NAIST-IS-DD1361009

Doctoral Dissertation

Real-time Whole-body Motion Generation for Humanoid Robots Using Torso Posture Regression and Center of Mass

Satoki Tsuichihara

June 18, 2018

Graduate School of Information Science
Nara Institute of Science and Technology

A Doctoral Dissertation
submitted to Graduate School of Information Science,
Nara Institute of Science and Technology
in partial fulfillment of the requirements for the degree of
Doctor of ENGINEERING

Satoki Tsuichihara

Thesis Committee:

Professor Tsukasa Ogasawara	(Supervisor)
Professor Kenji Sugimoto	(Co-supervisor)
Associate Professor Jun Takamatsu	(Co-supervisor)
Assistant Professor Ming Ding	(Co-supervisor)
Associate Professor Masanao Koeda	(Osaka Electro-Communication University)

Real-time Whole-body Motion Generation for Humanoid Robots Using Torso Posture Regression and Center of Mass*

Satoki Tsuichihara

Abstract

For humanoid robots, reaching with their hands as much workspace as possible is an important issue, since the locations of the target objects are ranging from the floor to the place above robot's head. For humanoid robots, it is an important issue to reach with their hands as much workspace as possible, since the locations of the target objects are ranging from the floor to the place above robot's head. Furthermore, it is necessary to solve inverse kinematics for the whole body in real time to adapt to the constantly-changing environment.

First, we propose a dual-arm inverse kinematics with a visual feedback mechanism. The proposed solution uses a weighted pseudo inverse matrix and a pipeline calculation. The weights improve the robot's balance without significantly increasing the calculation time.

Second, we propose a method to achieve real-time motion generation for a humanoid robot by separating the inverse kinematics calculation into simpler problems. Using regression to estimate the torso orientation, we solve the inverse kinematics for the lower body and both arms independently. Based on the target pose of both hands as input, we calculate the orientation of the torso and determine the target position of the center of mass considering the reachability of both arms. At each control step, we calculate the joint angles of the lower body from the position of the center of mass, feet poses, and torso orientation. Finally, we calculate the joint angles of both arms.

*Doctoral Dissertation, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-DD1361009, June 18, 2018.

We experimentally verify the proposed method by generating motions to approach objects. We verify the effectiveness of the proposed method to keep the robot's balance using the ZMP. We also apply the proposed method to a human-size humanoid robot for reaching low-height positions while hunkering down. From the experimental results, we prove the following advantages of the proposed method. The proposed inverse kinematics solver is ten times faster than the numerical solution using the Jacobian matrix. The accuracy of the hand pose control and the calculation time are better than the conventional method. The error and the calculation time are lower than with the conventional method. The error using the proposed method is smaller especially when the target positions are at the far side in the lateral direction. We verify the applicability of the proposed method by following a sequence of random hands' target positions and the precision of the proposed method by comparing to a conventional method which proposed in these years.

Keywords:

Whole-body motion generation, Humanoid robots, Vision-based feedback, Posture Estimation

胴体姿勢推定と重心を用いたヒューマノイドロボットのための実時間全身動作生成*

築地原 里樹

内容梗概

家事を代行するヒューマノイドロボットにとって，自身の手が広い領域に到達できることは重要な課題である．例えば，物体を操作するような作業において，対象物の場所は地面から人型ロボットの頭上まで及ぶ．また，環境が常に変化するためこれらの変化に適用するためには，全身の逆運動学が実時間で計算できることが必要になる．

本研究では，まず視覚フィードバックを用いた上体の逆運動学について提案する．上体の運動学に対して，重み付き擬似逆行列を用いて各部の動きやすさを調整することで，動作の安定性を向上する．重み付き擬似逆行列や OpenRTM におけるパイプライン処理を用いることで，実時間性を向上する．各関節に応じた重みを用いることで，計算時間を多くかけず，安定な動作を獲得する．

次に，実時間の動作生成を達成するために逆運動学を簡素な問題へ分解する方法を提案する．胴体姿勢の回帰器を利用することで，下半身と双腕の逆運動学を独立に計算する．入力の両手先位置に対して，胴体姿勢を計算し，双腕の到達度に基づき目標の重心位置を計算する．ロボットの各制御周期において，重心位置，足の位置姿勢と胴体姿勢から下半身の関節角を求め，最後に双腕の関節角を計算する．

実験では，ヒューマノイドロボットが目標位置に手を伸ばす動作に提案手法を適用する．動作中の ZMP の軌道を評価することで，提案手法を到達運動に適用した際の安定性を確認する．また，ヒューマノイドロボットに対して，屈伸動作を伴う低い位置への目標に対する到達運動に本手法を用いた．提案した逆運動学法は，ヤコビアンを用いた数値的解法よりも 10 倍早く動作生成ができることが

*奈良先端科学技術大学院大学 情報科学研究科 博士論文, NAIST-IS-DD1361009, 2018 年 6 月 18 日.

確認できた．複数の到達運動での手先の誤差ノルム，計算時間や到達時間を評価し，従来の数値解法よりも改善できることが確認できた．特に左右に目標位置が遠い場合に，提案手法が胴体姿勢を事前に獲得できることから精度向上や動作生成にかかる計算時間が短いことが確認できた．連続的にランダムな目標位置を追従する動作に適応することで本手法の応用性を，近年用いられる従来手法と比較することでより高い精度で重心と手先位置を制御できていることが確認できた．

キーワード

全身動作生成, ヒューマノイドロボット, 視覚フィードバック, 胴体姿勢推定

Contents

1. Introduction **1**
1.1 Background 1
1.2 Contribution 4
1.3 Dissertation layout 5

2. Related research **6**
2.1 Pre-defined motion generation 6
2.2 Motion genration from scratch 6
2.3 Relationship to this dissertation 7

3. Building the kinematics for the whole-body motion generation **8**
3.1 Overview of the manipulation system 8
3.2 Measuring 3D position of each object 10
3.3 Inverse kinematics for the upper body 11
3.4 Generating motions using inverse kinematics 12
3.5 Weighted pseudo-inverse matrix 12
3.6 Establish the parallelism in whole components 13

4. Speeding up the whole-body motion generation **14**
4.1 Estimation of torso posture 14
4.2 Target CoM position generation 17
4.3 Inverse kinematics for the lower body 19
4.4 Calculation cost 22
4.5 Implementation 23
4.5.1 Overview of the manipulation system 23
4.5.2 Generating motion using inverse kinematics 23
4.5.3 Posture estimator 24
4.5.4 Measuring the 3D position of each object 25

5. Experiments **26**
5.1 Evaluation of a weighted pseudo-inverse matrix 26
5.1.1 Testbed 26
5.1.2 Objective 26

5.1.3 Result 28
5.2 Evaluation of the whole-body controller 33
5.2.1 Comparing the calculation times using dynamics simulation 33
5.2.2 Real-robot experiments 37
5.2.3 Accuracy and calculation speed 39
5.2.4 Comparing the accuracy to a conventional method 40
5.3 Following markers 42
5.4 Sequential whole-body control 42
5.5 Application to a living environment 43

6. Conclusion **45**
6.1 Contribution 45
6.2 Future work 46

Acknowledgements **47**

References **50**

List of Figures

1 Example of household task performed by human and humanoid robot . . . 3
2 Overview of the proposed manipulation system . . . 9
3 The relations between each coordinate system. . . . 10
4 The initial posture of the humanoid robot for the learning procedure. 15
5 Humanoid's postures obtained from the SVR and the inverse kinematics of the arms. . . . 15
6 Movable areas (yellow boxs) of the proposed method in the three configuration of CoM. . . . 18
7 Overview of the proposed method for speeding up whole-body motion generation. . . . 21
8 The data flow in both the posture estimator and the whole-body controller. . . . 23
9 The calculation time of SVR, lower-body IK and both arms IK. . 24
10 The relations between each coordinate system. . . . 24
11 The results of estimated marker positions using the ARToolKit library. . . . 27
12 Standard pseudo-inverse matrix controls HRP–4's hand to the target position of AR marker. . . . 28
13 Weighted pseudo-inverse matrix controls HRP–4's hand to the target position of AR marker. . . . 28
14 A comparison between the standard and weighted pseudo-inverse matrices. . . . 31
15 Six examples of the target position of both hand using Inverse Kinematics with weighted pseudo-inverse matrix. . . . 32
16 The posture of HRP-4 generated by the proposed method. . . . 34
17 Comparison of the calculation times. . . . 35
18 The ZMP of both feet and CoM during the whole-body movement. 35
19 The error norm of the CoM position and the end effector during the whole-body movement. . . . 36
20 The result of the marker positions estimated using the ARToolKit library. . . . 37

21 The proposed method generates whole-body motion to make the HRP-4 reach with both hands the target positions indicated with AR markers. 38
22 The error norm of the CoM position and the end effector during the whole-body movement using a conventional method. 41
23 From a camera input, HRP-4 can follow the target position of AR markers using the proposed method. 42
24 The simulation result when approaching random target positions. 43
25 The simulation result is applied to the living environment using the proposed method. 44

List of Tables

1 Parameters for the regression algorithm. 16
2 Comparison of results in both methods. 39

1. Introduction

1.1 Background

As the aging society grows, service robots are expected to help elderly people to keep their independence. The service robots can perform household tasks instead of humans, such as cleaning rooms [1], cooking [2], and bringing tools [3]. Many types of service robots can work in a specific place. Human-living environments have many locations to place objects, for example, books in a shelf, dishes in the sink, and food below the kitchen counter. Recently, to use the human-living environments, robot motion generation based on human behavior is growing into a popular research area [4].

Humanoid robots have a human-like shape and a similar structure to a human body. Recently, humanoid robots have a higher functionality to work in the human-living environment than other robots. Each manipulator of humanoid's body, such as legs and arms, has more than six joints to move the end effector in 3D space (*e.g.*, six DoFs space of position and orientation). The humanoid robot's arms can manipulate tools and reach objects located on the table, shelf, and so on. The humanoid robot's legs can manage the balance, and can walk to many places. Combining these two abilities in the arms and the legs, the humanoid robots can bring to a far area, for example, move cups from a table to a shelf, bring coffee from another floor, and can reach all locations in the shelf using the whole body.

For a robot to execute the sequences of reaching motions, these steps are repeated:

1) determine the target configurations of the hands based on the current status of the environment,

2) solve the whole body IK to satisfy these configurations,

3) move the robot following the IK solution.

Unlike well-controlled environments in factories, household environments continuously change. Thus, the whole body IK should be solved every time. Since the

robot stops moving until finishing the IK calculation, the calculation time can not be ignored.

To accelerate the calculation of IK, a number of issues must be considered, including:

i) the large number of DOF of a humanoid robot, and

ii) the robot's balance during execution.

Furthermore, the target positions to be reached can range from objects on the floor to those above the robot's head. As exemplified in Figure 1, many household tasks require the robot to pick up objects at a very low height. Since it is very difficult to solve these issues in general, we concentrate on tasks where the robot keeps standing. Also, we ignore the collision avoidance issue (*i.e.*, the robot moves around a large free space). We consider that the collision avoidance can be achieved by inputting appropriate trajectories of hands using a path planning technique.

Kuffner *et al.* proposed a motion planner for a humanoid robot in a dynamics simulation, where the use of a large number of DoF in a humanoid robot can expand its manipulation space to be like that of a human [5]. Cognetti *et al.* proposed whole-body planning for a humanoid robot based on a given task in a dynamics simulation environment [6]. This planner can generate joint angle trajectories for the whole body based on the destination of a hand, and the humanoid robot can pick up a ball from the lower shelf and place it in the middle part of the shelf. These approaches can generate whole-body motion in a simulation environment but for the actual robot implementation additional issues, such as calculation cost and the accuracy of controlling the end effector.

Moreover, when performing these kinds of household tasks in human environments, task completion time should also be considered, since in several tasks robots including humanoid robots have a time limit to fulfill human expectations. One difficulty in reducing the task completion time comes from environmental changes by humans. Even if we ignore the collision avoidance issue (*e.g.*, moving around the huge free space), robots need to generate the motions to make the end effectors reach target poses in real time. Unfortunately, a humanoid robot

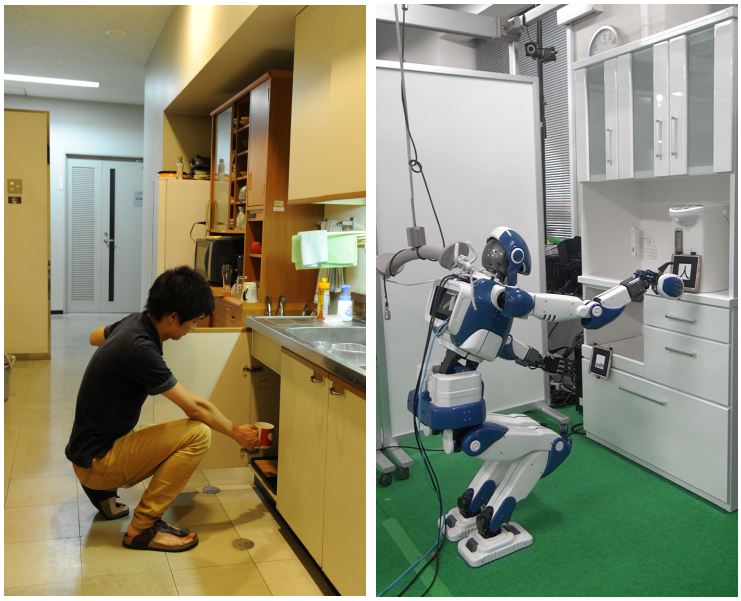


Figure 1. Human (left) and humanoid robot (right) perform a manipulation task that requires hunkering down to reach the lower shelf in a kitchen.

must decide the movement of a large number of joints to reach target positions while also considering its balance.

