \le n$. The existence of the solution is checked using the simplex method [15]; if there does not exist a no solution exists satisfying the inequality constraints, (23) is extended to higher priority tasks by taking Δh_{i-1}^a and g_{i-1}^a. These steps are repeated until a solution is found: this process is summarized in Algorithm 2. If we assume that there is a solution when $i = k$, the solution Δh_k^a of (23) is then substituted into $\ddot{q}_{n_T}^*$ with g_k^a and $\ddot{q}_{n_T}^*$ becomes

$$\ddot{q}_{n_T}^* = \ddot{q}_{n_T} + g_k^a \Delta h_k^a . \tag{25}$$

Algorithm 2: Solving a quadratic optimization problem (23) taking into consideration the task priority.

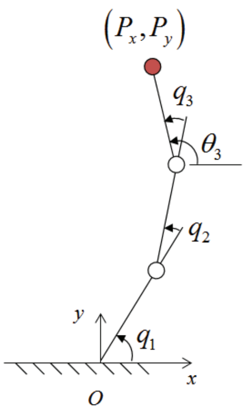
```
1: for $i = n_T$ to 1 do
2:     Check if there exists a solution of (23) by using the
       simplex method.
3:     if there is a solution then
4:         Solve (23) using QP to obtain $\Delta h_i^a$.
5:         break
6:     end if
7: end for
```

The joint acceleration solution $\ddot{q}_{n_T}^*$ given in (25) satisfies the inequality constraints by virtue of the correction tasks $\Delta h_k, \ldots, \Delta h_{n_T}$.

Before the solution has been obtained, the costly QP is carried out once, which increases the efficiency of the proposed algorithm remarkably better than does the use of the general method proposed in [8].

3.3 Simulation

A 3R planar manipulator, shown in Figure 3, is considered here to present the simulation. All links have the same unit length and the inverse kinematics is solved for the three prioritized tasks. Each joint has position range, velocity, and acceleration limits and, due to these constraints, task corrections happen in the IKTC algorithm. The simulation was conducted using two methods, the general framework (GF) proposed in [8] and the IKTC algorithm. In order to consider the joint constraints in GF, we used the inequality constraints (14) in GF.



Priority	Tasks
1	$P_y = P_y(t_0)$
2	$P_x = P_x(t)$
3	$\theta_3 = 90°$

Figure 3. 3R planar manipulator for the simulation in 3.3.

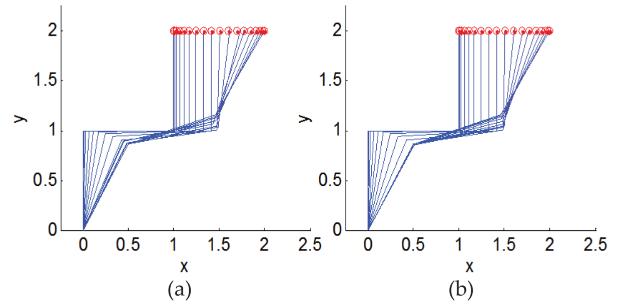


Figure 4. Tracking of the task trajectory (red circles) under joint limit constraints using (a) GF and (b) IKTC.

The joint range limits of the manipulator are q_{max} = $(180°, 120°, 120°)$, q_{min} = $(60°, -120°, -120°)$. The joint velocity and acceleration limits are $v_{max} = 180$ °/s , a_{max} = 1800 °/s^2 for all joints. The initial configuration is $q(t_0) = (90°, -90°, 90°)$ and thus $P_y(t_0) = 2$, $P_x(t_0) = 1$, $\theta_3(t_0) = 90°$. The desired P_x trajectory is a smooth sinusoidal curve connecting 1 and 2 for one second.

Figure 4 shows the simulation results generated by the GF and IKTC algorithms. Both sets of results show that the error of the lowest priority task increases when the three tasks become infeasible as the tip moves right and q_1 reaches its range limit. But, owing to the error, higher priority tasks can be achieved without violating the joint constraints. This can be verified in Figure 8, in which the task errors are shown.

The position, velocity, and acceleration trajectories of the three joints are shown in Figures 5, 6, and 7. None of these have violated their limits and thus the proposed inequality constraint in (14) is validated.

