Generating persistently exciting trajectory based on condition number optimization

Ko Ayusawa[1], Antoine Rioux[1], Eiichi Yoshida[1], Gentiane Venture[1,2] and Maxime Gautier[3]

***Abstract*— This paper presents a novel optimization method for generating persistently exciting trajectories for inertial parameters identification of a robot. The exciting performance of the trajectories is usually evaluated by the condition number of the regressor matrix, which appears in the linear regression model for identification. In this paper, the efficient formulation is presented to directly compute the gradient of the condition number with respect to joint trajectory parameters, by deriving the derivative of the singular values and regressor matrices. Direct gradient computation can enhance computational performance of optimization, which is essential for large DOF systems under many physical consistent conditions such as humanoid robots. The proposed method is validated by generating several trajectories for the humanoid robot HRP-4.**

I. Introduction

Owing to the recent technical advancement of humanoid robots, they are now expected to be applied to various fields like disaster response, human entertainment, and as a scientific tool for studying human behaviors. Such applications usually require the development of efficient whole body controllers or accurate simulators for the validation of motions. The knowledge about the system dynamics is essential in order to meet such requirements.

When modeling the dynamics of a robot, its geometric and inertial parameters need to be known. The geometric parameters such as joint locations and orientations are available from manufacturers, because they are usually determined when designing a robot. While the inertial parameters, typically mass, center of mass, and inertia tensors of each link can be estimated from CAD data, they are often inaccurate and subject to changes. It is difficult to determine the physical properties of the electronic cables or components inside a mechanical gear, and the modifications of hardware usually affect those parameters. For this reason, the process of identifying inertial parameters is crucial for obtaining an accurate dynamic model.

Methodologies for identifying inertial parameters of a robot has been extensively studied [1]. Though system identification itself is widely studied in many other fields, in robotics, it utilizes the linearity of inertial parameters in the equation of motion of a robot [2], [3]. Thanks to this convenient property, identification methods have been developed for many robotics applications [4], [5], [6]. More recently, identification for humanoid systems has attracted attention, and some methods specialized for humanoid systems have also been reported [7], [8], [9].

One of the most important processes for identification is how to generate appropriate joint trajectories, which are called persistently exciting trajectories (PE trajectories). Efficient techniques based on optimization to find the trajectories for robotics manipulators are detailed in [10], [11], [12], [13]. Yet, humanoid robots differ in several points from fixed manipulator robots. The most critical difference is the inexistence of a fixed base in humanoid robots (i.e., they have a floating base). It requires additional consideration about balancing conditions, even before identification. While the recent optimization-based approaches are expected to be useful for handling such constraints [14], the large number of degree of freedom (DOF) also leads to computational complexity. For those reasons, popular approaches [15], [16] are such that, at first, several physically possible motions are generated in advance, and then, the optimal set of motions are selected from those candidates. Though it is simple and convenient, many motion candidates need to be generated without guarantee to obtain optimal PE trajectories. Accordingly, an efficient method to find the optimal PE trajectories is needed.

In this paper, we propose a novel efficient optimization algorithm to find PE trajectories. The highlighted feature of our method is the analytical formulation of the gradient of the particular cost function in our optimization problem. The condition number of the regressor matrix is utilized as the cost function of optimization. We formulate the gradient computation of the condition number with respect to the joint trajectory parameters. It enables direct optimization of trajectory under several constraints such that joint limits, self-collisions, and balancing based on a-priori CAD data. We will demonstrate the proposed method is useful especially when generating PE trajectories for humanoid robots through experiments with the human-size humanoid HRP-4.

II. Background of identification

This section presents some preliminaries for identification of a robot and the related works.

The equations of motion of the robot can be transformed into the linear form with respect to its inertial parameters [2], [3]:

$$\boldsymbol{Y}(\boldsymbol{q},\dot{\boldsymbol{q}},\ddot{\boldsymbol{q}})\boldsymbol{\phi} = \boldsymbol{\tau} \tag{1}$$

[1]K. Ayusawa A. Rioux and E. Yoshida are with CNRS-AIST JRL (Joint Robotics Laboratory), UMI13218/RL Intelligent Systems Research Institute, National Institute of Advanced Industrial Science and Technology (IS-AIST), Japan. {k.ayusawa, e.yoshida}@aist.go.jp

[2]G. Venture is with Department of Mechanical Systems Engineering, Graduate School of Engineering, Tokyo University of Agriculture and Technology.

[3]M. Gautier is with University of Nantes/LS2N, France.

where vector $\boldsymbol{\phi}$ indicates the inertial parameters of the robot, vector $\boldsymbol{\tau}$ means the joint torques, and matrix $\boldsymbol{Y}$ is called a regressor matrix which is computed from generalized coordinates $\boldsymbol{q}$ and their derivatives $\dot{\boldsymbol{q}}$ and $\ddot{\boldsymbol{q}}$,

It is also well known that vector $\boldsymbol{\phi}$ is redundant in order to construct the equations of motion; all the parameters of $\boldsymbol{\phi}$ usually cannot be identified from **Eq.**(1). The minimum set which represents the equations of motion is called base parameters [17], [18]. By utilizing base parameters, **Eq.**(1) can be transformed into the followings:

$$\boldsymbol{Y}\boldsymbol{\phi} = \boldsymbol{Y}_B\boldsymbol{\phi}_B \triangleq (\boldsymbol{Y}\boldsymbol{X})\boldsymbol{\phi}_B = \boldsymbol{\tau} \tag{2}$$

where $\boldsymbol{\phi}_B$ is the vector of base parameters, and $\boldsymbol{X}$ is the composition matrix to reconstruct new regressor $\boldsymbol{Y}_B$ from $\boldsymbol{Y}$ [9]. The structure of $\boldsymbol{X}$ depends on the kinematic configuration of the joints. Due to the linear characteristics of **Eq.**(2), the parameters of $\boldsymbol{\phi}_B$ can be identified by a linear least squares method [1]. In order to identify $\boldsymbol{\phi}_B$, the regressor matrices and the joint torques of **Eq.**(2) need to be collected at several time instances: $t = t_1, t_2, \cdots t_T$:

$$\widehat{\boldsymbol{Y}}_B\boldsymbol{\phi}_B \triangleq \begin{bmatrix} \boldsymbol{Y}_B{}^{(1)} \\ \vdots \\ \boldsymbol{Y}_B{}^{(T)} \end{bmatrix} \boldsymbol{\phi}_B = \widehat{\boldsymbol{\tau}} \triangleq \begin{bmatrix} \boldsymbol{\tau}^{(1)} \\ \vdots \\ \boldsymbol{\tau}^{(T)} \end{bmatrix} \tag{3}$$

where '$*^{(t)}$' is variable '$*$' at time instance t.

The most fundamental formula in order to identify $\boldsymbol{\phi}_B$ is written as follows:

$$\boldsymbol{\phi}_B^{ref} = (\widehat{\boldsymbol{Y}}_B{}^T\widehat{\boldsymbol{Y}}_B)^{-1}\widehat{\boldsymbol{Y}}_B{}^T\widehat{\boldsymbol{\tau}} \tag{4}$$

where $\boldsymbol{\phi}_B^{ref}$ is the solution of the least squares.