1.2 Contribution

We consider three problems in this dissertation:

- whole-body balance
- limitation of the manipulation space
- real-time adaptation to environmental changes.

We propose the method to calculate the whole-body inverse kinematics when inputting the target configurations of both end effectors (*i.e.*, hands). To solve the first problem, we tune the kinematics in each part using a weight pseudo-inverse matrix. To solve the second problem, we consider the motion including height change, (*e.g.*, hunkering down). To solve the third problem, we split the whole-body inverse kinematics problem into simpler inverse kinematics subproblems. The idea is that if the torso posture is known, the remaining inverse kinematics is simpler by separately solving the inverse kinematics for both arms and the lower body. In our method, first, we use regression to estimate the torso posture from the input position of the hands. Then, we solve the IK of the lower body to determine the torso posture and the target CoM. Finally, we solve the IK of each arm independently. The purpose of the proposed method is to accelerate the whole body motion generation. We select only the torso without both arms for the regression.

We perform experiments using the humanoid robot HRP-4 [7] to prove the effectiveness of the proposed method. We verify that the proposed method calculates the inverse kinematics ten times faster than the numerical solution (*i.e.*, Jacobian-based method). In the real-robot experiments, we verify that the proposed method generates whole-body motions to reach low-height positions while hunkering down and keeping balance. In these experiments, both the inverse kinematics calculation and the robot's movement take three seconds in average. In the dynamics simulation experiments, we additionally verify the applicability of the proposed method to several sequential random target positions of the hands; the humanoid stands and continuously reaches the targets while keeping its balance. Additionally, we verify the applicability of the proposed method when target positions are sequentially input. In practice, it is unrealistic for a

robot to move to the initial pose after reaching the target position every time. In that sense, we verified the usefulness of the proposed method in an extreme condition.

The proposed method assumes that the humanoid robot meets the following conditions:

1. The robot changes the configuration of the waist by moving the feet.
2. The torso is connected to the waist by two chest joints (in the pitch and yaw directions).
3. Each arm has at least 6 DoF.

Most human-size humanoid robots satisfy these conditions. To consider the singularities, we can solve the inverse kinematics of the arms analytically.

1.3 Dissertation layout

The rest of this dissertation is organized as follows:

Chapter 2 This chapter introduces research related to this dissertation.

Chapter 3 This chapter shows how to manage the kinematics of the whole body in the humanoid robots to keep its balancing.

Chapter 4 This chapter describes a method to speed up the whole-body motion generation.

Chapter 5 This chapter shows the results and discussion regarding the whole-body motion generation.

Chapter 6 This chapter concludes this dissertation and summarizes future work.

2. Related research

Existing research efforts regarding motion generation [8, 9] can be roughly classified into two categories. The first category includes methods to refine a pre-defined motion to satisfy the target configuration, while the second category includes methods to generate the motion from scratch.

2.1 Pre-defined motion generation

In the first category, Vannoy *et al.* proposed selecting one of multiple pre-defined motions considering the environment and a criterion of the motion quality [10]. Park *et al.* presented the use of non-linear optimization to adjust the pre-defined motions [11]. Otte *et al.* refined and repaired the pre-defined motions using a search-graph algorithm [12]. Since it is difficult for pre-defined motions to cover the whole range of target motions, the main purpose of these methods is real-time collision avoidance for sudden changes in the environment.

2.2 Motion genration from scratch

In the second category, Fok *et al.* used a numerical method (*i.e.*, Jacobian-based) to solve the IK of the whole body [13]. Their main concern was implementing a middleware structure, not accelerating the calculation. Nishiwaki *et al.* proposed whole-body motion generation for reaching an object using one arm [14]. They tested their method with a real robot grasping an object on the floor. Nevertheless, they did not show the applicability of their method to reaching objects using both arms. Yamane *et al.* proposed a pin-and-drag interface for a human agent in a dynamics simulator [15]. They controlled the humanoid robot by pulling *links* in the dynamics simulation. Their method is suitable for computer graphics, where the animator controls the kinematics of the animated model. Ferrari *et al.* proposed a manipulation movement including locomotion for the humanoid robot [16]. They plan the CoM trajectory based on the distance to the target object and use the Jacobian-based whole-body IK to satisfy the planned CoM trajectory [17]. They tested their method in a dynamics simulation with a motion of reaching an object on a table. All these methods employ the standard

Jacobian-based IK.

On the other hand, Khatib *et al.* proposed to use a potential field for real-time motion generation [18]. In this method, at every time step, the control input is decided from the derivatives of the field. Zucker *et al.* proposed to use distance fields for the trajectory formulation by optimizing a function that trades off between a smoothness component and an obstacle avoidance component [19]. On the other hand, Yang *et al.* proposed to use deep learning to determine the control input from an observed image [20]. They applied their proposed method to folding a towel. Unfortunately, they did not prove the applicability of their method for whole-body motion generation of humanoid robots. They applied their method to a 6-DoF manipulator in [18] and upper-body dual-arm robot in [20].

2.3 Relationship to this dissertation

Our proposed method belongs to the second category. All methods in the second category use a Jacobian-based numerical method. Unlike all the previous methods, the proposed method reduces the calculation time by solving the center of the whole body (*e.g.*, the torso) for managing the redundancy of the whole body well with a machine learning technique. Also, the methods in the first category are complementary to the second category. It is possible to use methods from the second category to modify the pre-defined motion. We expect that combining methods from the second category with the first category can enable the proposed method to deal with real-time collision avoidance.

3. Building the kinematics for the whole-body motion generation

3.1 Overview of the manipulation system

Figure 7 shows the structure of the proposed system used to control the whole body of a humanoid robot for performing household tasks in daily-life environments. In this figure, the flow of the data goes from left to right. The system is composed of three major parts, environment recognition, motion generation, and the robot controller. As shown in the figure, the beginning of the proposed system consists of two elements for environment recognition: *Semantic Map* and ARToolKit. Semantic map includes the object and the method to manipulate the object. If the type of the object is identified, we obtain the target position to manipulate in the object from Semantic Map.

We use ARToolKit library [21] to estimate the position of the AR marker attached on the object as well as ID of the marker. In this chapter, we prove the effectiveness of the proposed method by controlling the upper-body based on the position of the objects obtained by ARToolKit library.

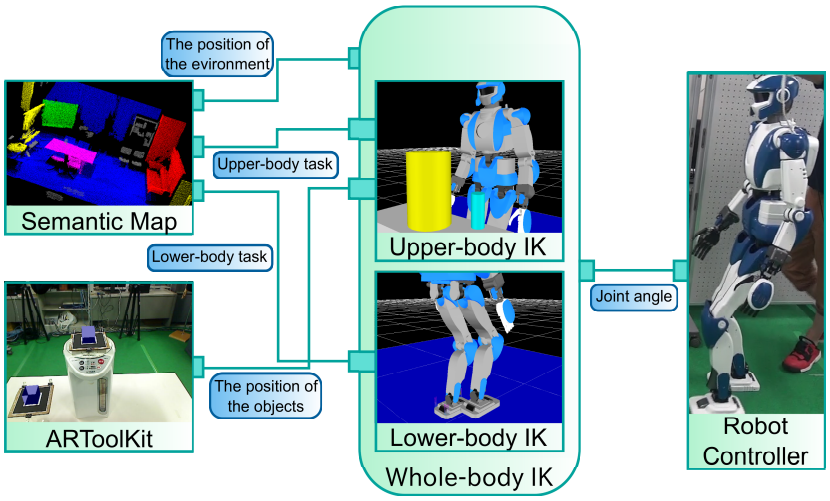


Figure 2. Overview of the proposed system. This diagram shows the flow (from left to right) of the data. The system is composed of the environment recognizer, the motion generator, and the robot controller. In the environment recognizer, a semantic map provides the robot with the method to manipulate the target object. ARToolKit estimates scene motion, such as a cup and a pot on the white table in this case. The whole-body inverse kinematics calculates the joint angles based on the positions, the objects, and the tasks separated between the upper body and lower body. Finally, the robot follows the joint angles calculated by the whole body inverse kinematics.

3.2 Measuring 3D position of each object

To manipulate or grasp an object in an unknown location, we calculate each joint angle by the following steps:

1. estimate the position of each object in the coordinate system of the camera mounted on the robot
2. translate from the coordinate system of the camera to that of the robot
3. calculate each joint angle using the inverse kinematics of the upper body for each target position in the coordinate system of the robot

We use the ARToolKit [21] library to acquire the position and orientation of a known marker attached to the object. Figure 10 shows the three coordinate systems of the robot $\mathbf{\Sigma}_{\mathrm{R}}$, the camera $\mathbf{\Sigma}_{\mathrm{C}}$, and the marker $\mathbf{\Sigma}_{\mathrm{M}}$. The matrices ${}^{\mathrm{R}}\boldsymbol{T}_{\mathrm{C}}$, ${}^{\mathrm{C}}\boldsymbol{T}_{\mathrm{M}}$, and ${}^{\mathrm{M}}\boldsymbol{T}_{\mathrm{O}}$ are homogeneous transformations from the coordinate system of the robot to that of the camera, the camera to that of the marker, and the marker to that of the object. The homogeneous transformation matrix from the coordinate system of the robot to that of the object is computed as

$$ {}^{\mathrm{R}}\boldsymbol{T}_{\mathrm{O}} = {}^{\mathrm{R}}\boldsymbol{T}_{\mathrm{C}}\ {}^{\mathrm{C}}\boldsymbol{T}_{\mathrm{M}}\ {}^{\mathrm{M}}\boldsymbol{T}_{\mathrm{O}}. \tag{1} $$

The matrix ${}^{\mathrm{R}}\boldsymbol{T}_{\mathrm{C}}$ is calculated by forward kinematics. We know ${}^{\mathrm{M}}\boldsymbol{T}_{\mathrm{O}}$ from where the marker is on the object.

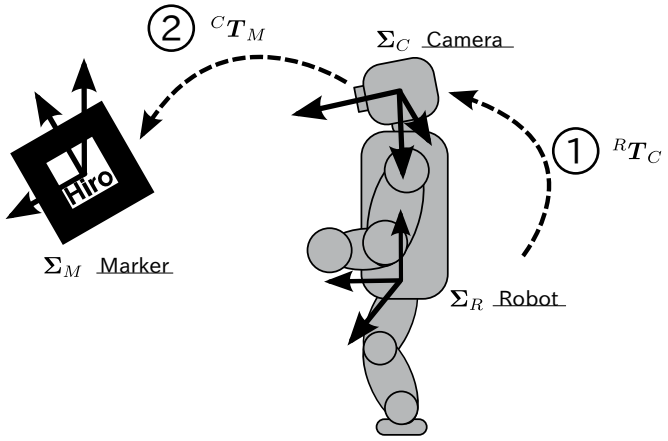


Figure 3. The relations between each coordinate system.

3.3 Inverse kinematics for the upper body

We control the position and orientation of the robot hands using both arms and the chest joint. Using the chest joint in a typical humanoid robot offers two advantages:

- expands the reachable area of the hand
- allows adjustment of the orientation of the upper body without using the lower body

Algorithm 1 Inverse kinematics

Input: $\boldsymbol{p}^*_{\text{right}}, \boldsymbol{R}^*_{\text{right}}, \boldsymbol{p}^*_{\text{left}}, \boldsymbol{R}^*_{\text{left}}$

Output: $\boldsymbol{q}$

$\boldsymbol{q} \leftarrow \boldsymbol{q}_0$
for $i = 1$ to $Iter$ **do**
  $\boldsymbol{p}_{\text{right}}, \boldsymbol{R}_{\text{right}}, \boldsymbol{p}_{\text{left}}, \boldsymbol{R}_{\text{left}}, \boldsymbol{J} \leftarrow \text{FK}(\boldsymbol{q})$
  $\boldsymbol{err} \leftarrow \begin{pmatrix} \boldsymbol{p}^*_{\text{right}} - \boldsymbol{p}_{\text{right}} \\ \text{LN}\left(\boldsymbol{R}^\top_{\text{right}} \boldsymbol{R}^*_{\text{right}}\right) \\ \boldsymbol{p}^*_{\text{left}} - \boldsymbol{p}_{\text{left}} \\ \text{LN}\left(\boldsymbol{R}^\top_{\text{left}} \boldsymbol{R}^\star_{\text{left}}\right) \end{pmatrix}$
  if $|\boldsymbol{err}| < tol$ **then**
    break
  end if
  $\boldsymbol{q} \leftarrow \boldsymbol{q} + \alpha \boldsymbol{J}^\sharp \boldsymbol{err}$
end for

Algorithm 1 describes the inverse kinematics algorithm shown in [22]. A vector $\boldsymbol{q} = \left(\boldsymbol{q}_{\text{chest}}, \boldsymbol{q}_{\text{right}}, \boldsymbol{q}_{\text{left}}\right)$ represents the upper body joint angles, the joint angles of the chest, right arm, and left arm. $\boldsymbol{q}_0$ is the current joint angles. $\boldsymbol{p}$ is the position of the end effector, $\boldsymbol{R}$ is its orientation, and $\boldsymbol{J}$ is the Jacobian matrix in the upper body. The subscript right or left indicates a right or left hand. The vector $\boldsymbol{x}^*$ is the target value of the value $\boldsymbol{x}$, and $\boldsymbol{Y}^\sharp$ is the pseudo-inverse matrix of the matrix $\boldsymbol{Y}$. FK is a forward kinematics function. FK's input values are the joint angles $\boldsymbol{q}$ and output values are the position $\boldsymbol{p}$, the orientation $\boldsymbol{R}$ and

the Jacobian $\boldsymbol{J}$. The input value of the function LN is the rotation matrix $\boldsymbol{R}$, and the output value is the vector of angular velocities [23]. The value *iter* is the number of maximum iterations. The value *tol* is the tolerance in the norm of the error vector $\boldsymbol{err}$. The error vector is 12 dimensions that consist of the positions (x, y, z) and angular vectors (ϕ, θ, ψ) of each hand. Using the divided Jacobian $\boldsymbol{J}_{\text{chest}}$, $\boldsymbol{J}_{\text{right}}$ and $\boldsymbol{J}_{\text{left}}$ for each body part from the Jacobian $\boldsymbol{J}$, the Jacobian can be represented as

$$\boldsymbol{J} = \begin{pmatrix} \boldsymbol{J}_{\text{chest}} & \boldsymbol{J}_{\text{right}} & \boldsymbol{0} \\ & \boldsymbol{0} & \boldsymbol{J}_{\text{left}} \end{pmatrix}. \tag{2}$$

3.4 Generating motions using inverse kinematics

To define the intermediate points from the initial position to the position of the marker, Algorithm 1 uses the input value the target position and orientation linearly interpolated. The fourth line in Algorithm 1 shows that the target position and orientation are used to linearly interpolate the difference between the target value and an initial value for execution time. The robot is controlled by the interpolated target position and orientation. The method of interpolation for the orientation is to multiply the initial rotation matrix and the rotation matrix calculated from the Rodrigues equation using a vector of the angular velocity from a difference between the target value and the initial value.