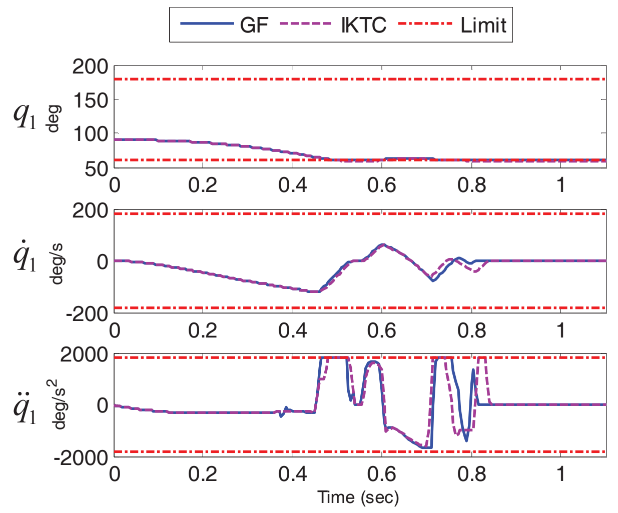


Figure 5. Trajectories of q_1, $\dot{q}_1$, and $\ddot{q}_1$ corresponding to the simulation results in Figure 4.

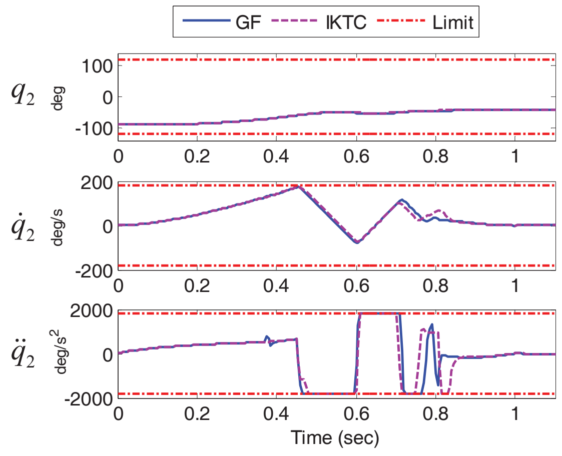


Figure 6. Trajectories of q_2, $\dot{q}_2$, and $\ddot{q}_2$ corresponding to the simulation results in Figure 4.

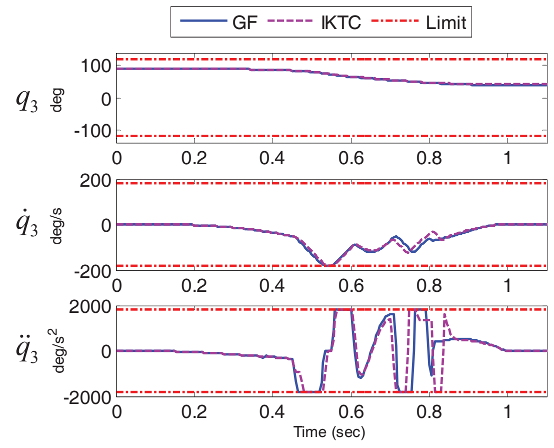


Figure 7. Trajectories of q_3, $\dot{q}_3$, and $\ddot{q}_3$ corresponding to the simulation results in Figure 4.

From Figure 8, it can be said that both the GF and IKTC algorithms generated almost the same results. In this simulation, however, GF carried out QP three times at every time step while IKTC carried out QP once at most. This leads to a situation in which the computation cost of IKTC is less than that of the GF algorithm.

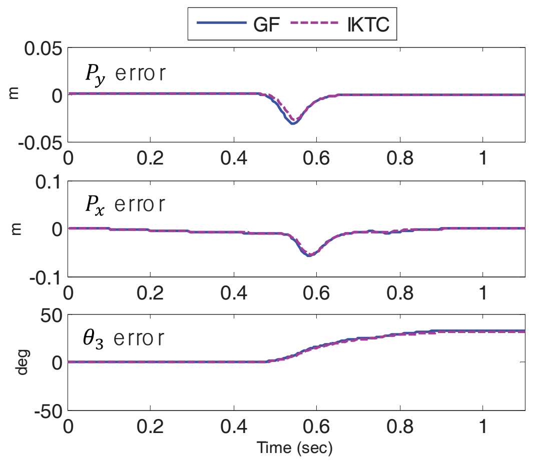


Figure 8. Error trajectories of the three tasks in the simulation in Section 3.3.

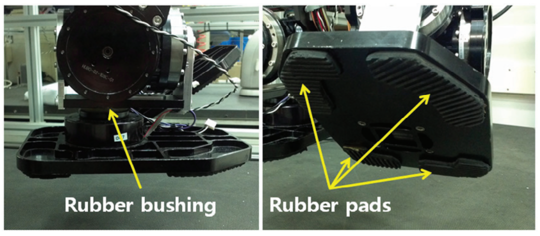


Figure 9. Compliant materials around the foot of humanoid robot *Hubo*.

4. Damping Controller

4.1 Inverted Pendulum Model with a Compliant Joint (IPCJ)

The compliance around the foot of a humanoid robot can be thought of as a combination of translational springs and rotational springs, as illustrated in Figure 10. Since the translational movement is negligible and a position controlled humanoid robot can be regarded as a rigid body in a moment, we use the IPCJs as illustrated in Figure 11 for the damping controller. We use an IPCJ even in DSP because the coupling moment due to translational springs is not ignorable in DSP, as shown in Figure 10.