As can be seen from **Eq.**(4), the performance of identification depends on the matrix characteristics of $\widehat{\boldsymbol{Y}}_B$; the parameters of $\boldsymbol{\phi}_B$ is difficult to be identified when matrix $\widehat{\boldsymbol{Y}}_B^T\widehat{\boldsymbol{Y}}_B$ is almost singular. Matrix $\widehat{\boldsymbol{Y}}_B$ is the functional matrix with respect to the trajectories of $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$, and $\ddot{\boldsymbol{q}}$. In order to enhance the accuracy of identification, it is important to design the appropriate trajectories. Such trajectories are called persistently exciting trajectories (PE trajectories) [10]. The following criteria is often used in order to find PE trajectories:

$$\text{cond}\ (\widehat{\boldsymbol{Y}}_B) \triangleq \frac{\sigma_{max}}{\sigma_{min}} \tag{5}$$

where σ_{max} and σ_{min} are the maximum and minimum singular values; therefore, **Eq.**(5) computes the condition number of matrix $\widehat{\boldsymbol{Y}}_B$. When the condition number is close to 1, the performance of the trajectory is considered to be good. For the convenience of explanation, let us simplify the notation of the regressor as follows:

$$\boldsymbol{C} \triangleq \widehat{\boldsymbol{Y}}_B \tag{6}$$

There are several methodologies to find trajectories to minimize the condition number of **Eq.**(5). In order to reduce computational complexity, the joint trajectories are parameterized by utilizing a polynomial interpolation [10], a Fourier series [12], [19], B-splines [13], [14], and so on. The constraints like the limits of joint angle range, velocities, and accelerations are usually considered at the same time. In the case of humanoid identification, however, there are additional constraints about the balancing problem and self-collision avoidance. The large number of DOF also makes it difficult to optimize the parameters by normal nonlinear optimization techniques. Therefore, problem **Eq.**(5) is usually replaced with a simple case like a static one [14] or the selection problem [15], [16] which finds the optimal set from the candidates of the motions that are generated in advance. Efficient optimization techniques usually require analytical gradient computation of the cost function. To the best of our knowledge, there has been no analytical formulation about the gradient of the condition number in **Eq.**(5), which is detailed in the next section.

III. Direct gradient computation for PE trajectories

This section introduces how to compute the gradient of the cost function in the following optimization problem:

$$\min_{\forall t, \boldsymbol{q}^{(t)}, \dot{\boldsymbol{q}}^{(t)}, \ddot{\boldsymbol{q}}^{(t)}} \ c \triangleq \text{cond}(\boldsymbol{C}) \tag{7}$$

The important matrix derivatives are derived in section III-A, the structure of the regressor matrix is shown in section III-B, and the derivative of the regressor matrix with respect to the generalized coordinates is formulated in section III-C. Finally, the gradient computation of **Eq.**(7) is presented in section III-D.

As this section contains several equations used only for convenience's sake from mathematical point of view, let us first summarize the computation flow of the gradient. The listed equations are later shown in this section.

1) Compute concatenated regressor $\boldsymbol{C}$ which is defined in **Eq.**(41).
2) Compute $\boldsymbol{Z}^{(t)}$: the partial derivative of the condition number with respect to the regressor at time instance t according to **Eq.**(46).
3) Compute $(\boldsymbol{Z}_{i,j} : \partial \boldsymbol{Y}_{i,j}{}^{(t)} / \partial \boldsymbol{x}^{(t)})$ for all i and j: the partial derivative w.r.t. joint angles, velocities and accelerations. It can be computed by utilizing **Eq.**(25), **Eq.**(28), **Eq.**(30), **Eq.**(36) and **Eq.**(40).
4) Compute the final gradient by **Eq.**(47)

The overview of the whole flow of the gradient computation is also shown in **Fig.**1.

A. Direct derivative computation of condition number

In this subsection, matrix $\boldsymbol{C}$ is treated as $m \times n$ real matrix. The singular value decomposition of $\boldsymbol{C}$ can be computed as follows:

$$\boldsymbol{C} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T \tag{8}$$

where $\boldsymbol{U}$ and $\boldsymbol{V}$ are unitary matrices, and $\boldsymbol{\Sigma}$ is a rectangular diagonal matrix. The variation of $\boldsymbol{C}$ is represented as follows:

$$\partial \boldsymbol{C} = \partial \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T + \boldsymbol{U}\partial\boldsymbol{\Sigma}\boldsymbol{V}^T + \boldsymbol{U}\boldsymbol{\Sigma}\partial\boldsymbol{V}^T \tag{9}$$

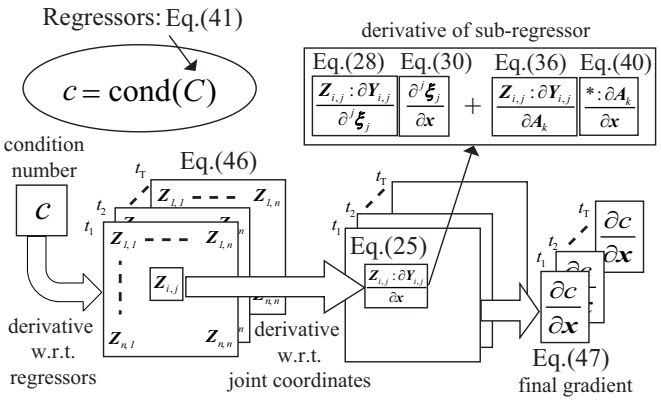

Fig. 1. Overview of gradient computation

Let us multiply by $\boldsymbol{U}^T$ from the left of **Eq.**(8) and $\boldsymbol{V}$ from the right, the following holds:

$$\partial\boldsymbol{\Sigma} = \boldsymbol{U}^T\partial\boldsymbol{C}\boldsymbol{V} - \boldsymbol{U}^T\partial\boldsymbol{U}\boldsymbol{\Sigma} - \boldsymbol{\Sigma}\partial\boldsymbol{V}^T\boldsymbol{V} \tag{10}$$

As equation $\boldsymbol{U}\boldsymbol{U}^T = \boldsymbol{I}$ holds, we have:

$$\partial\boldsymbol{U}\boldsymbol{U}^T = -\boldsymbol{U}\partial\boldsymbol{U}^T \tag{11}$$

As can be seen from **Eq.**(11), matrix $\partial\boldsymbol{U}\boldsymbol{U}^T$ is a skew symmetric matrix:

$$(\boldsymbol{U}^T\partial\boldsymbol{U})_{(k,k)} = 0 \quad (1 \leq k \leq m) \tag{12}$$

where '$*_{(q,r)}$' means the element in the q-th row and r-th column of matrix '$*$'. Let us also define '$*_{(s)}$' as s-th element of vector '$*$'. As $\boldsymbol{\Sigma}$ is a $m \times n$ rectangular diagonal matrix, the following equation holds:

$$(\boldsymbol{U}^T\partial\boldsymbol{U}\boldsymbol{\Sigma})_{(i,i)} = 0 \quad (1 \leq i \leq \min(m,n)) \tag{13}$$

The same discussion can be applied to the case of matrix $\boldsymbol{V}$, and we can also get:

$$(\boldsymbol{\Sigma}\partial\boldsymbol{V}^T\boldsymbol{V})_{(i,i)} = 0 \quad (1 \leq i \leq \min(m,n)) \tag{14}$$