3.5 Weighted pseudo-inverse matrix

If a standard pseudo-inverse matrix in the eighth line of Algorithm 1, the joint angles calculated under the assumption that all joints move equally easily. A method for solving inverse kinematics with a weight value was referred in [24]. Using a diagonal matrix $\boldsymbol{W}$ with factors to make some joints *harder* to move than others, the pseudo-inverse matrix $\boldsymbol{J}_W^\sharp$ is defined as

$$\boldsymbol{J}_W^\sharp = \boldsymbol{W}^{-1}\boldsymbol{J}^\top \left(\boldsymbol{J}\boldsymbol{W}^{-1}\boldsymbol{J}^\top\right)^{-1}. \tag{3}$$

Instead of a standard pseudo-inverse matrix in the eighth line of Algorithm 1, we replace the updating rule of the inverse kinematics

$$\boldsymbol{q} \leftarrow \boldsymbol{q} + \alpha \boldsymbol{J}_W^\sharp \boldsymbol{err}. \tag{4}$$

3.6 Establish the parallelism in whole components

The proposed manipulation system has three components: the image capture using the HRP-4 camera, the motion generator, and the robot controller. We establish parallelism of the three components in the middleware OpenRTM [25]. Using this system, to send data between the components via the pipeline process, a *data port* is defined for each component. The data port consists of an *in-port* to receive the data and an *out-port* to send data. These connections are defined below:

Image capture: Camera capture and **ARToolKit**

Camera capture: capturing an image from the camera mounted on the HRP-4

- outport: send the captured image (to **ARToolKit**)

ARToolKit: proceeded by the ARToolKit library

- inport: receive the captured image (from **Camera capture**)
- outport: send the position of each object (to **Motion generator**)

Motion generator: generating each joint angle using the proposed inverse kinematics system

- inport: receive the position of each object (from **ARToolKit**)
- outport: send the joint angles (to **Robot controller**)

Robot controller: controlling the robot with the torque based on the joint angles

- inport: receive the joint angles (from **Motion generator**)

4. Speeding up the whole-body motion generation

4.1 Estimation of torso posture

We use regression to estimate the orientation of the torso for whole-body motion generation, The inputs of this regression are the positions of both arms, i.e., (x_1, y_1, z_1) and (x_2, y_2, z_2). The outputs of this regression are the joint angles of the chest joint (q_{pitch}, q_{yaw}). If we assume that the orientation of the waist is the same as in the resting pose, the torso orientation is obtained from the chest joint. We use the Support Vector Regression (SVR) [26] to calculate the orientation of the torso. The training dataset generation and the learning process are done offline, while in the actual robot the motion generator uses this regression online.

We use the Algorithm 2 to generate the dataset, which implements a numerical solution of inverse kinematics. We select the robot's configurations that satisfy the positions of both arms and then use only the chest joints' angles for the dataset. The height of the torso is fixed during learning and only the chest joints are learned. In this paper, we assume that the workspace is in front of the humanoid robot, and define the initial posture for the numerical inverse kinematics as shown in Figure 4. The robot's workspace is defined with 0.45, 0.8, and 0.75 m in the x (frontal), y (lateral) and z (vertical) axis from the posture as shown in the yellow boxes of Figure 4.

Algorithm 2 Dataset generation

Input: $\boldsymbol{q}_0, \boldsymbol{p}^*_{\text{left}}, \boldsymbol{R}^*_{\text{left}}, \boldsymbol{p}^*_{\text{right}}, \boldsymbol{R}^*_{\text{right}}$

Output: $\boldsymbol{p}^*_{\text{left}}, \boldsymbol{p}^*_{\text{right}}, \boldsymbol{q}_{\text{chest}}$

1: $\boldsymbol{q} \leftarrow \boldsymbol{q}_0$
2: **while** isSolvable **do**
3: $\quad \boldsymbol{q}, \text{isSolvable} \leftarrow BothArmsIK\left(\boldsymbol{q}, \boldsymbol{p}^*_{\text{left}}, \boldsymbol{R}^*_{\text{left}}, \boldsymbol{p}^*_{\text{right}}, \boldsymbol{R}^*_{\text{right}}, \omega\right)$
4: $\quad \omega \leftarrow \omega + 1.0$
5: **end while**
6: $\boldsymbol{q}_{\text{chest}} \leftarrow \left(q_{\text{chest}_{\text{pitch}}}, q_{\text{chest}_{\text{yaw}}}\right) \in \boldsymbol{q}$

The numerical method of the inverse kinematics *BothArmsIK* with the initial joint angles $\boldsymbol{q}_0$ solves the joint angles in the upper body for the target pose of both arms. We use the Levenberg–Marquardt (LM) method [24]. In LM, ω represents the value of the diagonal matrix added to the diagonal elements of the Jacobian matrix. We improve the solvability of the inverse kinematics near the singularity points of the end-effector by trying to increase the ω value exploratory. The boolean value, isSolvable, indicates whether this inverse kinematics solution converges.

The numerical method of the inverse kinematics *BothArmsIK* (step 3 of Algorithm 2) with the initial joint angles $\boldsymbol{q}_0$ solves the joint angles in the upper body for the target pose of both arms using the Levenberg–Marquardt (LM) method [24]. In LM, ω represents the value of the diagonal matrix added to the diagonal elements of the Jacobian matrix. We improve the solvability of the inverse kinematics near the singularity points of the end-effector by increasing the ω value in a exploratory way. The boolean value *isSolvable* indicates whether this

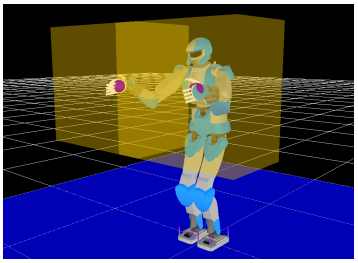


Figure 4. The initial posture of the humanoid robot for the learning procedure, with yellow boxes indicating the humanoid's workspace and purple spheres indicating the position of the end effector.

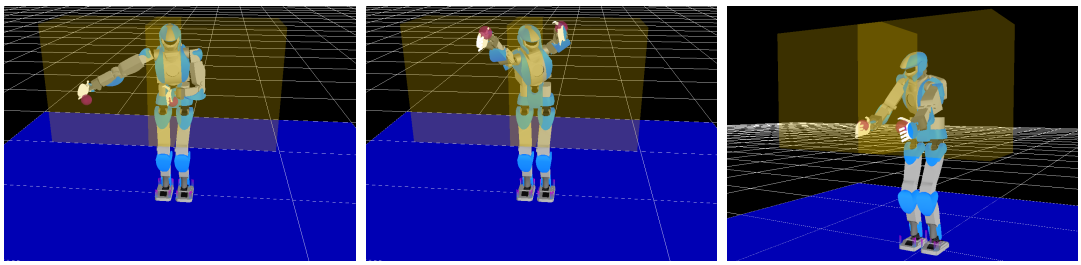


Figure 5. Humanoid's postures obtained from the SVR and the inverse kinematics of the arms.

Table 1. Parameters for the regression algorithm.

Parameter	Value
Kernel width	0.1
Constraint	1.0
Slack variable	0.0001

inverse kinematics solution converges.

We apply several target positions for testing the learned regression. Figure 5 shows the postures of the humanoid applied by the generated regression, and purple spheres indicate the target position of both hands as the input of the regression. Table 1 shows the three parameters we use for the regression algorithm.

4.2 Target CoM position generation

n a humanoid robot, changing the height of the body using their legs increases the manipulation area. We select the target position of the CoM of the whole body from three candidates considering the reachability of both arms for approaching the objects. Figure 6 shows the three candidates and the overlapping workspace in order to reduce the calculation cost. The robot can maintain balance by controlling the CoM in the quasi-static case.

We consider the reachability of humanoid robots using the position of both hands' targets ($\boldsymbol{p}^*$) and both shoulders ($\boldsymbol{p}_{\mathrm{shoulder}}$) in the initial configuration of the robot. For the reachability, the following condition should be satisfied:

$$|\boldsymbol{p}^* - \boldsymbol{p}_{\mathrm{shoulder}}| < L_{\mathrm{arm}}. \tag{5}$$

The length L_{arm} represents the length from the shoulder joint to the wrist. If each arm can not reach the target position in the current configuration, we set the target position of the CoM ($\boldsymbol{p}^*_{\mathrm{CoM}}$) as

$$\boldsymbol{p}^*_{\mathrm{CoM}} \leftarrow \begin{cases} \boldsymbol{p}_{\mathrm{CoM_{init}}} + \boldsymbol{b} & (p^*_z - p_{\mathrm{shoulder_z}} > L_{\mathrm{arm}}) \\ \boldsymbol{p}_{\mathrm{CoM_{init}}} - \boldsymbol{b} & (p^*_z - p_{\mathrm{shoulder_z}} < -L_{\mathrm{arm}}) \\ \boldsymbol{p}_{\mathrm{CoM_{init}}} & (\text{otherwise})\,. \end{cases} \tag{6}$$

A vector of the CoM position $\boldsymbol{p}_{\mathrm{CoM_{init}}}$ is calculated from the initial configuration of the robot, and a vector $\boldsymbol{b}$ is a pre-defined displacement $(0, 0, 0.1)$ in the coordinate system of the robot. If the target hand position is set to a location far from the shoulder, Equation 6 calculates the CoM position ($\boldsymbol{p}^*_{\mathrm{CoM}}$) in a higher or lower position. Note that the CoM trajectory is calculated by interpolating the initial and the target CoM positions.

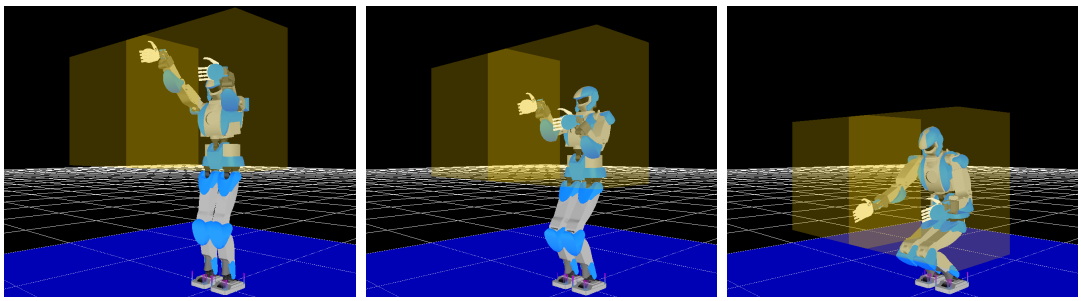


Figure 6. Movable areas (yellow boxs) of the proposed method in the three configuration of CoM.