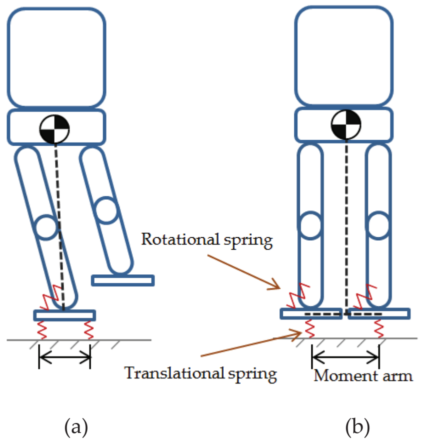


Figure 10. Inverted pendulum models with springs that represent compliances around feet. (a) SSP (b) DSP.

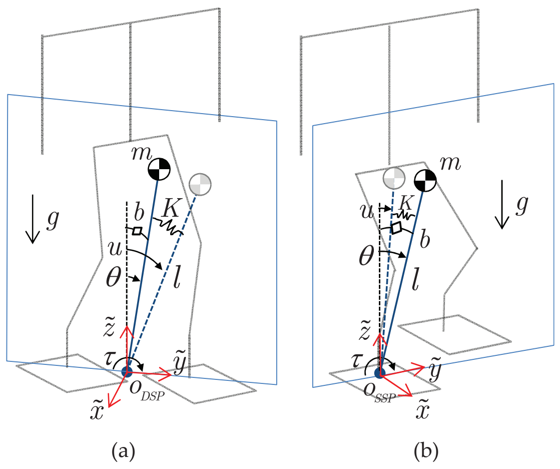


Figure 11. IPCJs defined in local $\tilde{y}\tilde{z}$ planes in (a) DSP and (b) SSP.

In each of the supporting phases, we define a local coordinate frame $\tilde{x}\tilde{y}\tilde{z}$, as shown in Figure 11, and apply damping controllers in the $\tilde{x}\tilde{z}$ plane and $\tilde{y}\tilde{z}$ plane, respectively. Figure 11 depicts only the two IPCJs used in the $\tilde{y}\tilde{z}$ plane, but two more IPCJs are used in the $\tilde{x}\tilde{z}$ planes in the same way.

The linearized equation of motion (EOM) of an IPCJ is

$$ml^2\ddot{\theta} + b\dot{\theta} - mgl\theta = \tau = -K(\theta - u) \qquad (26)$$

where m and l are the mass and length of the pendulum, θ denotes the actual inclined angle due to the compliance, and u is the reference angle derived from the reference position of the centre of mass (COM) of the humanoid robot. K is the stiffness and b is the damping coefficient of the compliance; τ is the exerted torque on the pendulum and g is gravity.

In designing a damping controller, we use ZMP rather than the torque τ because ZMP can be measured not only in SSP but also in DSP.

If we choose ZMP as the output of IPCJ, then the state-space representation of IPCJ is

$$\frac{d}{dt}\begin{bmatrix}\theta\\ \dot{\theta}\end{bmatrix} = \underbrace{\begin{bmatrix}0 & 1\\ \frac{1}{g}-\frac{K}{ml^2} & \frac{b}{ml^2}\end{bmatrix}}_{\mathbf{A}}\begin{bmatrix}\theta\\ \dot{\theta}\end{bmatrix} + \underbrace{\begin{bmatrix}0\\ \frac{K}{ml^2}\end{bmatrix}}_{\mathbf{B}}u, \qquad (27)$$

$$y_{ZMP} = \underbrace{\begin{bmatrix}\frac{K}{mg} & \frac{b}{mg}\end{bmatrix}}_{\mathbf{C}}\begin{bmatrix}\theta\\ \dot{\theta}\end{bmatrix} \underbrace{-\frac{K}{mg}}_{\mathbf{D}}u. \qquad (28)$$

The measured ZMP is given by

$$\begin{bmatrix}\tilde{x}_{ZMP}\\ \tilde{y}_{ZMP}\end{bmatrix} = \frac{1}{f_{RFz}+f_{LFz}}\begin{bmatrix}-\tau_{\tilde{y}}\\ \tau_{\tilde{x}}\end{bmatrix}, \qquad (29)$$

$$\begin{bmatrix}\tau_{\tilde{x}}\\ \tau_{\tilde{y}}\\ \tau_{\tilde{z}}\end{bmatrix} = \vec{\tau}_{LF} + \vec{\tau}_{RF} + \overrightarrow{OP}_{RF}\times\vec{f}_{RFz} + \overrightarrow{OP}_{LF}\times\vec{f}_{LFz} \qquad (30)$$

where f_{RFz} and f_{LFz} are measured normal forces, and $\vec{\tau}_{RF}$ and $\vec{\tau}_{LF}$ are measured torques on the right foot and the left foot, respectively. $\overrightarrow{OP}_{RF}$ and $\overrightarrow{OP}_{LF}$ denote the position vectors of the right foot and the left foot in the $\tilde{x}\tilde{y}\tilde{z}$ coordinate frame.