From **Eq.**(10), **Eq.**(13), and **Eq.**(14), we can obtain:

$$\partial\sigma_i \triangleq \partial\boldsymbol{\Sigma}_{(i,i)} = (\boldsymbol{U}^T\partial\boldsymbol{C}\boldsymbol{V})_{(i,i)} = \boldsymbol{u}_i{}^T(\partial\boldsymbol{C})\boldsymbol{v}_i \tag{15}$$

where σ_i is i-th singular value of matrix $\boldsymbol{C}$, and $\boldsymbol{u}_i$ and $\boldsymbol{v}_i$ are the corresponding singular vectors. The partial derivative of σ_i with respect to $\boldsymbol{C}$ is computed as:

$$\frac{\partial\sigma_i}{\partial\boldsymbol{C}} = \boldsymbol{v}_i\boldsymbol{u}_i{}^T \quad \left(\frac{\partial\sigma_i}{\partial C_{(p,q)}} = v_{i(q)}u_{i(p)}\right) \tag{16}$$

Finally, by utilizing **Eq.**(16), the partial derivative of condition number of matrix $\boldsymbol{C}$ with respect to $\boldsymbol{C}$ can be computed as follows:

$$\frac{\partial\ \mathrm{cond}(\boldsymbol{C})}{\partial\boldsymbol{C}} = \mathscr{C}(\boldsymbol{C}) \triangleq \frac{1}{\sigma_{min}}\boldsymbol{v}_{max}\boldsymbol{u}_{max}{}^T - \frac{\sigma_{max}}{\sigma_{min}^2}\boldsymbol{v}_{min}\boldsymbol{u}_{min}{}^T \tag{17}$$

where '$*_{max}$' and '$*_{min}$' are the maximum and minimum singular vectors.

It should be noted that, in actual implementations, **Eq.**(16) and **Eq.**(17) are not symbolic derivatives; $\boldsymbol{u}_i$, $\boldsymbol{v}_i$, and σ_i are usually computed by the numerical singular value decomposition. The singular values are usually listed in descending order; if $i > j$, then $\sigma_i \geq \sigma_j$. However, $\boldsymbol{U}$ and $\boldsymbol{V}$ are determined according to a choice of bases. For example, when $\exists\ i(\neq j)\ \sigma_i = \sigma_j$, corresponding $\boldsymbol{u}$ and $\boldsymbol{v}$ are arbitrary and depend on the implementation of the numerical decomposition. Therefore, the derivatives in **Eq.**(16) and **Eq.**(17) are computed according to the same rule of choosing the bases.

When computing **Eq.**(16) and **Eq.**(17), bases $\boldsymbol{u}_i$ or $\boldsymbol{v}_i$ $(i > \min(m,n))$ are not required. It leads that, in the actual implementation, the full decomposition is unnecessary; a reduced version of the singular value decomposition should be computed instead: for example, if $m < n$, $\boldsymbol{C} = \boldsymbol{U}\boldsymbol{\Sigma}^*\boldsymbol{V}^{*T}$, where $\boldsymbol{\Sigma}^*$ is $m \times m$ diagonal and $\boldsymbol{V}^*$ is $m \times n$ and contains the first m columns of $\boldsymbol{V}$.

Though **Eq.**(7) is fundamental, other efficent criteria are also proposed [20]. Since they also use the singular values of $\boldsymbol{C}$, its derivative can be also computed by utilizing **Eq.**(16).

B. Structure of regressor matrix

Let n be the number of joints. Then regressor $\boldsymbol{Y}$ is structured as follows [3], [1]:

$$\boldsymbol{Y} \triangleq \begin{bmatrix} \boldsymbol{Y}_{1,1} & \cdots & \boldsymbol{Y}_{n,1} \\ \vdots & \ddots & \vdots \\ \boldsymbol{Y}_{1,n} & \cdots & \boldsymbol{Y}_{n,n} \end{bmatrix} \tag{18}$$

Each block matrix of $\boldsymbol{Y}$ is formulated as follows:

$$\boldsymbol{Y}_{i,j} \triangleq s_{i,j}\boldsymbol{K}_i\boldsymbol{A}_i{}^{-1}\boldsymbol{A}_j\boldsymbol{Y}_j \tag{19}$$

Scalar value $s_{i,j}$ in **Eq.**(19) is a boolean value in order to represent the connectivity of the links in the system. In the formulation of this paper, let the multi-body system be considered as an open kinematic tree:

$$\boldsymbol{s}_{i,j} \triangleq \begin{cases} 1 & (j \in \mathscr{L}_i) \\ 0 & (j \notin \mathscr{L}_i) \end{cases} \tag{20}$$

where $\mathscr{L}_i$ indicates the index set of the joints which are included in the kinematic sub-tree starting from joint i.

Matrix $\boldsymbol{K}_i$ in **Eq.**(19) is the orthogonal projection matrix which maps the 6-axis forces acting on the joint coordinate into the joint torque. For example, if joint i is a rotational joint, matrix $\boldsymbol{K}_i$ is written as follows:

$$\boldsymbol{K}_i \triangleq \begin{bmatrix} \boldsymbol{0}_3{}^T & \boldsymbol{e}_i{}^T \end{bmatrix} \tag{21}$$

where $\boldsymbol{e}_i$ is the joint axis vector represented in the joint coordinate. Matrix $\boldsymbol{K}_i$ is usually constant and the example of other cases are shown, for example, in [9].

Matrix $\boldsymbol{A}_i$ in **Eq.**(19) denotes the coordinate transformation matrix of 6-axis forces and can be represented as:

$$\boldsymbol{A}_i \triangleq \mathscr{A}(\boldsymbol{p}_i, \boldsymbol{R}_i) \tag{22}$$

where $\boldsymbol{p}_i$ and $\boldsymbol{R}_i$ are the position vector and the orientation matrix of link i. The functional matrix $\mathscr{A}$ is detailed in Appendix.

Sub-regressor matrix $\boldsymbol{Y}_j$ in **Eq.**(19) is formulated as follows:

$$\boldsymbol{Y}_j \triangleq \mathscr{Y}({}^j\dot{\boldsymbol{p}}_j, {}^j\dot{\boldsymbol{\omega}}_j, {}^j\boldsymbol{\omega}_j) \triangleq \mathscr{Y}(\boldsymbol{R}_j{}^T\dot{\boldsymbol{p}}_j, \boldsymbol{R}_j{}^T\dot{\boldsymbol{\omega}}_j, \boldsymbol{R}_j{}^T\boldsymbol{\omega}_j) \quad (23)$$

where $\boldsymbol{\omega}_j$ is the vector about the angular velocity of link j, and notation '${}^j*$' means '$\boldsymbol{R}_j{}^T*$' which indicates the local representation with respect to the joint coordinate. The functional matrix $\mathscr{Y}$ is also detailed in Appendix.