Algorithm 3 Inverse Kinematics for both feet

Input: $\boldsymbol{q}_0, \boldsymbol{p}^*_{\mathrm{CoM}}, \boldsymbol{R}^*_{\mathrm{torso}}, \boldsymbol{p}^*_{\mathrm{swing}}, \boldsymbol{R}^*_{\mathrm{swing}}$

Output: $\boldsymbol{q}_{\mathrm{legs}}$

$\boldsymbol{q}_{\mathrm{legs}}, \boldsymbol{q}_{\mathrm{upperBody}} \leftarrow \boldsymbol{q}_0$

for $i = 0$ to N **do**

  $\boldsymbol{p}_{\mathrm{CoM}}, \boldsymbol{R}_{\mathrm{torso}}, \boldsymbol{p}_{\mathrm{swing}}, \boldsymbol{R}_{\mathrm{swing}}, \boldsymbol{J}$
  $\leftarrow \mathrm{FK}\left(\boldsymbol{q}_{\mathrm{legs}}, \boldsymbol{q}_{\mathrm{upperBody}}\right)$

  $\boldsymbol{e} \leftarrow \begin{pmatrix} \boldsymbol{p}^*_{\mathrm{CoM}} - \boldsymbol{p}_{\mathrm{CoM}} \\ \mathrm{LN}\left(\boldsymbol{R}^\top_{\mathrm{torso}} \boldsymbol{R}^*_{\mathrm{torso}}\right) \\ \boldsymbol{p}^*_{\mathrm{swing}} - \boldsymbol{p}_{\mathrm{swing}} \\ \mathrm{LN}\left(\boldsymbol{R}^\top_{\mathrm{swing}} \boldsymbol{R}^\star_{\mathrm{swing}}\right) \end{pmatrix}$

  if $|\boldsymbol{e}| < tol$ **then**

    break

  end if

  $\boldsymbol{q}'_{\mathrm{legs}} \leftarrow \boldsymbol{q}_{\mathrm{legs}} + \alpha \boldsymbol{J}^\sharp \boldsymbol{e}$

  $\boldsymbol{p}'_{\mathrm{CoM}}, \boldsymbol{R}'_{\mathrm{torso}}, \boldsymbol{p}'_{\mathrm{swing}}, \boldsymbol{R}'_{\mathrm{swing}}, \boldsymbol{J}'$
  $\leftarrow \mathrm{FK}\left(\boldsymbol{q}'_{\mathrm{legs}}, \boldsymbol{q}_{\mathrm{upperBody}}\right)$

  $\boldsymbol{e}' \leftarrow \begin{pmatrix} \mathrm{LN}\left(\boldsymbol{R}'^\top_{\mathrm{torso}} \boldsymbol{R}^*_{\mathrm{torso}}\right) \\ \boldsymbol{p}^*_{\mathrm{swing}} - \boldsymbol{p}'_{\mathrm{swing}} \\ \mathrm{LN}\left(\boldsymbol{R}'^\top_{\mathrm{swing}} \boldsymbol{R}^\star_{\mathrm{swing}}\right) \end{pmatrix}$

  $\boldsymbol{q}_{\mathrm{legs}} \leftarrow \boldsymbol{q}'_{\mathrm{legs}} + \beta \boldsymbol{J}'^\sharp \boldsymbol{e}'$

end for

4.3 Inverse kinematics for the lower body

We use Algorithm 3 to calculate the joint angles of both feet. We control the CoM of the whole body, the torso's orientation and the poses of the legs of the humanoid robot. The CoM of the whole body is calculated by assuming the configuration of the upper body (*e.g.*, the configuration in the previous time step). We use the numerical inverse kinematics to solve the joint angles of the lower part. Since the number of DoF of the lower body is smaller than the DoF of the whole body, the calculation is faster than the inverse kinematics of the whole body.

In Algorithm 3, we assume that the robot's sole of the support leg is on the

ground, and the coordinate system of the robot is in the support leg's sole. If the robot stands in both legs, we simply choose one of the legs as the support leg, and define the swing leg as the other leg. A vector $\boldsymbol{q}_{\text{legs}} = \left(\boldsymbol{q}_{\text{right}}, \boldsymbol{q}_{\text{left}}\right)$ represents the lower-body joint angles, that is, the joint angles of the right and left legs. A vector $\boldsymbol{q}_{\text{upperBody}}$ represents the upper-body joint angles, that is, the joint angles of both arms and the chest. The initial joint angles of the whole body are $\boldsymbol{q}_0$. $\boldsymbol{p}_{\text{CoM}}$ is the position of the CoM of the whole body (See Section 4.2), $\boldsymbol{p}_{\text{swing}}$ and $\boldsymbol{R}_{\text{swing}}$ are the position and the orientation of the foot sole in the swing leg, $\boldsymbol{R}_{\text{torso}}$ is the orientation of the torso link, and $\boldsymbol{J}$ is the Jacobian matrix of the lower body. The vector $\boldsymbol{x}^*$ is the target value of the value $\boldsymbol{x}$, and $\boldsymbol{Y}^\sharp$ is the pseudo-inverse of the matrix $\boldsymbol{Y}$. FK is a forward kinematics function. FK's input are the joint angles $\boldsymbol{q}$ and the output are the position $\boldsymbol{p}$, the orientation $\boldsymbol{R}$ (CoM, the torso and the swing leg) and the Jacobian of the lower body $\boldsymbol{J}$. The function LN converts the rotation matrix $\boldsymbol{R}$ to the angular vector [23]. The value N is the number of maximum iterations. The value *tol* is the tolerance for the norm of the error vector $\boldsymbol{e}$. The error vector has 12 dimensions that consist of the CoM positions, the torso orientation, and the position and orientation of the swing leg. Using the Jacobian $\boldsymbol{J}_{\text{CoM}}$, $\boldsymbol{J}_{\text{torso}}$ and $\boldsymbol{J}_{\text{swing}}$ for the CoM, the torso and the sole of the swing leg, the Jacobian can be represented as

$$\boldsymbol{J} = \begin{bmatrix} \boldsymbol{J}_{\text{CoM}} & \boldsymbol{J}_{\text{torso}} & \boldsymbol{J}_{\text{swing}} \end{bmatrix}^\top . \tag{7}$$

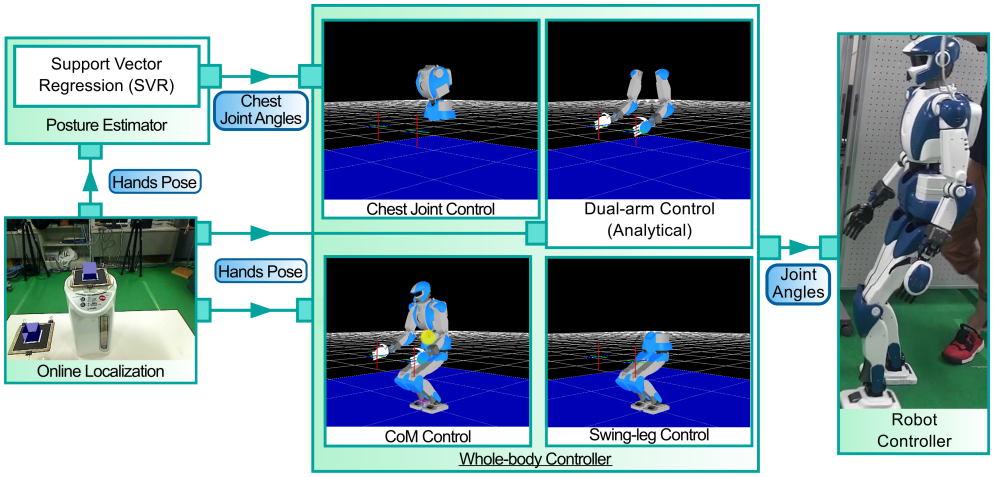


Figure 7. Overview of our system. Arrows indicate the data flow. The system is formed by the online localization, the posture estimator, the whole-body controller, and the robot controller. The environment recognizer estimates scene configurations using the AR markers. The posture estimator generates the target angles in the chest joints based on the target configuration of both hands. The whole-body controller is formed by four parts, the dual-arm controller, the lower-body controller, the CoM controller, and the swing leg controller. The dual-arm controller calculates the joint angles for both arms using the analytical solution of inverse kinematics. To control the coordination of the robot's height and maintain the robot's balance, the CoM controller and swing leg controller calculate the joint angles of both legs based on the desired whole CoM value. Finally, the robot follows the joint angles calculated by the whole body controller.

4.4 Calculation cost

We describe a mathematical ability of our proposed method using Landau symbol. We evaluate the computational burden of the proposed method compared to the conventional method. In the proposed method (Figure 7), there are two main processes, Dual-arm control which uses analytical Inverse Kinematics and Lower-body control which uses numerical Inverse Kinematics (including CoM Control and Swing-leg Control). The analytical Inverse Kinematics uses a simple linear algebra, *i.e.* that requires the constant calculation time. The numerical computation uses an inverse matrix calculation that requires calculation time with cubic of the matrix dimension. The computational complexity c_1 in the proposed method is described as below,

$$\begin{aligned} c_1(M) &= \mathcal{O}\left(1 + cM^3\right), \\ &= \mathcal{O}\left(cM^3\right). \end{aligned} \tag{8}$$

The number of the DoF is M, and the number of an iteration is c.

In the conventional method, the main computation is the inverse matrix calculation. The computational complexity c_2 in the conventional method is described as below,

$$c_2(N) = \mathcal{O}\left(dN^3\right). \tag{9}$$

The number of the DoF is N, and the number of an iteration is d.

In the proposed method, we use a machine learning for calculating the chest joint's angles, by calculating both arms and legs independently a kinematic chain of whole body can be divided on the chest joint. The number of the DoF (M) can write $\frac{N}{2}$, and the relationship is $M \simeq \frac{N}{2}$. We assume that the number of the iteration in both proposed and conventional numerical calculation is similar, and the relationship is $c \simeq d$. Comparing Equation 8 and 9, the calculation of the proposed method is roughly 8 times faster than the conventional method.

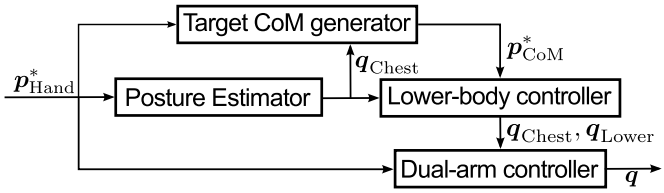


Figure 8. The data flow in both the posture estimator and the whole-body controller (the target CoM generator, the posture estimator, the lower-body controller, and the dual-arm controller).

4.5 Implementation

4.5.1 Overview of the manipulation system

Figure 7 shows the software structure of the system. The system is formed by four major parts: the online localization, the posture estimator, the whole-body controller, and the robot controller. In this figure, the arrows indicate the data flow. The object localization and recognition is out of the scope of this paper and we use AR markers for simplification. We use the ARToolKit library [21] to estimate the position of the AR marker and their ID's. Figure 8 shows the data flow between the posture estimator and elements of the whole-body controller. First, the target CoM generator determines the target CoM position corresponding to both hands at the target. After updating the value of the chest joint from the posture estimator, the lower-body controller calculates the joint angles in the chest and the legs' joints. Finally, the dual-arm controller calculates the joint angles of both arms.

4.5.2 Generating motion using inverse kinematics

To make a motion, we define the intermediate hand configurations calculated by linear interpolation between the initial value and the target value. The method of interpolation for the orientation is to multiply the initial rotation matrix and the rotation matrix calculated from the Rodrigues equation using a vector of the angular velocity from a difference between the target value and the initial value.

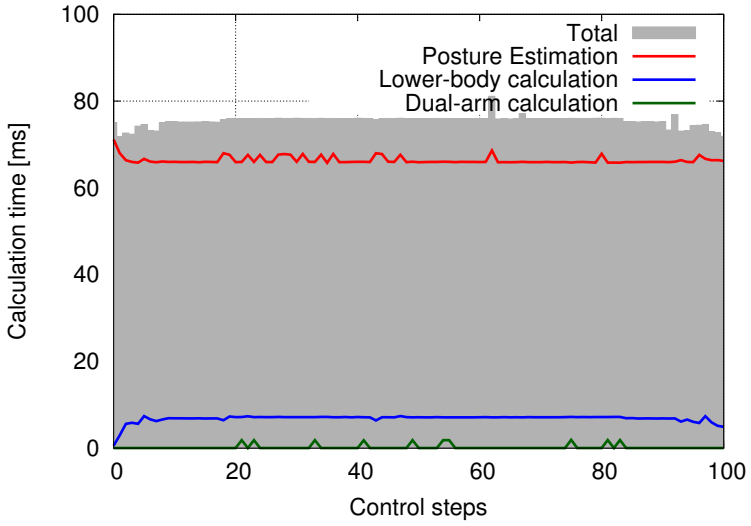


Figure 9. The calculation time of SVR, lower-body IK and both arms IK. The most part of the total calculation is the SVR calculation.

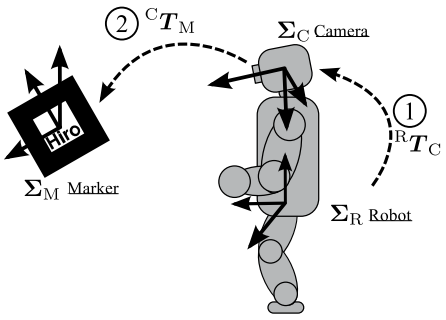


Figure 10. The relations between each coordinate system.

4.5.3 Posture estimator

We select the Gaussian kernel as the kernel function in the SVR. In this research, we use the regression once when inputting the target position of the end effectors and interpolate the results instead of calculating the regression in every control step. For the actual robot implementation, the robot's CPU (Intel Pentium M Processor 1.6 GHz) has a very low clock frequency, so it is slow to calculate the regression. Figure 9 shows the calculation times for the regression, the lower-body calculation, the analytical inverse kinematics for both arms and the total calculation time. The mean values of the calculation times in 100 samples are:

SVR, 66.4 ms; lower body, 6.8 ms; analytical inverse kinematics, 0.2 ms; and total, 75.5 ms. It turns out that the most time-consuming process is the SVR calculation.

4.5.4 Measuring the 3D position of each object

To manipulate an object located by the ARToolKit, we determine the hand pose by the following steps:

1. estimate the position of each object in the coordinate system of the camera mounted on the robot.

2. translate from the coordinate system of the camera to that of the robot.

Figure 10 shows the three coordinate systems of the robot $\boldsymbol{\Sigma}_{\mathrm{R}}$, the camera $\boldsymbol{\Sigma}_{\mathrm{C}}$, and the marker $\boldsymbol{\Sigma}_{\mathrm{M}}$. The matrices ${}^{\mathrm{R}}\boldsymbol{T}_{\mathrm{C}}$, ${}^{\mathrm{C}}\boldsymbol{T}_{\mathrm{M}}$, and ${}^{\mathrm{M}}\boldsymbol{T}_{\mathrm{O}}$ are homogeneous transformations from the coordinate system of the robot to the camera, from the camera to the marker, and from the marker to the object, respectively. The homogeneous transformation matrix from the coordinate system of the robot to the object is computed as

$$ {}^{\mathrm{R}}\boldsymbol{T}_{\mathrm{O}} = {}^{\mathrm{R}}\boldsymbol{T}_{\mathrm{C}}\ {}^{\mathrm{C}}\boldsymbol{T}_{\mathrm{M}}\ {}^{\mathrm{M}}\boldsymbol{T}_{\mathrm{O}}. \tag{10} $$

The matrix ${}^{\mathrm{R}}\boldsymbol{T}_{\mathrm{C}}$ is calculated using forward kinematics. We know ${}^{\mathrm{M}}\boldsymbol{T}_{\mathrm{O}}$ from the location of the marker on the object.