4.2 Model Identification

The parameters of IPCJ can be identified through frequency response methods. We measured sinusoidal responses for various input frequencies with our humanoid robot *Hubo* [16] in four cases: $\tilde{x}\tilde{z}$-DSP, $\tilde{y}\tilde{z}$-DSP, $\tilde{x}\tilde{z}$-SSP, and $\tilde{y}\tilde{z}$-SSP; one set of results is presented in Figure 12.

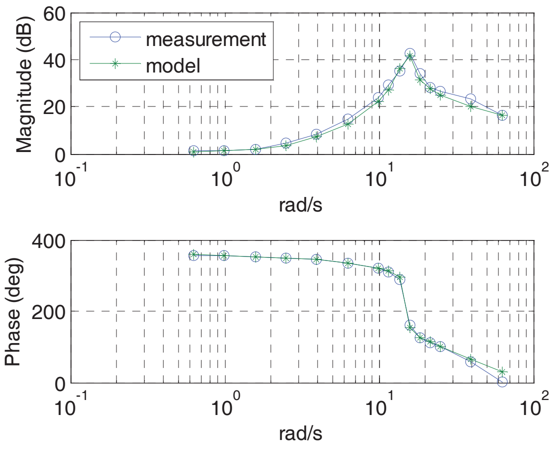


Figure 12. Bode plot of $\frac{\tilde{y}_{ZMP}}{u}$ in the local $\tilde{y}\tilde{z}$ plane in DSP.

In Figure 12, the measured response shows increasing phase shift and nonzero slope of the magnitude plot in the high frequency domain. This means time delay and unmodelled dynamics exist in addition to IPCJ. Since the slope of the magnitude plot in the high frequency domain is about -20dB/dec, the unmodelled dynamics can be approximated by a first-order model. With the Pade approximation [14] of the time delay and the additional use of the first-order model, the transfer function of IPCJ is given by

$$\frac{y_{ZMP}}{u} = \left(\frac{-\frac{K}{mg}\left(s^2+\frac{g}{l}\right)}{s^2+\frac{b}{ml^2}s+\frac{g}{l}-\frac{K}{ml^2}}\right)\left(\frac{1}{\tau_s s+1}\right)\left(\frac{-s+\frac{1}{T}}{s+\frac{1}{T}}\right) \qquad (31)$$

where τ_s is the time constant of the first-order model and T is the time delay. With the transfer function in (31), the real system can be approximated quite well, as shown in Figure 12.

4.3 Pole Placements

Figure 13 shows the root locus of the IPCJ system identified in Figure 12. As shown in the figure, the system is a lightly damped non-minimum phase system.

The damping characteristic of the system can be improved by moving the IPCJ poles to more damped poles. The poles of (31) are

$$P = \left\{-\zeta w_n \pm w_n\sqrt{1-\zeta^2},\ -\frac{1}{\tau_s},\ -\frac{1}{T}\right\} \qquad (32)$$

where $w_n = \sqrt{\frac{g}{l}-\frac{K}{ml^2}}$ and $\zeta = \frac{b}{2w_n ml^2}$.

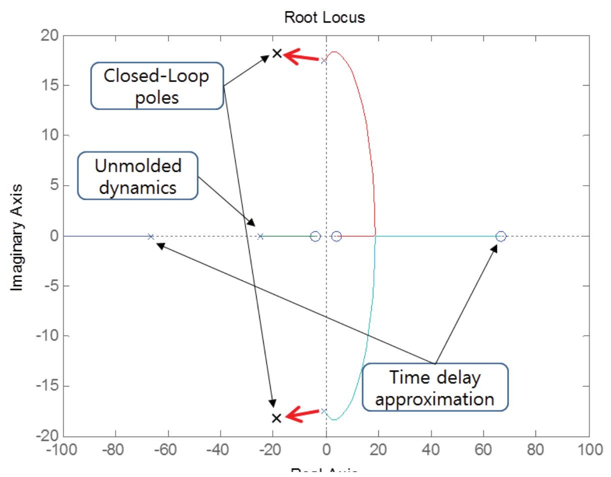


Figure 13. Root locus of the IPCJ system identified in Figure 12.