C. Analytical derivative of regressor matrix

We define new vectors by concatenating the following ones for convenience of explanation.

$$\boldsymbol{x} \triangleq \begin{bmatrix}\boldsymbol{q}\\ \dot{\boldsymbol{q}}\\ \ddot{\boldsymbol{q}}\end{bmatrix}, \quad \boldsymbol{\xi}_j \triangleq \begin{bmatrix}\dot{\boldsymbol{p}}_j\\ \dot{\boldsymbol{\omega}}_j\\ \boldsymbol{\omega}_j\end{bmatrix}, \quad {}^j\boldsymbol{\xi}_j \triangleq \begin{bmatrix}{}^j\dot{\boldsymbol{p}}_j\\ {}^j\dot{\boldsymbol{\omega}}_j\\ {}^j\boldsymbol{\omega}_j\end{bmatrix} \quad (24)$$

As mentioned in **Eq.**(23), vector ${}^j\boldsymbol{\xi}_j$ indicates the local coordinate representation of $\boldsymbol{\xi}_j$.

Now let us consider the derivative of the Frobenius inner product between the following two matrices: $\boldsymbol{Z}_{i,j} : \boldsymbol{Y}_{i,j}$, where operation ':' computes the Frobenius inner product (i.e. $\boldsymbol{Z}_{i,j} : \boldsymbol{Y}_{i,j} = \mathrm{trace}(\boldsymbol{Z}_{i,j}{}^T\boldsymbol{Y}_{i,j})$). The Frobenius inner products are used for multiplying several partial derivatives with respect to matrices. Let $\boldsymbol{Z}_{i,j}$ be the matrix whose size is equal to $\boldsymbol{Y}_{i,j}$. Though $\boldsymbol{Z}_{i,j}$ is defined in detail later, let us assume that $\boldsymbol{Z}_{i,j}$ is an arbitrary constant matrix in this subsection.

Then, the partial derivative of the inner product is computed as follows:

$$\frac{\boldsymbol{Z}_{i,j} : \partial \boldsymbol{Y}_{i,j}}{\partial \boldsymbol{x}} = \frac{\boldsymbol{Z}_{i,j} : \partial \boldsymbol{Y}_{i,j}}{\partial {}^j\boldsymbol{\xi}_j}\frac{\partial {}^j\boldsymbol{\xi}_j}{\partial \boldsymbol{x}} + \sum_k \frac{\frac{\boldsymbol{Z}_{i,j}:\partial \boldsymbol{Y}_{i,j}}{\partial \boldsymbol{A}_k{}^T} : \partial \boldsymbol{A}_k}{\partial \boldsymbol{x}} \quad (25)$$

The computation of each component in **Eq.**(25) is shown in this subsection.

1) Computation of $(\boldsymbol{Z}_{i,j} : \partial \boldsymbol{Y}_{i,j})/\partial {}^j\boldsymbol{\xi}_j$ *:*

Let us transform $(\boldsymbol{Z}_{i,j} : \partial \boldsymbol{Y}_{i,j})/\partial {}^j\boldsymbol{\xi}_j$ by utilizing **Eq.**(19):

$$\frac{\boldsymbol{Z}_{i,j} : \partial \boldsymbol{Y}_{i,j}}{\partial {}^j\boldsymbol{\xi}_j} = \frac{\left(\boldsymbol{B}_{i,j}{}^T\boldsymbol{Z}_{i,j}\right) : \partial \boldsymbol{Y}_j}{\partial {}^j\boldsymbol{\xi}_j} \quad (26)$$

$$\boldsymbol{B}_{i,j} \triangleq s_{i,j}\boldsymbol{K}_i\boldsymbol{A}_i{}^{-1}\boldsymbol{A}_j \quad (27)$$

Since $\boldsymbol{B}_{i,j}$ is independent from ${}^j\boldsymbol{\xi}_j$, **Eq.**(27) can be transformed by using **Eq.**(23) as follows:

$$\frac{\boldsymbol{Z}_{i,j} : \partial \boldsymbol{Y}_{i,j}}{\partial {}^j\boldsymbol{\xi}_j} = \boldsymbol{g}_{i,j}(\boldsymbol{Z}_{i,j}) \triangleq \begin{bmatrix}(\boldsymbol{B}_{i,j}{}^T\boldsymbol{Z}_{i,j}) : \mathscr{Y}_A(\boldsymbol{e}_x)\\ (\boldsymbol{B}_{i,j}{}^T\boldsymbol{Z}_{i,j}) : \mathscr{Y}_A(\boldsymbol{e}_y)\\ (\boldsymbol{B}_{i,j}{}^T\boldsymbol{Z}_{i,j}) : \mathscr{Y}_A(\boldsymbol{e}_z)\\ (\boldsymbol{B}_{i,j}{}^T\boldsymbol{Z}_{i,j}) : \mathscr{Y}_B(\boldsymbol{e}_x)\\ (\boldsymbol{B}_{i,j}{}^T\boldsymbol{Z}_{i,j}) : \mathscr{Y}_B(\boldsymbol{e}_y)\\ (\boldsymbol{B}_{i,j}{}^T\boldsymbol{Z}_{i,j}) : \mathscr{Y}_B(\boldsymbol{e}_z)\\ (\boldsymbol{B}_{i,j}{}^T\boldsymbol{Z}_{i,j}) : \mathscr{Y}_C(\boldsymbol{e}_x, {}^j\boldsymbol{w}_j)\\ (\boldsymbol{B}_{i,j}{}^T\boldsymbol{Z}_{i,j}) : \mathscr{Y}_C(\boldsymbol{e}_y, {}^j\boldsymbol{w}_j)\\ (\boldsymbol{B}_{i,j}{}^T\boldsymbol{Z}_{i,j}) : \mathscr{Y}_C(\boldsymbol{e}_z, {}^j\boldsymbol{w}_j)\end{bmatrix}^T \quad (28)$$

where $\boldsymbol{I}_3 = [\boldsymbol{e}_x\ \boldsymbol{e}_y\ \boldsymbol{e}_z]$ is an 3×3 identity matrix, and functional matrices $\mathscr{Y}_A$, $\mathscr{Y}_B$ and $\mathscr{Y}_C$ are defined in Appendix.

2) Computation of $\partial {}^j\boldsymbol{\xi}_j/\partial \boldsymbol{x}$*:*

The relation between $\boldsymbol{\xi}_j$ and ${}^j\boldsymbol{\xi}_j$ is written as follows:

$$\boldsymbol{\xi}_j = \widehat{\boldsymbol{R}}_j{}^j\boldsymbol{\xi}_j, \quad \widehat{\boldsymbol{R}}_j \triangleq \begin{bmatrix}\boldsymbol{R}_j & \boldsymbol{O}_{3\times3} & \boldsymbol{O}_{3\times3}\\ \boldsymbol{O}_{3\times3} & \boldsymbol{R}_j & \boldsymbol{O}_{3\times3}\\ \boldsymbol{O}_{3\times3} & \boldsymbol{O}_{3\times3} & \boldsymbol{R}_j\end{bmatrix} \quad (29)$$

Then the partial derivative of $\boldsymbol{\xi}_j$ with respect to $\boldsymbol{x}$ can be computed as follows:

$$\frac{\partial {}^j\boldsymbol{\xi}_j}{\partial \boldsymbol{x}} = \widehat{\boldsymbol{R}}_j\frac{\partial \boldsymbol{\xi}_j}{\partial \boldsymbol{x}} + \begin{bmatrix}[\ddot{\boldsymbol{p}}_j\times]\boldsymbol{J}_{Wj}\\ [\dot{\boldsymbol{\omega}}_j\times]\boldsymbol{J}_{Wj}\\ [\boldsymbol{\omega}_j\times]\boldsymbol{J}_{Wj}\end{bmatrix} \quad (30)$$

$$\boldsymbol{J}_{Wj} \triangleq \begin{bmatrix}\boldsymbol{O}_3 & \boldsymbol{I}_3\end{bmatrix}\boldsymbol{J}_{Bj} \quad (31)$$

where $\boldsymbol{J}_{Bj}$ is the basic Jacobian matrix [21] of link j; the following relationship holds: $[\dot{\boldsymbol{p}}_j{}^T\ \boldsymbol{\omega}_j{}^T]^T = \boldsymbol{J}_{Bj}\dot{\boldsymbol{q}}$.