5. Experiments

5.1 Evaluation of a weighted pseudo-inverse matrix

5.1.1 Testbed

In this research, we used the human size humanoid robot HRP-4 shown in the robot controller part of Figure 7. This robot is produced by KAWADA Robotics Corporation [7]. The robot specifications are: 1.514 [m] height, 39 [kg] weight, and 34 DoF (each leg: 6, chest: 2, each arm: 7, each hand: 2, neck: 2). For the motion generated by the proposed inverse kinematics, we use 16 DoF (each arm: 7, chest: 2), and for grasping objects, we use 2 DoF in each finger. The HRP-4's camera is connected to its internal PC and has a resolution of two mega-pixels (1600×1200).

5.1.2 Objective

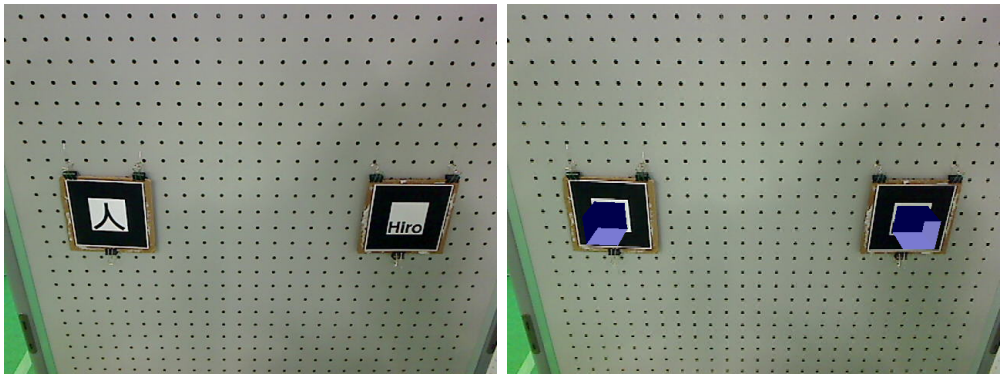
In this section, we describe two experiments:

- Vision-based manipulation of the HRP-4
- Motion generation using the manipulation system on a dynamic simulator

On the real robot HRP-4, we investigated the trajectory of the chest joint angle and the Zero Moment Point (ZMP) [27] to check the stability of the motion using this manipulation system. Then, to verify the movable area of the motion generator in detail, we used the dynamic simulator OpenHRP [28].

To control both hands simultaneously on the real robot, we implemented tracking of two target positions. As shown in Figure 12, there are two markers on a gray board and HRP-4 in the experimental space. In this experiment, HRP-4 moves both hands to the two markers on the board. We evaluated the effectiveness of the proposed method from the two aspects:

- the trajectory of the ZMP
- the movable area of the motion generated by inverse kinematics



(a) The image captured from HRP–4's camera

(b) Blue cubes rendered at the estimated positions

Figure 11. The results of estimated marker positions using the ARToolKit library.

Using a weighted pseudo-inverse matrix restricted the chest joint angle. This was proposed to improve the stability of the motion. To verify the stability, we used the stability index ZMP. We also verified whether the manipulation system could generate the motion needed to approach the object detected by a camera.

Figure 11 shows the images used for the proposed manipulation system. Figure 11 (a) shows the image as captured by the HRP-4 camera. Figure 11 (b) displays the estimated positions of the AR markers and rendered virtual objects. Using the estimated positions shown in Figure 11 (b), we generate the necessary motion.

We also showed the chest pitch joint angle and the ZMP in the frontal direction when controlling both hands to point at the AR markers. Since the chest joint of the humanoid robot can move the entire upper body, controlling the chest joint tends to affects the stability of the whole body. We compared the following two inverse kinematics to verify which motion is more stable:

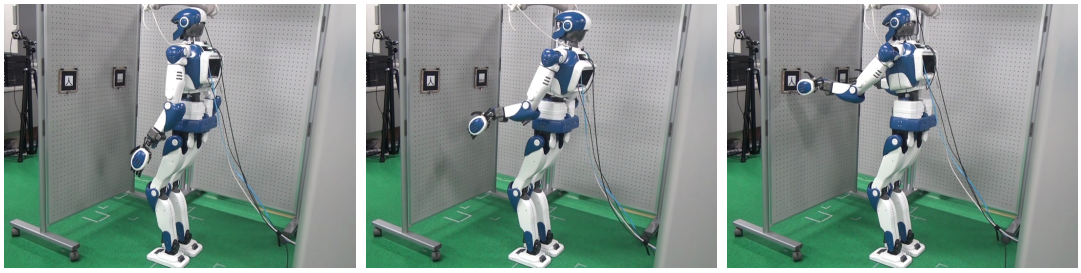
- using the standard pseudo-inverse matrix $\boldsymbol{J}_W$
- using the weighted pseudo-inverse matrix $\boldsymbol{J}_W^\sharp$

In the weighting matrix of Equation (3), the diagonal elements corresponding to the chest joint are equal to 0.1, and the other elements are equal to 1.0.

5.1.3 Result

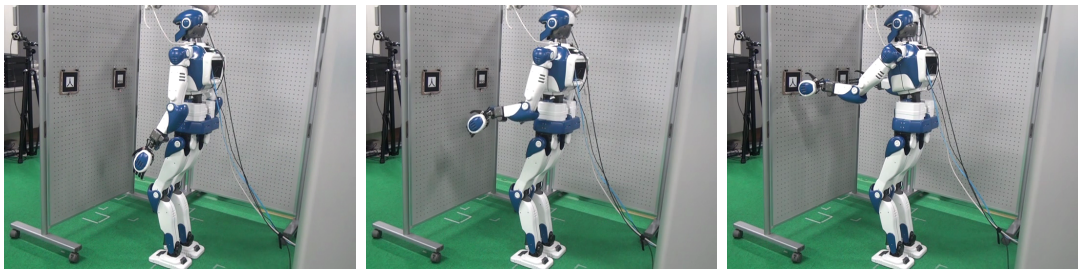
Figures 12 and 13 show the robot performing motions generated using the two method. As shown in Figure 12(c) and 13(c), the robot can move both two hands to the position of each marker. The axis of rotation in the frontal direction of HRP-4 chest joints is regarded as the y axis. When the chest joint is at zero rotation in the y axis, HRP-4 is in an upright stance. The origin of the ZMP in the sole of both feet is the middle point between each the ZMP. From Figures 12(c) and 13(c), there might seem to be no difference between the motion generated using the two methods. However, as can be seen by comparing Figures 12(b) and 13(b), when using the standard pseudo-inverse matrix in 12(b), the chest of HRP-4 seems to bend backward, which indicates a loss of stability.

Figure 14 shows comparisons of the trajectories of the chest pitch joint angle and the ZMP values using the two methods. Figure 14(a) shows a comparison



(a) Start movement: 0.0 s (b) 2.2 s (c) End movement: 3.9 s

Figure 12. Standard pseudo-inverse matrix controls HRP–4's hand to the target position of AR marker.



(a) Start movement: 0.0 s (b) 2.2 s (c) End movement: 3.4 s

Figure 13. Weighted pseudo-inverse matrix controls HRP–4's hand to the target position of AR marker.

of the trajectories of the chest y joint angle. Figure 14(b) shows a comparison of the trajectories of the ZMP in the frontal direction. In the chest y joint angle, a positive means the robot is bent forward and a negative means the robot is bent backward. In Figure 14(a), there is a large gap between the trajectories of the chest y joint produced using the two methods.

In Figure 14(b), there is also a small gap in the ZMP values in the frontal direction produced by the two methods. When the chest joint moves away from its initial value in the negative direction, the ZMP also moves away from its initial value in the negative direction. This is visible at 2.2 s in Figure 14(a) and 14(b). Using the standard pseudo-inverse matrix, in Figure 14 (b) from 1.0 s to 5.0 s, the trajectory of the ZMP vibrates because of large movement in the chest joint. On the other hand, using the weighted pseudo-inverse matrix in Figure 14 (b), the trajectory of the ZMP moves with stability to the converged value. Though HRP-4 is stable in an upright stance, and the motion is quasi-static, large values of motion of the ZMP may cause falling. The results shown in 14(a) and 14(b) support the stability of the motion generated using the weighted pseudo-inverse matrix.

In Figure 15, we generated six example motion with the dynamic simulator OpenHRP. In this experiment, we defined the motion as towards target positions and orientations from the HRP-4 state of Figure 12 (a):

(a) Rotation $\left(0.0, 0.0, \frac{\pi}{4}\right)$ rad

(b) Rotation $\left(0.0, 0.0, -\frac{\pi}{4}\right)$ rad

(c) Translation $(0.25, 0.0, 0.0)$ m

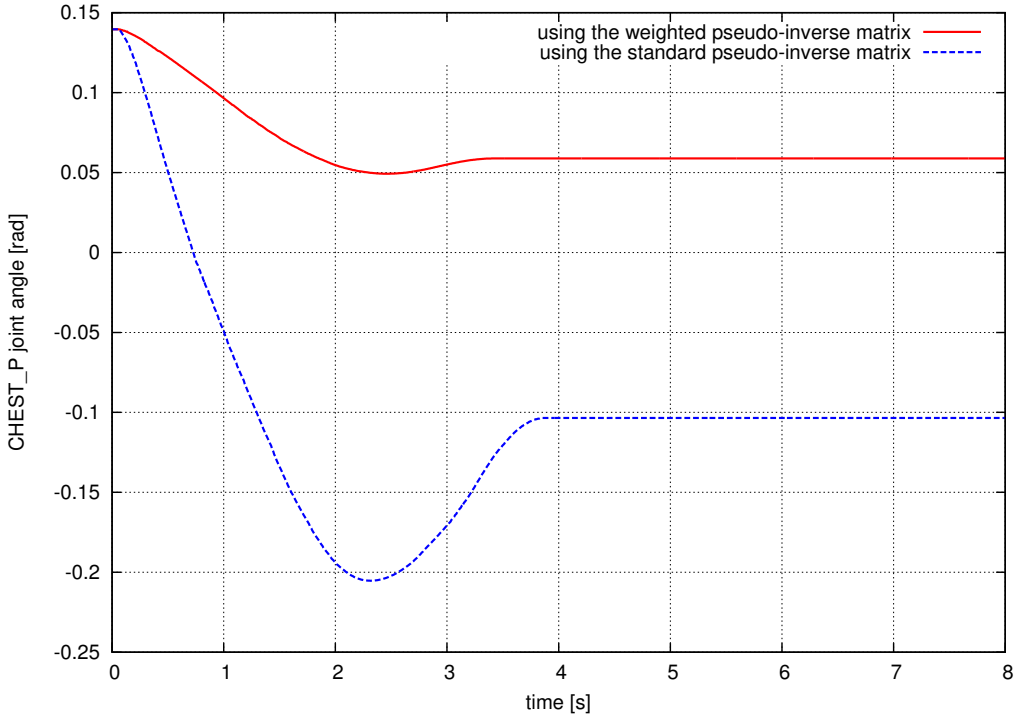
(d) Translation $(-0.25, 0.0, 0.0)$ m

(e) Translation $(0.0, 0.25, 0.0)$ m

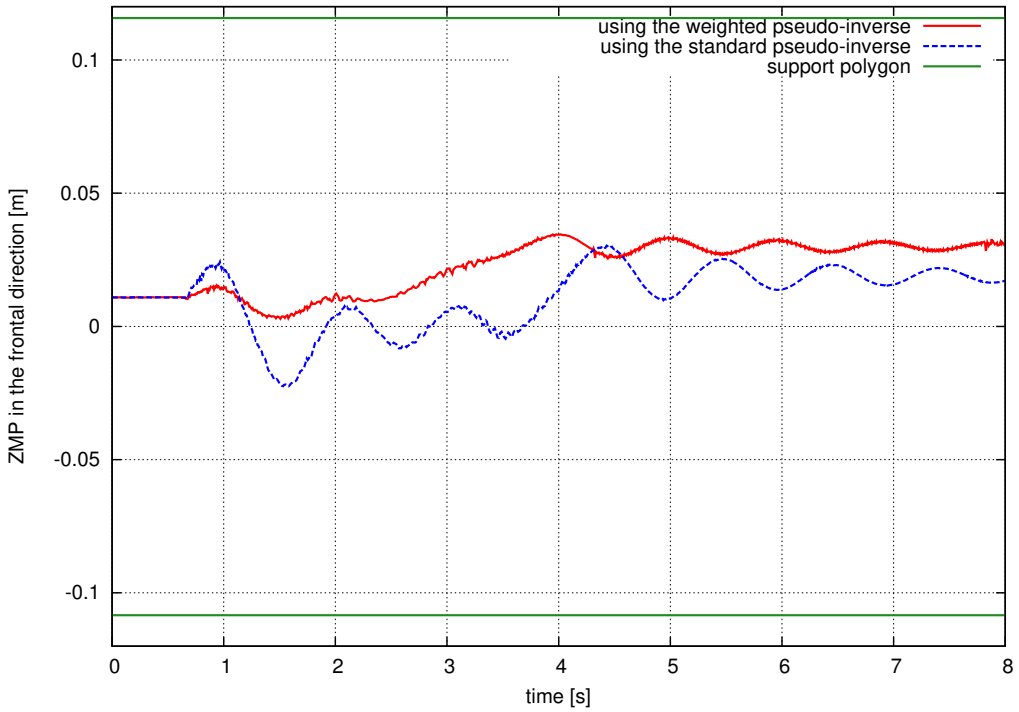
(f) Translation $(0.0, 0.0, -0.25)$ m

We select these six examples to test whether the humanoid robot can be in these postures. In Figure 15, the three color (red, green, and blue) lines represent the coordinate systems (x-, y-, and z-axis). As shown in Figure 15, we can check whether both the HRP-4's hands moved to the arbitrary target position in front

of its body. We made a trial of these motion samples within the range of positions and orientations needed to manipulate the objects. This simulation supports the effectiveness of the manipulation system.