Since the poles of the IPCJ dynamics are dominant, we move only those poles. Then, the desired poles can be written as

$$P_{des} = \left\{ -\zeta_{des} w_{des} \pm w_{des}\sqrt{1-\zeta_{des}^2},\ -\frac{1}{\tau_s},\ -\frac{1}{T} \right\} \quad (33)$$

where ζ_{des} and w_{des} are the desired damping ratio and desired natural frequency, respectively. The state feedback gain $\mathbf{K}_f$ can be found by comparing the coefficients of the characteristic equation $\det\left(s\mathbf{I}-\left(\mathbf{A}-\mathbf{B}\mathbf{K}_f\right)\right)$ with those of the characteristic equation that has the desired poles as its roots. A block diagram of the damping controller is shown in Figure 14.

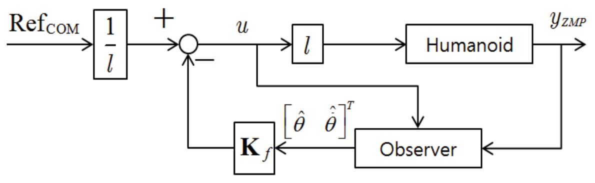


Figure 14. Block diagram of the damping controller.

5. Experiments

5.1 Experimental Setup

For the experiments, we use a life-size humanoid robot, *Hubo* [16], developed at KAIST. Each of the robot's joints has a local servo controller that uses high-gain PD position control law and controls the joint motor at 1 kHz control frequency. The inverse kinematic control framework runs in an embedded computer and sends position commands to the lower body joints at 200 Hz and to the upper body joints at 100 Hz.

The kinematic and inertial parameters are obtained from a 3D CAD model of *Hubo*. While the joint position limit is derived explicitly from the mechanical design of the joint, we decided on the joint velocity and acceleration limits experimentally.

5.2 Effect of the Damping Controller

In order to identify the system parameters, we conducted frequency response tests for four cases: $\tilde{x}\tilde{z}$-DSP, $\tilde{y}\tilde{z}$-DSP, $\tilde{x}\tilde{z}$-SSP, and $\tilde{y}\tilde{z}$-SSP. Damping controllers were designed to move the open-loop poles to damped ones such that damping ratios became 0.72.

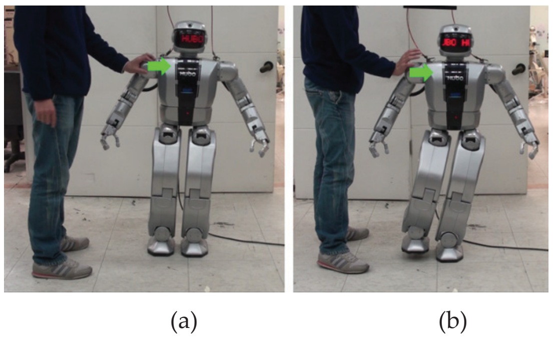


Figure 15. Exerting external force on the humanoid robot to verify the effectiveness of the proposed damping controller. (a) DSP (b) SSP

As shown in Figure 15, we exerted external forces on the humanoid robot and measured ZMP. Figure 16 and Figure 17 show the results. From these figures, it can be verified that the damping controller damps out the oscillation caused by external force.

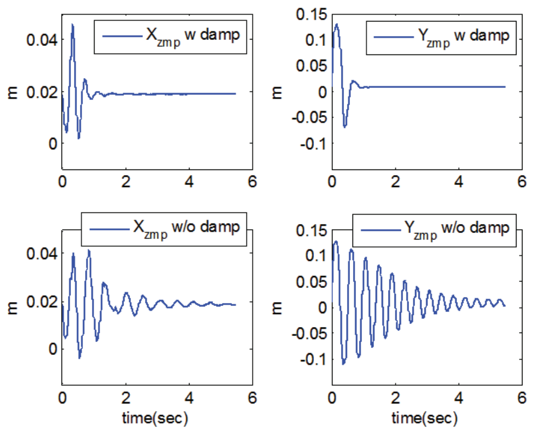


Figure 16. Time response of measured ZMP in DSP.

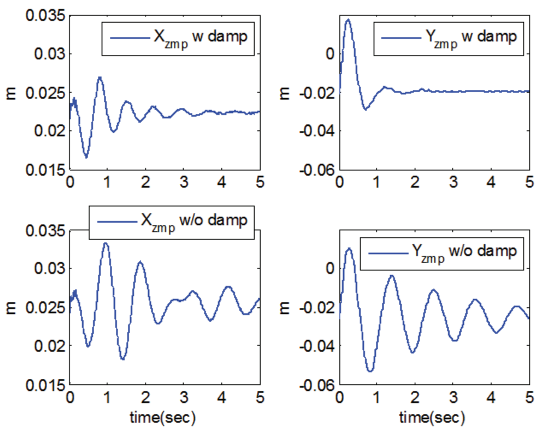


Figure 17. Time response of measured ZMP in SSP.