Partial derivative $\partial \boldsymbol{\xi}_j/\partial \boldsymbol{x}$ consists of the Jacobian matrices of the angular velocity and the linear and angular acceleration of link i with respect to $\boldsymbol{x}$. The computation method of those Jacobian matrices can be found in [22].

3) Computation of $(\boldsymbol{Z}_{i,j} : \partial \boldsymbol{Y}_{i,j})/\partial \boldsymbol{A}_k$ *:*

Since $\boldsymbol{Y}_{i,j}$ contains $\boldsymbol{A}_i$ and $\boldsymbol{A}_j$, let us compute the partial derivative of the inner product with respect to the two respectively:

$$\frac{\boldsymbol{Z}_{i,j} : \partial \boldsymbol{Y}_{i,j}}{\partial \boldsymbol{A}_j} = \frac{\boldsymbol{G}_{A_{i,j}} : \partial \boldsymbol{A}_j}{\partial \boldsymbol{A}_j} = \boldsymbol{G}_{A_{i,j}}{}^T \quad (32)$$

$$\frac{\boldsymbol{Z}_{i,j} : \partial \boldsymbol{Y}_{i,j}}{\partial \boldsymbol{A}_i} = \frac{\widehat{\boldsymbol{G}}_{A_{i,j}} : \partial \boldsymbol{A}_i{}^{-1}}{\partial \boldsymbol{A}_i} = \boldsymbol{A}_j{}^{-1}\widehat{\boldsymbol{G}}_{A_{i,j}}{}^T\boldsymbol{A}_j{}^{-1} \quad (33)$$

where $\boldsymbol{G}_{A_{i,j}}$ and $\widehat{\boldsymbol{G}}_{A_{i,j}}$ are defined as follows:

$$\boldsymbol{G}_{A_{i,j}} \triangleq s_{i,j}(\boldsymbol{K}_i\boldsymbol{A}_i{}^{-1})^T\boldsymbol{Z}_{i,j}(\boldsymbol{Y}_j)^T \quad (34)$$

$$\widehat{\boldsymbol{G}}_{A_{i,j}} \triangleq s_{i,j}(\boldsymbol{K}_i)^T\boldsymbol{Z}_{i,j}(\boldsymbol{A}_j\boldsymbol{Y}_j)^T \quad (35)$$

Then the partial derivative of $\boldsymbol{A}_k$ for arbitrary index k can be written as:

$$\frac{\boldsymbol{Z}_{i,j} : \partial \boldsymbol{Y}_{i,j}}{\partial \boldsymbol{A}_k} = \delta_{j,k}\boldsymbol{G}_{A_{i,j}}{}^T - \delta_{i,k}\boldsymbol{A}_j{}^{-1}\widehat{\boldsymbol{G}}_{A_{i,j}}{}^T\boldsymbol{A}_j{}^{-1} \quad (36)$$

where $\delta_{q,r}$ is a Kronecker delta.

4) Computation of $(* : \partial \boldsymbol{A}_{i,j})/\partial \boldsymbol{x}$ *:*

Let us define matrix $\boldsymbol{G}_{i,j}$ whose size is equal to $\boldsymbol{A}_{i,j}$ and compute the following inner product:

$$\frac{\boldsymbol{G}_{i,j} : \partial \boldsymbol{A}_k}{\partial \boldsymbol{x}} = \sum_p \boldsymbol{e}_p{}^T\boldsymbol{G}_{i,j}{}^T\frac{\partial(\boldsymbol{A}_k\boldsymbol{e}_p)}{\partial \boldsymbol{x}} \quad (37)$$

Before the further computation, we assume the following equation:

$$\boldsymbol{b} = \boldsymbol{A}_k\boldsymbol{a} \quad (38)$$

where $\boldsymbol{a}$ is an arbitrary constant vector. Then the partial derivative of $\boldsymbol{b}$ with respect to $\boldsymbol{x}$ is computed as follows:

$$\frac{\partial \boldsymbol{b}}{\partial \boldsymbol{x}} = \boldsymbol{A}_k[(-\boldsymbol{a})\hat{\bullet}]\mathscr{A}(\boldsymbol{0}_3, \boldsymbol{R}_k{}^{-1})\boldsymbol{J}_{Bk} \quad (39)$$

From **Eq.**(37) and **Eq.**(39), we have:

$$\frac{\boldsymbol{G}_{i,j} : \partial \boldsymbol{A}_k}{\partial \boldsymbol{x}} = \sum_p \boldsymbol{e}_p{}^T\boldsymbol{G}_{i,j}{}^T\boldsymbol{A}_k[-\boldsymbol{e}_p\hat{\bullet}]\mathscr{A}(\boldsymbol{0}_3, \boldsymbol{R}_k{}^{-1})\boldsymbol{J}_{Bk} \quad (40)$$

D. Gradient computation

The measurement data used in the identification is often obtained through several experimental trials. Though matrix $\boldsymbol{C}$ was defined as **Eq.**(6) in section II, let us redefine $\boldsymbol{C}$ as follows:

$$\boldsymbol{C} \triangleq \left[\boldsymbol{C}_0{}^T \quad \widehat{\boldsymbol{Y}}_B{}^T \right]^T \tag{41}$$

where $\boldsymbol{C}_0$ is an $m_1 \times n$ arbitrary matrix, and let us consider $\widehat{\boldsymbol{Y}}_B$ as a $m_2 T \times n$ matrix. For example, when some measurement data were already available for the identification, $\boldsymbol{C}_0$ corresponds to the matrix which is computed from the data. The regressor at t-th time instance can be extracted as follows:

$$\boldsymbol{Y}_B^{(t)} = \boldsymbol{S}^{(t)} \boldsymbol{C} \tag{42}$$

$$\boldsymbol{S}^{(t)} \triangleq \left[\boldsymbol{O}_{m_2 \times m_1} \quad \widehat{\boldsymbol{S}}^{(t,1)} \cdots \quad \widehat{\boldsymbol{S}}^{(t,l)} \cdots \quad \right] \tag{43}$$

$$\widehat{\boldsymbol{S}}^{(t,l)} \triangleq \delta_{t,l} \boldsymbol{E}_{m_2 \times m_2} \tag{44}$$