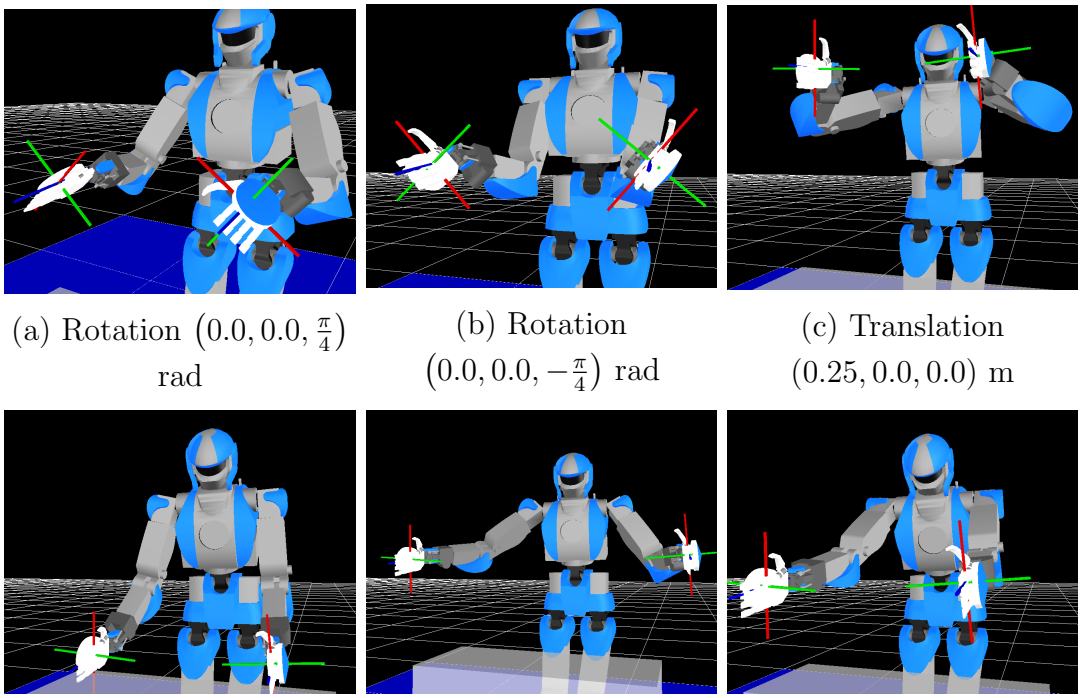


(a) The trajectory of the chest pitch joint angle



(b) The trajectory of the ZMP in the frontal direction

Figure 14. A comparison between the standard and weighted pseudo-inverse matrices.



(a) Rotation $\left(0.0, 0.0, \frac{\pi}{4}\right)$ rad

(b) Rotation $\left(0.0, 0.0, -\frac{\pi}{4}\right)$ rad

(c) Translation $(0.25, 0.0, 0.0)$ m

(d) Translation $(-0.25, 0.0, 0.0)$ m

(e) Translation $(0.0, 0.25, 0.0)$ m

(f) Translation $(0.0, 0.0, -0.25)$ m

Figure 15. Six examples of the target position of both hand using Inverse Kinematics with weighted pseudo-inverse matrix.

5.2 Evaluation of the whole-body controller

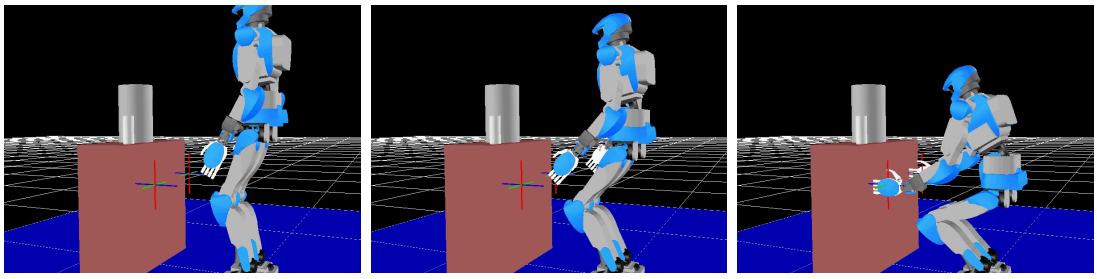
5.2.1 Comparing the calculation times using dynamics simulation

We show the effectiveness of the whole-body controller using the humanoid robot HRP-4 in the dynamics simulator OpenHRP [28]. We use the LM method for the numerical solution of inverse kinematics as a conventional method [24], as mentioned in Section 3. In the conventional method, the Jacobian matrix has 28 rows corresponding to the DoF and 24 columns corresponding to the position of both hands, the CoM, and the orientation of both hands, the torso, and the swing leg. The target position of both hands is in front of a table as shown in Figure 16.

Figure 16 shows the postures of the HRP-4: the initial posture (a), the interpolated posture (b), and the posture after the movement (c). We apply the proposed and conventional methods for 200 interpolated points from the initial to the final configurations. In the setting of Figure 16, the length between the initial configuration (a) and the final configuration (c) is 0.4 m, and the distance between each intermediate position is 1.9 mm. The number of the intermediate points are calculated from an acceptable speed of both of the end effector and CoM of 0.38 m/s, assuming that the control cycle is 5 ms.

Figure 17 shows the comparison of the calculation times. The mean of the calculation times using the proposed method is 1.8 ms and using the conventional method is 38.8 ms. Since the target position of both hands is further from the singularities than in the upper body experiments, the calculation time of the conventional method is smaller. We also verified that the calculation of the proposed method was faster than the conventional method in general. As shown in Figure 17, from control step 10 to 30, the conventional method rapidly calculates the joint angles. As can be seen in Figure 16, the postures (a) and (b) are very similar (near the half sitting pose), so the conventional controller only takes a short time. The maximum value of the calculation time is 9.5 ms in the proposed method, 74.8 ms in the conventional method. The proposed method is around eight times faster than the conventional method. This result proves the relationship between Equation 8 and 9 which express the computation order using Landau symbol.

Figure 18 shows the Zero Moment Point (ZMP) using a moving average filter



(a) Start movement (b) The interpolated posture (c) End movement

Figure 16. The posture of HRP-4 generated by the proposed method. HRP-4 completely reaches the target poses (three-colored frames) with the end effectors.

and the CoM position when moving to a hunkering-down posture. The upper blue line shows the position of the toe, and the lower blue line represents the heel. The area between the two lines represents the support region of the HRP-4's feet. The trajectory of the ZMP is always inside of the support region during this movement. This result shows that the movement generated by our method is well balanced. We also show the current and desired values of the CoM used in the proposed method. The trajectories of the CoM are the same during this movement, so the proposed controller is accurate.

We evaluate the accuracy of the proposed method with the error of the CoM position and the end effector, as shown in Figure 19. In this experiment, the velocity of the CoM is larger in the first half of the movement. Because of the CoM behavior, the error norm increases. In the second half, when approaching the target value, the velocity of the CoM decreases, and the CoM converges to the target value. The error of the end effector position is less than 1×10^{-15} m.

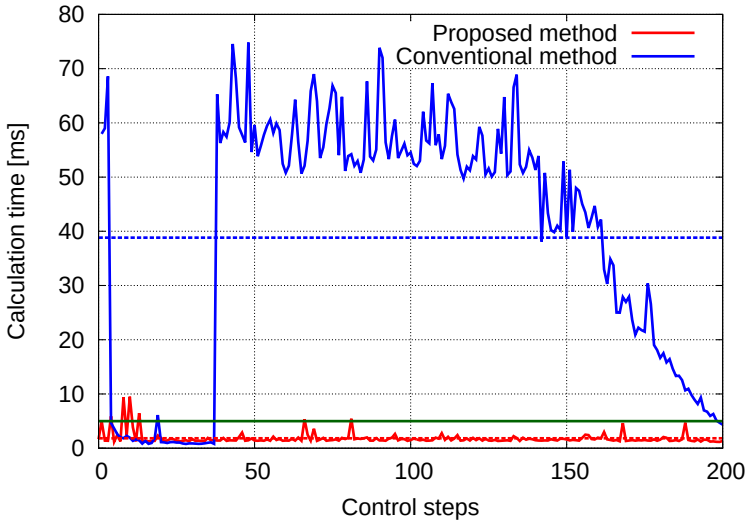


Figure 17. Comparison of the calculation times. In the control steps 10 to 30, the conventional method calculates the joint angles rapidly because the robot's pose is similar to the half sit pose.

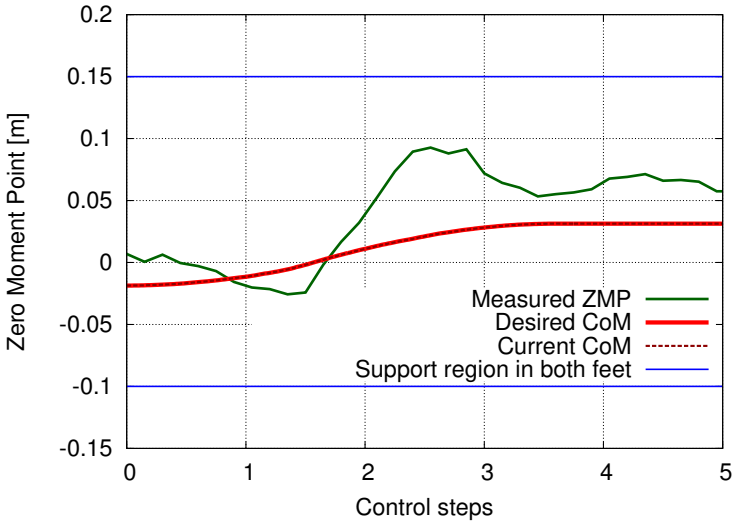


Figure 18. The ZMP of both feet and CoM during the whole-body movement. The support region between two blue lines represents the area from toe to heel.

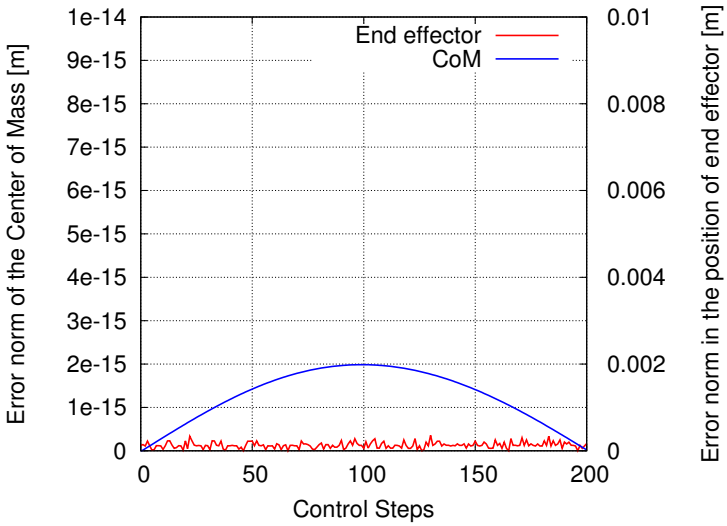
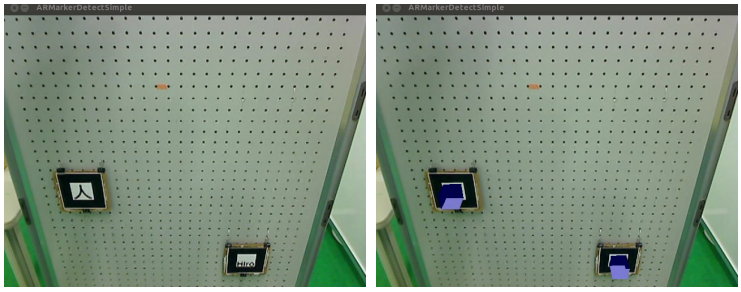


Figure 19. The error norm of the CoM position and the end effector during the whole-body movement.

5.2.2 Real-robot experiments

We also verify the effectiveness of the proposed whole-body controller using an actual HRP-4. The target positions of both hands are obtained from the AR markers, which are observed from the camera mounted on the head of the HRP-4. Figure 20 shows the images captured from the HRP-4's camera and processed by the ARToolKit.

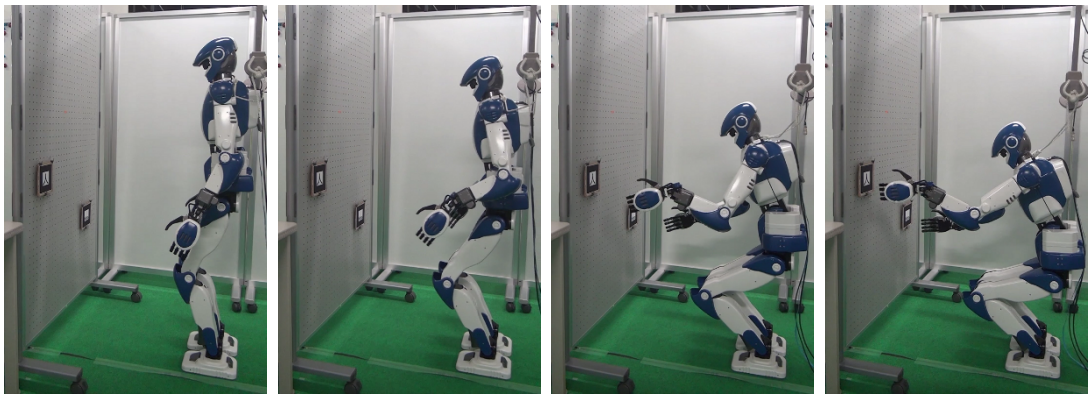
Figure 21 shows the generated movement of the HRP-4 in the initial posture (a), the interpolated postures (b) and (c), and the posture after the movement (d). The HRP-4 reaches the two AR markers with both hands, while maintaining balance. This movement takes 3 s.