5.3 Imitating Captured Motion by a Humanoid Robot

In this experiment, a captured human motion was imitated by the humanoid robot in order to verify the proposed inverse kinematic control framework. We used captured motion data given as joint trajectories for the arms and attitudes for the pelvis. Tracking of the captured motion data was treated as one task among others, e.g., positioning of COM for balancing. For the given joint constraints, a solution satisfying the constraints and tasks in order of priority was obtained using the proposed methods.

The captured motion data contain stepping but we used our own walking pattern for the lower body motion. Because of the kinematic and dynamic differences between humanoids and humans, the imitation of stepping motion requires more complicated mapping of the captured motion and thus is beyond the scope of this paper.

The tasks for imitating the captured motion are shown in Table 1.

Priority	Tasks	DOF
1 (Highest)	Tracking of foot trajectories.	12
2	Tracking of COM trajectories (horizontal plane).	2
3	Tracking of pelvis attitude trajectories.	3
4	Tracking of a pelvis height trajectory.	1
5 (Lowest)	Tracking of the upper body joint trajectories (arms, waist and neck).	16
	Total	34

Table 1. Tasks for imitating a captured motion.

In order to verify the IKTC algorithm, we narrowed the range limits of the left-knee joint (LKN) and the left-hip-roll (LHR) joint compared to the ordinary ranges. This caused corrections of the lower priority tasks, pelvis height, and attitude tasks.

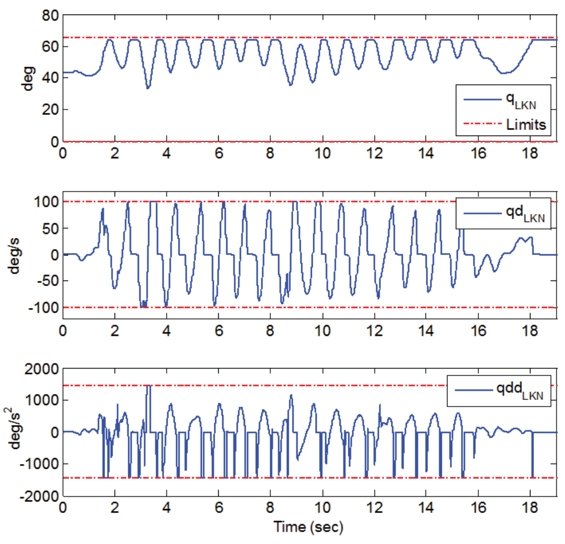


Figure 18. Trajectories of q_{LKN}, $\dot{q}_{LKN}$ and $\ddot{q}_{LKN}$ in Experiment 5.3.

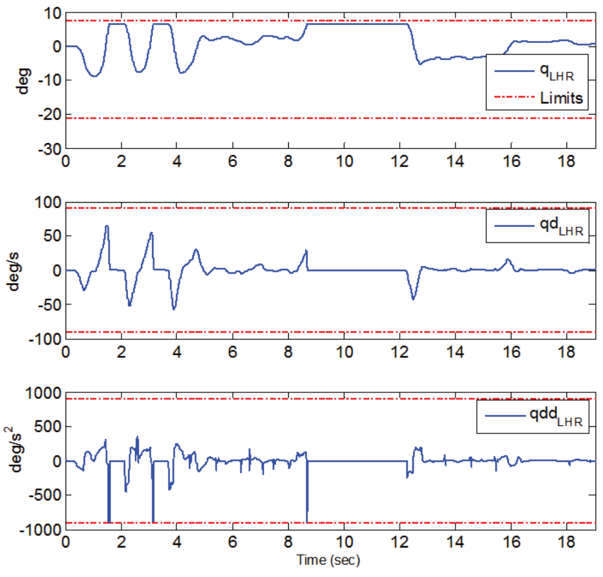


Figure 19. Trajectories of q_{LHR}, $\dot{q}_{LHR}$ and $\ddot{q}_{LHR}$ in the experiment 5.3.

As shown in Figures 18 and 19, the LKN and LHR joints did not violate constraints. Figure 20 shows task corrections of the pelvis height and pelvis attitude tasks. Figure 21 shows a comparison of the captured motion and the motion imitated by the robot.