The goal is to compute the gradient of cost function c with respect to $\boldsymbol{x}^{(t)}$. Before doing this, let us compute the gradient with respect to normal regressor $\boldsymbol{Y}^{(t)}$ in **Eq.**(2) as follows:

$$\frac{\partial c}{\partial \boldsymbol{Y}^{(t)}} = \frac{\frac{\partial c}{\partial \boldsymbol{C}^T} : \partial \boldsymbol{C}}{\partial \boldsymbol{Y}^{(t)}} = \boldsymbol{X} \mathscr{C}(\boldsymbol{C}) \boldsymbol{S}^{(t)T} \tag{45}$$

Then we define the following matrix:

$$\boldsymbol{Z}^{(t)} \triangleq \begin{bmatrix} \boldsymbol{Z}_{1,1}{}^{(t)} & \cdots & \boldsymbol{Z}_{n,1}{}^{(t)} \\ \vdots & \ddots & \vdots \\ \boldsymbol{Z}_{1,n}{}^{(t)} & \cdots & \boldsymbol{Z}_{n,n}{}^{(t)} \end{bmatrix} \triangleq \boldsymbol{X} \mathscr{C}(\boldsymbol{C}) \boldsymbol{S}^{(t)T} \tag{46}$$

Block matrix $\boldsymbol{Z}_{i,j}$ corresponds to the matrix which has been already shown in the previous subsection.

Finally, the gradient vector with respect to $\boldsymbol{x}^{(t)}$ can be computed as:

$$\frac{\partial c}{\partial \boldsymbol{x}^{(t)}} = \sum_i \sum_j \frac{\boldsymbol{Z}_{i,j} : \partial \boldsymbol{Y}_{i,j}{}^{(t)}}{\partial \boldsymbol{x}^{(t)}} \tag{47}$$

IV. Optimization results

In this section, the proposed method has been tested on a humanoid robot and some results are presented, after showing the setting of our optimization problem.

A. Setting of optimization problem

1) Trajectory parameterization: In the previous section, all the generalized coordinates and their derivatives are considered as the variables in the optimization problem of **Eq.**(7). In order to reduce computational cost or to smooth the trajectories, they are usually parameterized by using functional trajectory representation.

$$\boldsymbol{x}^{(t)} = f(\boldsymbol{\alpha}) \tag{48}$$

where $\boldsymbol{\alpha}$ is the set of the parameters used in the functional trajectory.

When using trajectory representation with **Eq.**(48), gradient **Eq.**(47) can be transformed as follows:

$$\frac{\partial c}{\partial \boldsymbol{\alpha}} = \sum_t \sum_i \sum_j \frac{\partial \left(\boldsymbol{Z}_{i,j} : \boldsymbol{Y}_{i,j}{}^{(t)} \right)}{\partial \boldsymbol{x}^{(t)}} \frac{\partial \boldsymbol{x}^{(t)}}{\partial \boldsymbol{\alpha}} \tag{49}$$

In this paper, we utilized a Fourier series in order to parameterize the joint trajectory [12], [19]; coordinate $\boldsymbol{q}^{(t)}$ is represented as follows:

$$q_{(j)}{}^{(t)} = \sum_{k=1}^{N} (a_{j,k} \sin(k\omega t) + b_{j,k} \cos(k\omega t)) + q_{0j} \tag{50}$$

We can easily obtain $\dot{\boldsymbol{q}}^{(t)}$ and $\ddot{\boldsymbol{q}}^{(t)}$ by differentiating **Eq.**(50).

Let us assume that N, ω_i and time instances $t = t_1, t_2, \cdots$ are constant. In this case, parameter $\boldsymbol{\alpha}$ is equal to the set of $a_{j,k}$, $b_{j,k}$, and q_{0j} for all j and k. Since **Eq.**(50) and its derivatives are linear with respect to $a_{j,k}$, $b_{j,k}$, and q_{0j}, each component of $\frac{\partial \boldsymbol{x}^{(t)}}{\partial \boldsymbol{\alpha}}$ can be simply obtained from their coefficient values.

2) Constraints: When solving **Eq.**(7), the following conditions about the limits of joint angles and their derivatives:

$$\boldsymbol{x}_{min} \leq \boldsymbol{x}^{(t)} \leq \boldsymbol{x}_{max} \tag{51}$$

where vectors $\boldsymbol{x}_{min}$ and $\boldsymbol{x}_{max}$ mean the minimum or the maximum values respectively.

We also assume the following limitation of the center of total mass, in order to keep the balance of the robot:

$$\boldsymbol{p}_c^{est\,(t)} \subset \mathbb{P} \tag{52}$$

where $\boldsymbol{p}_c^{est}$ is the estimated center of total mass, and $\mathbb{P}$ represents the motion range of the center of total mass. Since the inertial parameters are unknown before identification, in order to compute $\boldsymbol{p}_c^{est}$, we used the a-priori knowledge like CAD data. In the field of motion optimization, the conditions of not only the center of total mass but also ZMP are usually considered. Unfortunately, before identifying the accurate inertial parameters, we need to set rather conservative safety margins in those equations. In our case, $\mathbb{P}$ in **Eq.**(52) was designed such that the distance between the projected center of total mass on the XY plane and the center of the supporting area is within 1 cm.

The self-collision avoidance was also considered by approximating the robot shape with several capsule primitives [23]. Then the collision among the primitives can be detected by checking the distance between line segments. The condition about the distance between line segment i and j is as follows:

$$d_{line}(\boldsymbol{p}_{L_i}^{(s)}, \boldsymbol{p}_{L_i}^{(e)}, \boldsymbol{p}_{L_j}^{(s)}, \boldsymbol{p}_{L_j}^{(e)}) < r_i + r_j \tag{53}$$

where r_i is the allowable penetration distance of line segment i (or the radius of capsule i), and $\boldsymbol{p}_{L_i}^{(s)}$ and $\boldsymbol{p}_{L_i}^{(e)}$ mean the position of the start and end point of line segment i. The points are located on the coordinate system of joint L_i. Since $\boldsymbol{p}_{L_i}^{(s)}$ and $\boldsymbol{p}_{L_i}^{(e)}$ are the function of the joint angles, the analytical derivative of the distance with respect to the joint angles can be computed by utilizing the basic Jacobians.

B. Results

In order to validate the proposed method, we generated the trajectories of the humanoid robot HRP-4 [24]. We focused on generating the motion of the two arms; each of which has 7 rotational joints; the number of DOF is $N_J = 14$. As fast movement of arms makes the robot easily fall down, the balancing condition needs to be considered. We solved problem **Eq.**(7) under inequality constraints **Eq.**(51), **Eq.**(52), and **Eq.**(53) with a Fourier series **Eq.**(50), and optimized parameter $\boldsymbol{\alpha}$.

In this paper, the number of Fourier bases was $N = 10$, the basic frequency was $\omega = \pi/4$. In the cost function, the number of time instance t was 40 and each timestep was set at $0.2[s]$; the time duration of the trajectory was total $8.0[s]$. Small timestep is not necessary for identification when concatenating the regressors in **Eq.**(41). However, the physical consistent conditions like **Eq.**(51), **Eq.**(52), and **Eq.**(53) need to be evaluated with higher sampling rate. For this reason, we check the constraints of 400 time instances at every $0.02[s]$. The optimization was solved by quasi-Newton method by utilizing the proposed gradient computation.