(a) The input image from HRP-4's camera

(b) Rendered blue cubes at the estimated position

Figure 20. The result of the marker positions estimated using the ARToolKit library (left: the input image from HRP-4's camera, right: rendered blue cubes at the estimated position).



(a) Initial state: 0 s (b) 1 s (c) 2 s (d) Final state: 3 s

Figure 21. The proposed method generates whole-body motion to make the HRP-4 reach with both hands the target positions indicated with AR markers. By managing the CoM in the center of the foot's soles, the robot can keep the balance while hunkering down.

5.2.3 Accuracy and calculation speed

We evaluate the applicability of the proposed method for the whole-body movement by using a numerical method as conventional method. We test both methods with 729 (27 poses for each hand) random target poses in the workspace, as shown in the Figure 5. Table 2 shows the results using both methods for all target poses. The error norm is the average norm of the difference vector between the target pose and the current pose. Calculation time is the average time to calculate the whole-joint angles at each control step. Total time is the average time for moving the whole body in the dynamics simulator.

The results of the error norm and the calculation time are better than with the conventional method. The error norm using the conventional method is large when the target position are the far side in the lateral direction. The conventional method can solve the whole-joint angles including the chest joint but it is difficult to reduce the error norm with the torso posture. The proposed method is faster in such cases because it uses the train data for the calculation of the torso posture. The total movement using the proposed method is around 2.6 times faster than the conventional method.

Table 2. Comparison of results in both methods.

	Error norm	Calculation time	Total time
Proposed	0.62 mm	1.1 ms	1.0 s
Conventional	9.99 mm	46.4 ms	2.6 s

5.2.4 Comparing the accuracy to a conventional method

For the evaluation of the accuracy in the proposed method, we compare our method to a conventional method using a same target pose. We use the inverse kinematics in the whole body motion generation for the humanoid robot conducted by Caron *et al.* [29] as the conventional method which proposed in two years. Figure 22 shows the error of the CoM and both hand's positions comparing to the each target position. We configure the 200 interpolated position between the initial and the target position as same as the proposed method. In the conventional method , the mean of the CoM's error is 1.4×10^{-3}m, the mean of both hands' position are 3.8×10^{-4}m (in the left hand), and 4.0×10^{-4}m (in the right hand). In the proposed method , the mean of the CoM's error is 5.1×10^{-5}m, the mean of both hands' position are 4.9×10^{-5}m. Comparing these two result using both method, our proposed method can perform more precise control than the conventional method.

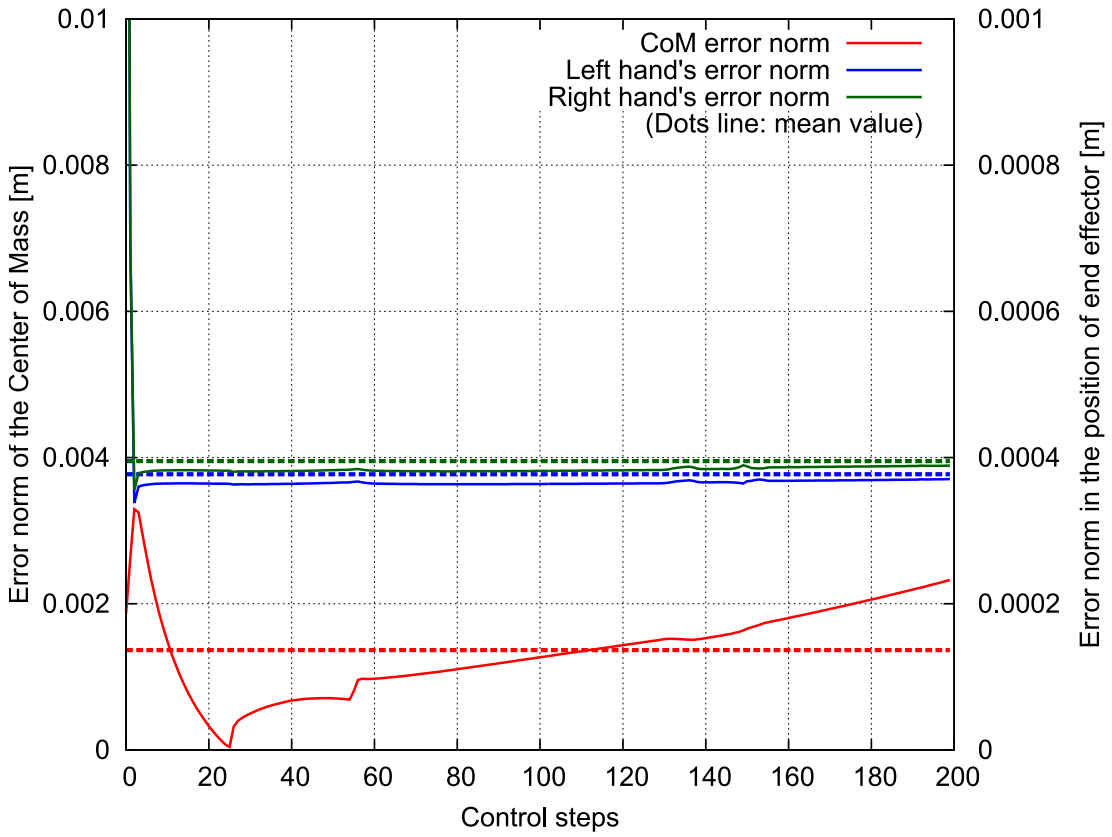


Figure 22. The error norm of the CoM position and the end effector during the whole-body movement using a conventional method.

5.3 Following markers

We apply the proposed method to the HRP-4 in the dynamics simulator for following the AR markers. Figure 23 shows the poses of the humanoid robot in following the markers. HRP-4 can follow the every marker pose captured by the camera mounted on the head.

5.4 Sequential whole-body control

We apply the proposed method to the HRP-4 in the dynamics simulator for approaching several positions sequentially. Figure 24 shows the poses of the humanoid robot approaching target positions generated randomly. The three color arrows in Figure 24 indicate the target positions inputted to the HRP-4. In this figure, the HRP-4 successfully reaches three target positions using its whole body, while maintaining balance. Figure 24 (a) depicts the motion of the HRP-4 to reach target positions above the humanoid's head. Figure 24 (b) and (c) show cases where one of the hands is reaching a high position and the other is reaching a low position. These figures and verify the applicability of the proposed method to whole-body control with sequentially random inputs.

In Figure 6, we can see the movable area using the proposed method. Comparing both of Figure 6 and 24, large workspace and the applicability of the proposed method is verified.

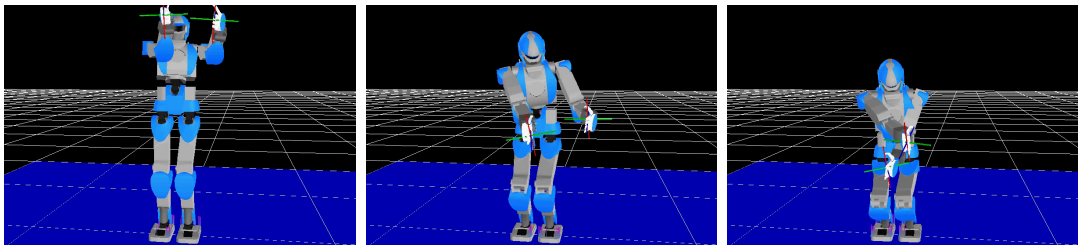


Figure 23. From a camera input, HRP-4 can follow the target position of AR markers using the proposed method.

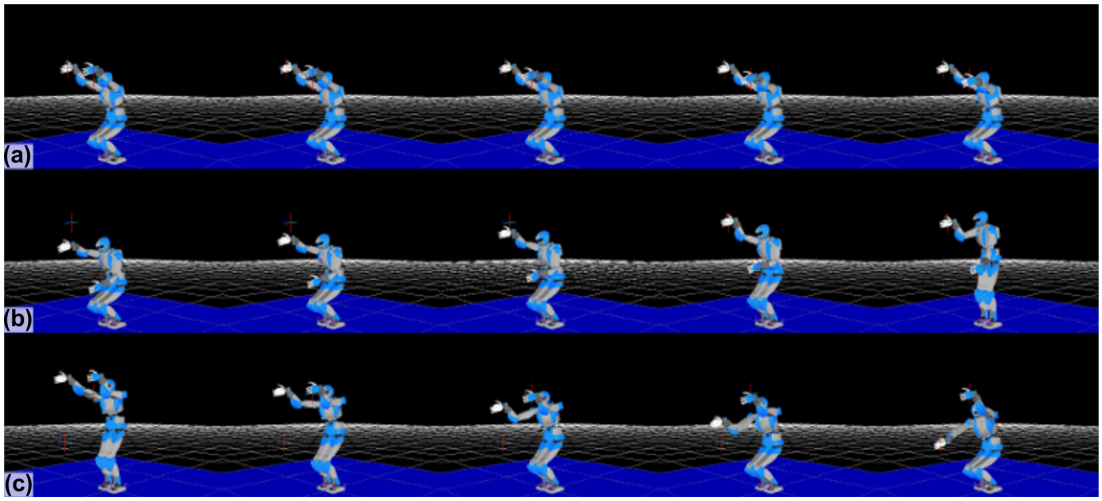


Figure 24. The simulation result when approaching random target positions. These movements include the whole-body motion in the vertical direction and the arms in the vertical and diagonal direction.

5.5 Application to a living environment

We also apply the proposed method to a human living environment using a dynamics simulator. Figure 25 shows the pose of the humanoid robot approaching two target positions on a table and a shelf. In this experiment, we solve the collision avoidance issue by assigning waypoints to the hands' trajectories. Figure 25 (a) shows the initial pose of the HRP-4. Figures 25 (c) and (f) show the HRP-4 approaching the objects with both hands. In Figure 25 (d), the HRP-4 is moving to the next target position on the upper shelf.

It takes 3 s to approach the position on the table and 5 s to approach the position on the upper shelf. Because we set the waypoints for each target configuration, it takes longer to approach the position than using the motion planning without a waypoint. Multiple movements including chest rotation and squatting are successfully generated using the proposed method, and the time of the movement is reasonable for users.

Our proposed method uses only the CoM for keeping the robot's balance. Using our method, the robot can not reach an edge of the ZMP's support polygon with a fast whole-body movement. However, when the robot's balance is close

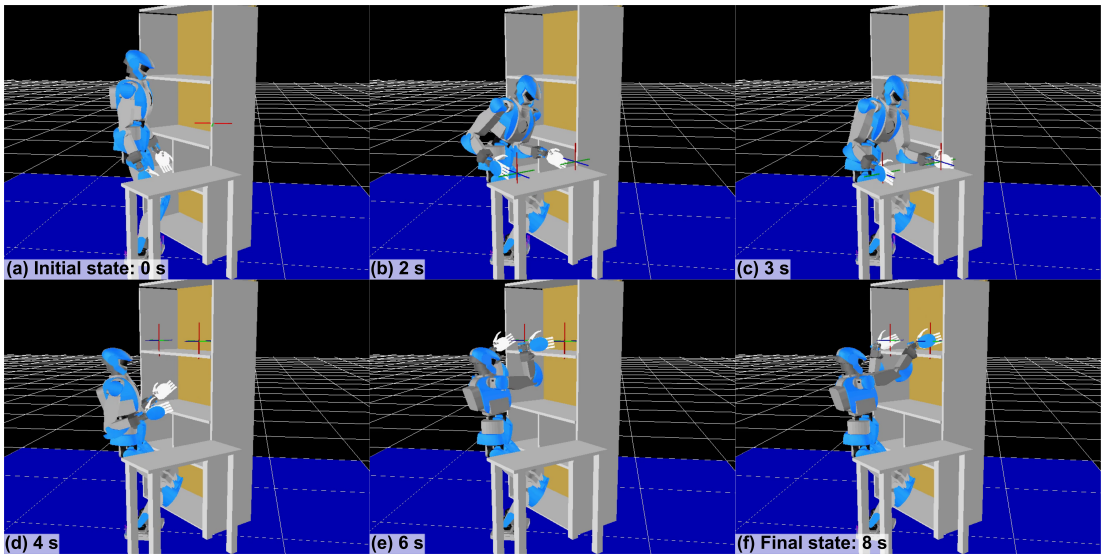


Figure 25. The simulation results when approaching both hands to target positions in a living environment. There is a table and a shelf in the dynamics simulation environment. To avoid collisions, we set waypoints when approaching the target positions of each object (parts (b) and (e) in this figure). First, the HRP-4 approaches the target objects on the table. Then, after approaching, the HRP-4 moves to the target objects on the shelf.

to this edge, the robot should move slowly in the last part of the motion. In the case of our proposed method, we should add a threshold of a horizontal CoM to avoid falling down. After approaching this threshold, the robot should move slowly. Another option is including a locomotion for approaching the whole body out of this threshold to our proposed method.