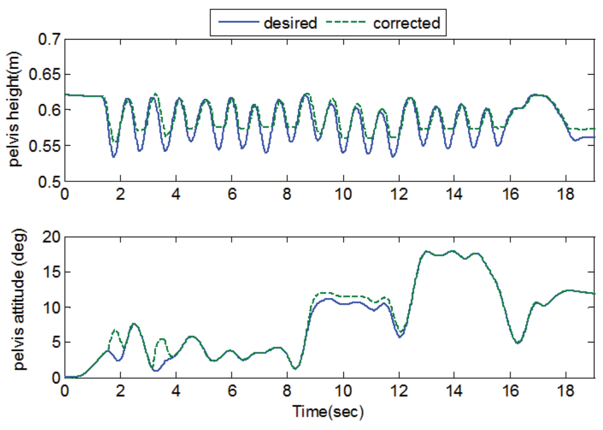


Figure 20. Desired task trajectories and their corrected trajectories in Experiment 5.3. Top: pelvis height task; bottom: pelvis attitude task.

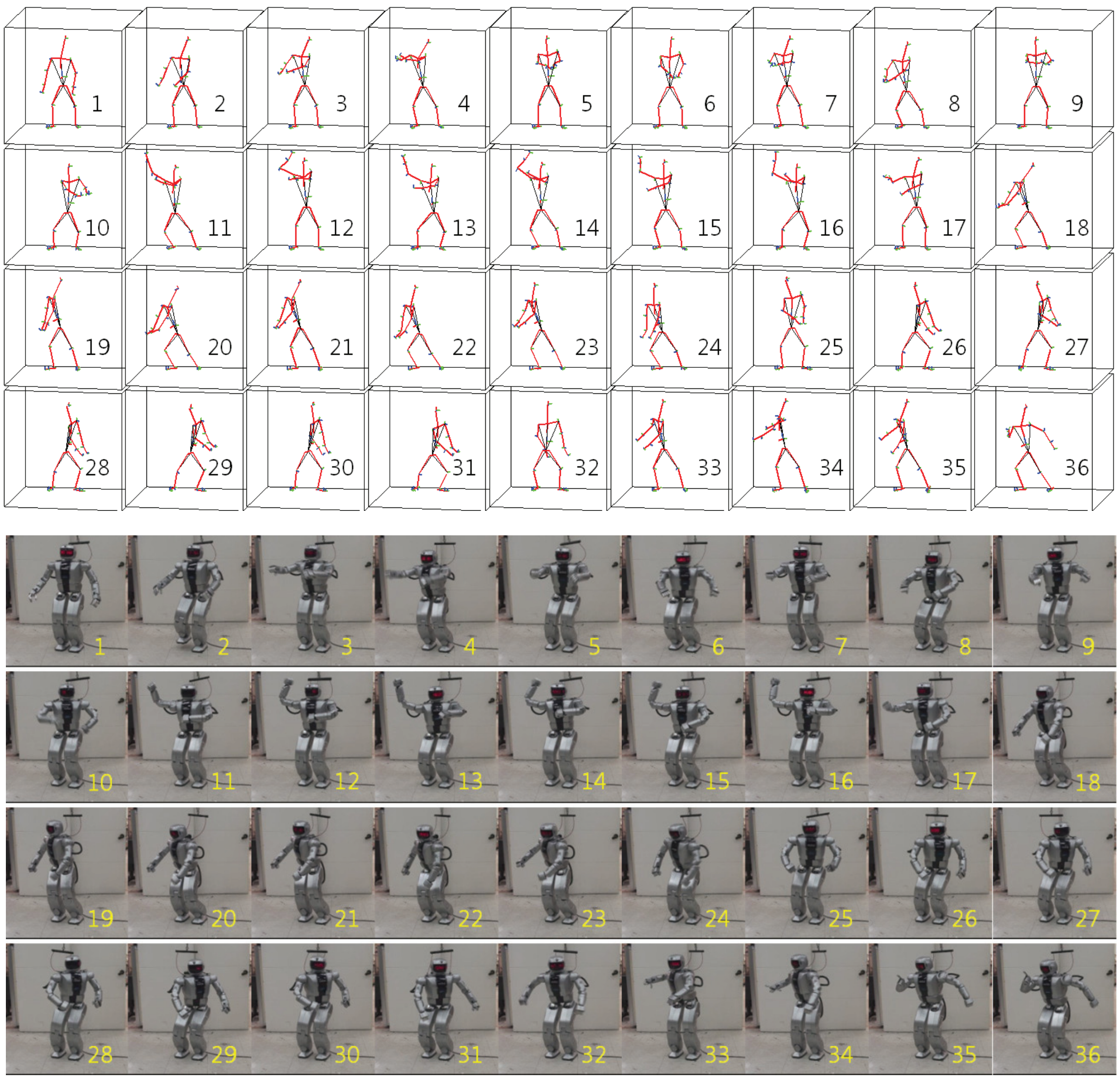


Figure 21. Captured human motion and imitated motion by a humanoid robot in Experiment 5.3.

6. Conclusion

In this paper, we proposed an inverse kinematic control framework for a position controlled humanoid robot with bounded joint range, velocity, and acceleration limits.

The proposed framework comprises IKTC and a damping controller.