We at first verified the correctness and the computational speed of our gradient computation (i.e. **Eq.**(49)) to the numerically computed one by the computer with Intel(R) Core(TM) i7-4800MQ CPU. In the current optimization procedure, the number of variables is $294(= (2N+1) \times N_J)$. When the step size of numerical gradient is $1.0E-6$, the mean relative error norm between the two results is $1.3E-5$, which verified the correctness of our method. The averaged computation time when computing **Eq.**(49) is 0.2 [s], while the time required for the numerical gradient is 17.8 [s]. The computation time was significantly improved by our method.

Since the formulation of **Eq.**(41) enables the optimization of the trajectories by utilizing the a-priori data, we generated the trajectories by iterating the following process:

1) At first, the trajectory was optimized and generated without a-priori experimental data: $\boldsymbol{C}_0 = \boldsymbol{O}$ in **Eq.**(41). Let $\boldsymbol{C}^{(1)}$ be obtained $\boldsymbol{C}$ in the first generation process.
2) In i-th generation process ($i > 1$), all the previous experimental data was utilized: ${\boldsymbol{C}_0}^T = [{\boldsymbol{C}^{(1)}}^T \dots {\boldsymbol{C}^{(i-1)}}^T]^T$, where $\boldsymbol{C}^{(i)} \triangleq \widehat{\boldsymbol{Y}}_B$ in i-th process.

The trajectories were generated according to the above procedure. **Table** I shows the condition number of each $\boldsymbol{C}^{(i)}$ and concatenated $\boldsymbol{C}$. After the fourth iteration, the improvement of the condition number was not recognized. For comparison, we have also generated several trajectories by using a conventional method [16], based on optimal selection from randomly generated trajectories with only consideration on physical consistent conditions. We performed the selection by limiting the total number of selected motions, and repeatedly updated the set of motions by adding the motions of better quality. **Table** II shows the condition number in the case of the random generation. As can be seen from the tables, all the condition numbers in **Table** I show small values with respect to the case in **Table** II. It could show the efficiency of our optimization-based method.

TABLE I

CONDITION NUMBER OF THE OPTIMIZED REGRESSORS

motion index	1	2	3
$(\mathrm{cond}(\boldsymbol{C}^{(i)}))$	47.6	114.2	125.5
$(\mathrm{cond}(\boldsymbol{C}))$	47.6	35.7	33.9

TABLE II

CONDITION NUMBER OF THE RANDOMLY GENERATED REGRESSOR

motion index	1	2	3
$(\mathrm{cond}(\boldsymbol{C}^{(i)}))$	274.1	332.0	436.2
$(\mathrm{cond}(\boldsymbol{C}))$	274.1	156.3	124.7

The generated trajectories were performed by using the humanoid robot HRP-4 in the experiment. **Fig.**2 show the snapshots of HRP-4 performing the motions generated by 1st iteration. The robot could perform the motion without violating mechanical limits, causing self-collision, or falling down; the physically consistent trajectories could be successfully generated in all cases.

V. CONCLUSION

We propose a novel method for generating PE trajectories by using optimization together with efficient gradient computation of the condition number with respect to the joint trajectory parameters. Generally in optimization technique, direct and analytical gradient computation can benefit from computational speed, accuracy, and stability, compared to the numerical ones. This computational improvement make tractable optimization for large DOF systems under many constraints such as humanoid robots. If some experimental data are already available, the method can also generate the trajectories by exploiting such existing data, which allows the sequential generation of PE trajectories.

Our method was tested on the humanoid robot HRP-4. In the optimization, we focused on generating the motion with 14 joints in the arms; other joints were fixed at constant values. The joint trajectories are represented and parameterized by a Fourier series, by taking into account the robot's mechanical constraints, self-collision avoidance, and the condition about the whole body COM. The proposed optimization framework could successfully generate the trajectories that satisfy such physical consistent conditions, allowing the humanoid robot to perform the trajectories without falling down during the experiments. We generated the trajectories sequentially by concatenating the data in the optimization procedure. The resulting condition number converges to 34 until third trial, which is significantly smaller than those generated from the conventional method of optimal selection of randomly generated motions.

In the theoretical point of view, the analytical formulation of the gradient also has a potential to clarify the dynamics feature of optimal trajectories by checking the analytical stationary conditions such that the gradient is equal to zero. It will be addressed in our future work.

REFERENCES

[1] W. Khalil and E. Dombre, *Modeling, identification and control of robots*, Hermès Penton, London-U.K, 2002.

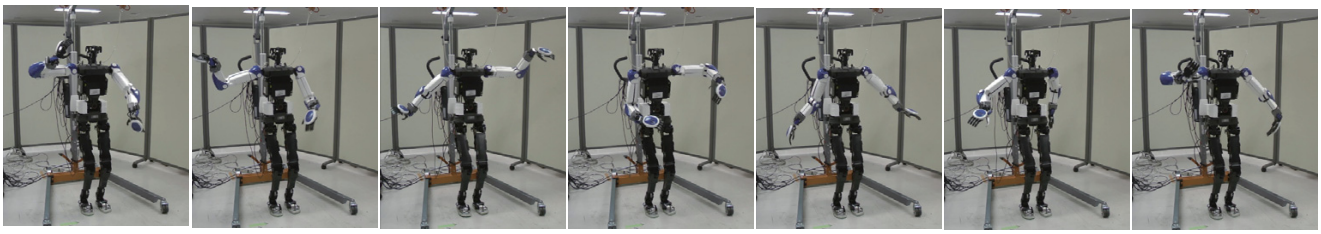

Fig. 2. Snapshots of HRP-4 performing the optimized trajectory generated at the first trial.

[2] H. Mayeda, K. Osuka, and A. Kanagawa, "A new identification method for serial manipulator arms," in *Proc. of 9th IFAC World Congress*, 1984, vol. 2, pp. 74–79.

[3] C.-G. Atkeson, C.-H. An, and J.-M. Hollerbach, "Estimation of inertial parameters of manipulator loads and links," *Int. J. of Robotic Research*, vol. 5, no. 3, pp. 101–119, 1986.

[4] G. Liu, K. Iagnemma, S. Dubowsky, and G. Morel, "A base force/torque sensor approach to robot manipulator inertial parameter estimation," in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 1998, pp. 3316–3321.

[5] K. Yoshida and S. Abiko, "Inertia parameter identification for a free-flying space robot," in *Proc. of the AIAA Guidance, Navigation, and Control Conference, (AIAA 2002-4568)*, 2002.

[6] G. Venture, P.J. Ripert, W. Khalil, M. Gautier, and P. Bodson, "Modeling and identification of passenger car dynamics using robotics formalism," *IEEE Trans. on Intelligent Transportation Systems*, vol. 7, no. 3, pp. 349–359, 2006.

[7] K. Ayusawa, G. Venture, and Y. Nakamura, "Identification of humanoid robots dynamics using floating-base motion dynamics," in *Proc. of the 2008 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2008, pp. 2854–2859.

[8] M. Mistry, S. Schaal, and K. Yamane, "Inertial parameter estimation of floating-base humanoid systems using partial force sensing," in *Proc. of the IEEE-RAS Int. Conf. of Humanoid Robots*, 2009, pp. 492–497.