6. Conclusion

6.1 Contribution

We proposed a method of whole-body motion generation to reach objects in real time. First, we proposed a system for controlling humanoid robots to allow manipulation of objects with the necessary quickness for daily life environment. We constructed a manipulation system on the middleware OpenRTM that could control both hands of HRP-4 in moving to positions detected by ARToolKit. Using a weighted pseudo-inverse matrix corresponding to each joint in inverse kinematics, we implemented stable control of both hands. We verified the effectiveness of the weighted pseudo-inverse matrix in achieving stability by evaluating the trajectories of the ZMP during the motion. Second, we proposed a method for whole-body motion generation for humanoid robots in household environments to reach objects with their hands in real time. To reduce the calculation time of motion generation, we estimated the torso posture using SVR before calculating the joint angles. Using the estimated torso posture, we can separate the IK for the whole body into simpler, independent IK for the arms and the lower body. We solve the IK of both arms analytically and solve the IK of the lower body numerically. To sequentially reach multiple targets for the hands, we built a target position generator for the CoM considering the reachability of both arms.

In terms of calculation time, the proposed method is faster than the conventional Jacobian-based numerical method, while achieving better accuracy. To verify the effectiveness of the proposed method, we implemented it on a human-sized humanoid robot, the HRP-4. In experiments, we successfully generated reaching motions for lower positions, which included hunkering motion while keeping the balance. In these experiments, it took 3 s to both generate and execute the target motion. We also showed the applicability of the proposed method to a step-by-step manipulation by sequentially inputting random targets. In experiments using a dynamics simulation, HRP-4 could approach the target positions smoothly. Finally, we also applied the proposed method to a kitchen environment, by generating a whole-body motion to approach a table and an upper shelf.

6.2 Future work

In this dissertation, we only generated motions when the humanoid's feet were fixed. Since the proposed lower-body controller manages the configuration of the swing leg, the proposed method has the potential to expand the reachable area by stepping. Further, in the posture estimation, we employed SVR as the regression method to simplify implementation, but we could investigate using other regression methods such as random forest regression and neural networks.

Though we did not test our method with the inertial changes present when grasping an object, we are planning to do it as future work. The proposed method should be able to adapt to these changes if the physical properties of the grasped object are known *a priori*.

Acknowledgements

I would like to express my special appreciation and thanks to my advisor Professor Tsukasa Ogasawara, I got a lot of kind helps and comments for this dissertation, my student life, and the way to be a researcher. I got a lot of skills to consider and proceed the research by his giving me many comments for the discussion, my paper works, and presentation.

I would also like to thank my committee member Professor Kenji Sugimoto for serving as my committee member even at hardship. I also want to thank you for letting my defense be an enjoyable moment, and for your brilliant comments and suggestions, thanks to you.

I really appreciate to Professor Jun Takamatsu, he kindly gave me a lot of the opportunity for discussion and helps. Particularly, I got many comments for my paper works in master and doctoral student. His direction and vitality to conduct research most effect on my style and thinking of research. Without his help, I can not conclude this dissertation.

Particuler thank you go to Professor Etsuko Ueda, Associate Professor Masanao Koeda, Assistant Professor Kentaro Takemura, Assistant Professor Atsutoshi Ikeda, Assistant Professor Akihiko Yamaguchi, Assistant Professor Ming Ding.

In my student life, I luckily got a lot of opportunity to discuss and talk with many friends. Particularly, my thinking the research and the life grows by talking with Assistant Professor Gustavo Alfonso Garcia Ricardez and Assistant Professor Daisuke Tanaka. Gustavo and I have been a friend for 7 years since I have met from my master student, and he kindly gave me many helps for the research, life, thinking, and so on. Luckily, I got many and many times to get an English proofreading from him, most of my English documents in NAIST get his comments, and a little Spanish. I talked a lot about how to make a result better and consider each failure with Daisuke. Our talking tends to be longer for no particular reason, but I learned a lot from this conversation.

A special thanks to my family. I would also like to thank all of my friends who supported me in writing, and incented me to strive towards my goal.

Publications

Peer-reviewed International Journal

1. Satoki Tsuichihara, Yuya Hakamata, Gustavo Alfonso Garcia Ricardez, Jun Takamatsu, and Tsukasa Ogasawara, Real-time whole-body motion generation using torso posture regression and center of mass, ROBOMECH Journal, 10.1186/s40648-018-0105-y, 2018 (in press, accepted on 16 April 2018).

Peer-reviewed International Conference Paper

1. Satoki Tsuichihara, Yuya Hakamata, Gustavo Alfonso Garcia Ricardez, Jun Takamatsu, and Tsukasa Ogasawara, Accelerating Whole-body Motion Generation Using Regression of Humanoid ' s Torso Posture, In Proceedings of the 16th IEEE-RAS International Conference on Humanoid Robots (Humanoids), pp.16-21, 2016.

2. Satoki Tsuichihara, Akihiko Yamaguchi, Jun Takamatsu, and Tsukasa Ogasawara, Using a Weighted Pseudo-Inverse Matrix to Generate Upper Body Motion for a Humanoid Robot Doing Household Tasks, In Proceedings of 2015 IEEE International Conference on Robotics and Biomimetics (ROBIO), pp.333-338, 2015.

3. Satoki Tsuichihara, Masanao Koeda, Seiji Sugiyama, and Tsuneo Yoshikawa, A sliding walk method for humanoid robots using ZMP feedback control, In Proceedings of 2011 IEEE International Conference on Robotics and Biomimetics (ROBIO), pp.275-280, 2011.

Peer-reviewed Doomestic Conference Paper

1. 築地原 里樹, 山口 明彦, 吉川 雅博, 高松 淳, 小笠原 司: ヒューマノイドロボットにおける重み付きヤコビアンを用いた視覚に基づく上半身動作の生成, 第 20 回ロボティクスシンポジア, 2C2, 2015.

Domestic Conference Paper

1. 築地原 里樹, 袴田 有哉, Garcia Ricardez Gustavo Alfonso, 高松淳, 小笠原司: 胴体姿勢回帰と重心補正を用いた高速な全身運動生成, ロボティクスメカトロニクス講演会 2018 (ROBOMECH2018), 2A2-I07, 2018.

2. 築地原 里樹, 袴田 有哉, Garcia Ricardez Gustavo Alfonso, 高松淳, 小笠原司: 胴体姿勢推定を用いた高速なヒューマノイドロボットの全身動作生成, ロボティクスメカトロニクス講演会 2017 (ROBOMECH2017), 1P2-N07, 2017.

3. 築地原 里樹, 袴田 有哉, 高松 淳, 小笠原 司: ヒューマノイドロボットの胴体姿勢推定を用いた全身動作生成, 第 34 回日本ロボット学会学術講演会 (RSJ2016), 3F1-04, 2016.

4. 築地原 里樹, 袴田 有哉, 高松 淳, 小笠原 司: 胴体姿勢推定を含んだ高速なヒューマノイドロボットの上体動作生成, ロボティクスメカトロニクス講演会 2016 (ROBOMECH2016), 2A1-10a5, 2016.

5. 築地原 里樹, 山口 明彦, 吉川 雅博, 高松 淳, 小笠原 司: 等身大ヒューマノイドロボットによる給仕動作の実現と実行時間に対する考察, 第 32 回日本ロボット学会学術講演会 (RSJ2014), 1C3-02, 2014.

6. 築地原 里樹, 山口 明彦, 高松 淳, 小笠原 司: ヒューマノイドロボットのステッピングを含む入力動作における転倒回避手法, 第 31 回日本ロボット学会学術講演会, 1K3-07, 2013.

7. 築地原 里樹, 杉山 正治, 小枝 正直, 吉川 恒夫: ZMP フィードバック制御によるヒューマノイドロボットの摺足歩行, 日本機械学会ロボティクス・メカトロニクス講演会 2011(ROBOMEC2011), 2P2-J01, 2011.

References

[1] IRobot, "Roomba Robot Vacuum," http://www.irobot.com/For-the-Home/Vacuuming/Roomba.aspx.

[2] Moley, "The world's first robotic kitchen," http://www.moley.com. [Online]. Available: http://www.moley.com

[3] M. Hirose and K. Ogawa, "Honda humanoid robots development." *Philosiphical Transactions of the Royal Society*, vol. 365, pp. 11–19, 2006.

[4] C. Ye, Y. Yang, C. Fermüller, and Y. Aloimonos, "What Can I Do Around Here? Deep Functional Scene Understanding for Cognitive Robots," *CoRR*, vol. abs/1602.0, 2016.

[5] J. Kuffner, K. Nishiwaki, S. Kagami, M. Inaba, and H. Inoue, "Motion planning for humanoid robots," in *International Symposium of Robotics Research (ISRR)*, pp. 365–374, 2005.

[6] M. Cognetti, V. Fioretti, and G. Oriolo, "Whole-body planning for humanoids along deformable tasks," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1615–1620, 2016.

[7] K. Kaneko, F. Kanehiro, M. Morisawa, K. Akachi, G. Miyamori, A. Hayashi, and N. Kanehira, "Humanoid robot HRP-4 –Humanoid robotics platform with lightweight and slim body–," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pp. 4400–4407, 2011.

[8] B. Henze, M. A. Roa, and C. Ott, "Passivity-based whole-body balancing for torque-controlled humanoid robots in multi-contact scenarios," *The International Journal of Robotics Research*, vol. 35, no. 12, pp. 1522–1543, 2016.

[9] S. Nozawa, M. Kanazawa, Y. Kakiuchi, K. Okada, T. Yoshiike, and M. Inaba, "Three-dimensional humanoid motion planning using COM feasible region and its application to ladder climbing tasks," in *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pp. 49–56, 2016.

[10] J. Vannoy and J. Xiao, "Real-time adaptive motion planning (RAMP) of mobile manipulators in dynamic environments with unforeseen changes," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 1199–1212, 2008.

[11] C. Park, J. Pan, and D. Manocha, "ITOMP: Incremental trajectory optimization for real-time replanning in dynamic environments," *Proceedings of the International Conference on Automated Planning and Scheduling*, no. May 2016, pp. 207–215, 2012.

[12] M. Otte and E. Frazzoli, "RRT X: Real-Time Motion Planning / Replanning for Environments with Unpredictable Obstacles," in *Algorithmic Foundations of Robotics XI: Selected Contributions of the Eleventh International Workshop on the Algorithmic Foundations of Robotics*, 2015, pp. 461–478.

[13] C.-L. Fok, G. Johnson, L. Sentis, A. Mok, and J. D. Yamokoski, "ControlIt! - A Software Framework for Whole-Body Operational Space Control," *International Journal of Humanoid Robotics*, vol. 13, no. 01, p. 1550040, 2015.

[14] K. Nishiwaki, M. Kuga, S. Kagami, M. Inaba, and H. Inoue, "Whole-body cooperative balanced motion generation for reaching," *International Journal of Humanoid Robotics*, vol. 2, no. 4, pp. 437–457, 2005.

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

[16] P. Ferrari, M. Cognetti, and G. Oriolo, "Humanoid Whole-Body Planning for Loco-Manipulation Tasks," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4741–4746, 2017.

[17] M. Cognetti, P. Mohammadi, and G. Oriolo, "Whole-body motion planning for humanoids based on CoM movement primitives," in *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pp. 1090–1095, 2015.

[18] O. Khatib, "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots." *International Journal of Robotics Research*, vol. 5, no. 1, pp. 90–98, 1986.

[19] M. Zucker, N. Ratliff, A. D. Dragan, M. Pivtoraiko, M. Klingensmith, C. Dellin, J. A. Bagnell, and S. Srinivasa, "CHOMP: Covariant Hamiltonian optimization for motion planning," *International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1164–1193, 2013.

[20] P.-C. Yang, K. Sasaki, K. Suzuki, K. Kase, S. Sugano, and T. Ogata, "Repeatable Folding Task by Humanoid Robot Worker using Deep Learning," *IEEE Robotics and Automation Letters (RA-L)*, vol. 2, no. 2, pp. 397–403, 2016.

[21] K. Hirokazu and M. Billinghurst, "Marker tracking and HMD calibration for a video-based augmented reality conferencing system," in *IEEE and ACM International Workshop on Augmented Reality (IWAR)*, ser. IWAR '99, pp. 85–94, 1999.

[22] S. Kajita, H. Hirohisa, H. Kensuke, and Y. Kazuhito, *Introduction to Humanoid Robotics*, 2014.

[23] R. M. Murray, S. S. Sastry, and L. Zexiang, *A Mathematical Introduction to Robotic Manipulation*, 1st ed. Boca Raton, FL, USA: CRC Press, Inc., 1994.

[24] T. Sugihara, "Solvability-Unconcerned Inverse Kinematics by the Levenberg-Marquardt Method," *IEEE Transaction on Robotics*, vol. 27, no. 5, pp. 984–991, 2011.

[25] N. Ando, T. Suehiro, K. Kitagaki, T. Kotoku, and W. K. Yoon, "RT-Middleware: Distributed component middleware for RT (Robot Technology)," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3555–3560, 2005.

[26] A. J. Smola and B. Schölkopf, "A Tutorial on Support Vector Regression," *Statistics and Computing*, vol. 14, no. 3, pp. 199–222, 2004.

[27] M. Vukobratović and B. Borovac, "Zero-Moment Point — Thirty Five Years of Its Life," *International Journal of Humanoid Robotics*, vol. 01, no. 01, pp. 157–173, mar 2004. [Online]. Available: http://www.worldscientific.com/doi/abs/10.1142/S0219843604000083

[28] F. Kanehiro, H. Hirukawa, and S. Kajita, "OpenHRP: Open Architecture Humanoid Robotics Platform," *International Journal of Robotics Research*, vol. 123, no. 2, pp. 155–165, 2004.

[29] S. Caron and A. Kheddar, "Multi-contact Walking Pattern Generation based on Model Preview Control of 3D COM Accelerations," in *IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, pp. 550–557, 2016.