In the IKTC algorithm, an inverse kinematics solution that considers task priority is found. The three different kinds of joint constraints are taken into account as second-order inequality constraints and are satisfied by optimal task corrections. The efficiency of the algorithm is attained by reducing the number of QPs to one.

The damping controller reduces the instability of the *stiff* position controlled humanoid robot. It uses IPCJ in order to take into account the compliance around the foot; this process is applicable not only in SSP but also in DSP by using ZMP measurement.

The validity of the proposed methods was shown through experiments.

7. References

[1] G. Antonelli, S. Chiaverini, G. Fusco, (2003) “A new on-line algorithm for inverse kinematics of robot manipulators ensuring path tracking capability under joint limits”, IEEE Trans. on Robotics and Automation, Vol. 19, No. 1, pp. 162-167.

[2] P. Chiacchio, S. Chiaverini, (1995) “Coping with joint velocity limits in first-order inverse kinematics algorithms: analysis and real-time implementation”, Robotica, Vol. 13, No. 5, pp. 515-519.

[3] T. F. Chan, R. V. Dubey, (1995) “A weighted least-norm solution based scheme for avoiding joint limits for redundant joint manipulators”, IEEE Trans. on Robotics and Automation, Vol. 11, No. 2, pp. 286-292.

[5] A. S. Deo, I. D. Walker, (1997) “Minimum effort inverse kinematics for redundant manipulators”, IEEE Trans. on Robotics and Automation, Vol. 13, No. 5, pp. 767-775.

[6] F. Flacco, A. De Luca, O. Khatib, (2012) "Motion control of redundant robots under joint constraints: Saturation in the null space", Proc. of IEEE Int. Conf. on Robotics and Automation, Saint Paul, MN, pp. 285-292.
[7] H. Hanafusa, T. Yoshikawa, Y. Nakamura, (1981) "Analysis and control of articulated robot with redundancy", IFAC, 8th Triennial World Congress, Vol. 4, pp. 1927-1932.
[8] O. Kanoun, F. Lamiraux, P. -B. Wieber, (2011) "Kinematic control of redundant manipulators: Generalizing the task-priority framework to inequality task", IEEE Trans. on Robotics, Vol. 27, No. 4, pp. 785-792.
[9] O. Khatib, (1986) "Real-time obstacle avoidance for manipulators and mobile robots", Int. J. of Robotics Research, Vol. 5, No. 1, pp. 90-98.
[10] J. H. Kim, J. H. Oh, (2004) "Walking control of the humanoid robot platform KHR-1 based on torque feedback control", Proc. IEEE Int. Conf. on Robotics and Automation, New Orleans, LA, pp. 623-628.
[11] A. Liegeois, (1977) "Automatic supervisory control of the configuration and behavior of multibody mechanisms", IEEE Trans. on Systems, Man and Cybernetics, Vol. SMC-7, No. 12, pp. 868-871.
[12] A. A. Maciejewski, C. A. Klein, (1985) "Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments", Int. J. of Robotics Research, Vol. 4, No. 3, pp. 109-117.
[13] Y. Nakamura, H. Hanafusa, T. Yoshikawa, (1987) "Task-priority based redundancy control of robot manipulators", Int. J. of Robotics Research, Vol. 6, No. 2, pp. 3-15.
[14] N. S. Nise, (2000) Control Systems Engineering (3rd Edition), John Wiley & Sons, Inc., New York.
[15] J. Nocedal, S. J. Wright, (2006) Quadratic Programming. In: Numerical Optimization (2nd Edition), Springer, New York.
[16] I. W. Park, J. Y. Kim, J. Lee, J. H. Oh, (2005) "Mechanical design of humanoid robot platform KHR-3 (KAIST Humanoid Robot - 3 : HUBO)", Proc. IEEE-RAS Int. Conf. on Humanoid Robots, Tsukuba, Japan, pp. 321-326.
[17] B. Siciliano, (1990) "A closed-loop inverse kinematic scheme for on-line joint-based robot control", Robotica, Vol. 8, pp. 231-243.
[18] B. Siciliano, J. -J. E. Slotine, (1991) "A general framework for managing multiple tasks in highly redundant robotic systems", Proc. of Int. Conf. on Advanced Robotics, 'Robots in Unstructured Environments', Pisa, Italy, pp. 1211-1216.
[19] C. W. Wampler II, (1986) "Manipulator inverse kinematic solutions based on vector formulation and damped least-squares methods", IEEE Trans. on Sys., Man and Cyb., Vol. SMC-16, No. 1, pp. 93-101.
[20] M. Vukobratovic, B. Borovac, (2004) "Zero-Moment Point – Thirty five years of its life", Int. J. of Humanoid Robotics, Vol. 1, No. 1, pp. 157-173.