[9] K. Ayusawa, G. Venture, and Y. Nakamura, "Identifiability and identification of inertial parameters using the underactuated base-link dynamics for legged multibody systems," *Int. J. of Robotics Research*, vol. 33, no. 3, pp. 446–468, 2014.

[10] M. Gautier and W. Khalil, "Exciting trajectories for the identification of base inertial parameters of robots," *Int. J. of Robotics Research*, vol. 11, no. 4, pp. 363–375, 1992.

[11] J. Swevers, C. Ganseman, D. Bilgin, J. De Schutter, and H. Van Brussel, "Optimal robot excitation and identification," *IEEE Trans. on Robotics and Automation*, vol. 13, no. 5, pp. 730–740, 1997.

[12] K.-J. Park, "Fourier-based optimal excitation trajectories for the identification of robots," *Robotica*, vol. 24, pp. 625–633, 2006.

[13] W. Rackl, R. Lampariello, and G. Hirzinger, "Robot excitation trajectories for dynamic parameter estimation using optimized B-splines," in *Proc of the 2009 IEEE Int. Conf. on Robotics and Automation*, 2012, pp. 2042–2047.

[14] V. Bonnet, P. Fraisse, A. Crosnier, M. Gautier, A. González, and G. Venture, "Optimal exciting dance for identifying inertial parameters of an anthropomorphic structure," *IEEE Trans. on Robotics*, vol. 32, no. 4, pp. 823–836, 2016.

[15] G. Venture, K. Ayusawa, and Y. Nakamura, "A numerical method for choosing motions with optimal excitation properties for identification of biped dynamics - an application to human," in *Proc of the 2009 IEEE Int. Conf. on Robotics and Automation*, 2009, pp. 1226–1231.

[16] J. Jovic, F. Philipp, A. Escande, K. Ayusawa, E. Yoshida, A. Kheddar, and G. Venture, "Identification of dynamics of humanoids: Systematic exciting motion generation," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2015, pp. 2173–2179.

[17] H. Mayeda, K. Yoshida, and K. Osuka, "Base parameters of manipulator dynamic models," *IEEE Trans. on Robotics and Automation*, vol. 6, no. 3, pp. 312–321, 1990.

[18] M. Gautier and W. Khalil, "Direct calculation of minimum set of inertial parameters of serial robots," *IEEE Trans. on Robotics and Automation*, vol. 6, no. 3, pp. 368–373, 1990.

[19] J. Vantilt, E. Aertbeliün, F. De Groote, and J. De Schutter, "Optimal excitation and identification of the dynamic model of robotic systems with compliant actuators," in *Proc of the 2015 IEEE Int. Conf. on Robotics and Automation*, 2015, pp. 2117–2124.

[20] C. Presse and M. Gautier, "New criteria of exciting trajectories for robot identification," in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 1993, pp. 907–912.

[21] O. Khatib, "A unified approach for motion and force control of robotic manipulators: the operational space formulation.," *IEEE J. Robotics and Automation*, vol. 3, no. 1, pp. 43–53, 1987.

[22] W. Suleiman, E. Yoshida, F. Kanehiro, J-P. Laumond, and A. Monin, "Human motion imitation by humanoid robot," in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 2008, pp. 2967–2704.

[23] O. Kanoun, J-P. Laumond, and E. Yoshida, "Planning foot placements for a humanoid robot: A problem of inverse kinematics," *Int. J. of Robotics Research*, vol. 30, no. 4, pp. 476–485, 2011.

[24] 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 *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2011, pp. 4400–4407.

APPENDIX

In this appendix, several basic mathematical notations used in the paper are introduced.

1) skew operator:

$$[\boldsymbol{x}\times] \triangleq \begin{bmatrix} 0 & -x_{(3)} & x_{(2)} \\ x_{(3)} & 0 & -x_{(1)} \\ -x_{(2)} & x_{(1)} & 0 \end{bmatrix} \tag{54}$$

2) operator for linear and angular velocities:

$$[\boldsymbol{\xi}_{\hat{\bullet}}] \triangleq \begin{bmatrix} \boldsymbol{O}_{3\times 3} & [\boldsymbol{v}\times] \\ [\boldsymbol{v}\times] & [\boldsymbol{\omega}\times] \end{bmatrix}, \quad \boldsymbol{\xi} \triangleq \begin{bmatrix} \boldsymbol{v} \\ \boldsymbol{\omega} \end{bmatrix} \tag{55}$$

where $\boldsymbol{O}_{m\times n}$ is a $m \times n$ zero matrix.

3) operator used in regressor matrix:

$$[\boldsymbol{x}\bullet] \triangleq \begin{bmatrix} x_{(1)} & 0 & 0 & 0 & x_{(3)} & x_{(2)} \\ 0 & x_{(2)} & 0 & x_{(3)} & 0 & x_{(1)} \\ 0 & 0 & x_{(3)} & x_{(2)} & x_{(1)} & 0 \end{bmatrix} \tag{56}$$

4) coordinate transformation of position and orientation:

$$\mathscr{A}(\boldsymbol{p},\boldsymbol{R}) \triangleq \begin{bmatrix} \boldsymbol{R} & [\boldsymbol{p}\times]\boldsymbol{R} \\ \boldsymbol{O}_{3\times 3} & \boldsymbol{R} \end{bmatrix} \tag{57}$$

5) functional matrix for regressor matrix of a rigid body:

$$\hat{\mathscr{Y}}(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c},\widehat{\boldsymbol{c}}) = \begin{bmatrix} \boldsymbol{a} & [\boldsymbol{b}\times]+[\boldsymbol{c}\times][\widehat{\boldsymbol{c}}\times] & \boldsymbol{O}_{3\times 3} \\ \boldsymbol{0}_3 & -[\boldsymbol{a}\times] & [\boldsymbol{b}\bullet]+[\boldsymbol{c}\times][\widehat{\boldsymbol{c}}\bullet] \end{bmatrix} \tag{58}$$

where $\boldsymbol{0}_m$ is a zero vector whose size is m.

6) several shortened forms of $\hat{\mathscr{Y}}$:

$$\mathscr{Y}(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c}) \triangleq \hat{\mathscr{Y}}(\boldsymbol{a},\boldsymbol{b},\boldsymbol{c},\boldsymbol{c}) \tag{59}$$

$$\mathscr{Y}_A(\boldsymbol{a}) \triangleq \hat{\mathscr{Y}}(\boldsymbol{a},\boldsymbol{0}_3,\boldsymbol{0}_3,\boldsymbol{0}_3) \tag{60}$$

$$\mathscr{Y}_B(\boldsymbol{b}) \triangleq \hat{\mathscr{Y}}(\boldsymbol{0}_3,\boldsymbol{b},\boldsymbol{0}_3,\boldsymbol{0}_3) \tag{61}$$

$$\mathscr{Y}_C(\boldsymbol{c},\widehat{\boldsymbol{c}}) \triangleq \hat{\mathscr{Y}}(\boldsymbol{0}_3,\boldsymbol{0}_3,\boldsymbol{c},\widehat{\boldsymbol{c}}) + \hat{\mathscr{Y}}(\boldsymbol{0}_3,\boldsymbol{0}_3,\widehat{\boldsymbol{c}},\boldsymbol{c}) \tag{62}$$