On the Derivation of the Contact Dynamics in Arbitrary Frames: Application to Polishing with Talos

Sébastien Kleff, Justin Carpentier, Nicolas Mansard, Ludovic Righetti

On the Derivation of the Contact Dynamics in Arbitrary Frames: Application to Polishing with Talos

Sébastien Kleff[1,2], Justin Carpentier[4], Nicolas Mansard[2,3], Ludovic Righetti[1]

***Abstract*— Contact dynamics relies on the simultaneous satisfaction of constraints at the robot body level and at the contact level. At both levels, various formulations can be chosen that all must lead to the same results, given the same hypothesis, hence the little importance of their details. Yet when using it in an optimal control problem, a particular formulation is often imposed by the task to be performed by the robot. In this paper, we detail the formulation of the contact quantities (force, movement) in an arbitrary frame imposed by the task. In that case, we will show that we are typically not interested in working in the local frame (attached to the robot contact point), nor in the world frame, but in a user-defined frame centered at the contact location with a fixed orientation in the world. The derivations can then be used for 6D, 3D or normal (pure-sliding) contact. We implemented the corresponding derivatives on top of the contact dynamics of the rigid-body dynamics library Pinocchio in the optimal control solver Crocoddyl. We show that a unique formulation is able to handle several operational orientations, by achieving several surfacing tasks in model predictive control with the robot Talos.**

I. INTRODUCTION

The advent of fast nonlinear model predictive control (MPC) on torque-controlled robots is enabled by dedicated numerical optimal control solvers that exploit the structure of the dynamics, such as DDP [1], [2]. Despite the successful deployment of these controllers on hardware [3]–[8], contact tasks are still challenging to realize and subject to active research. In this context, *designing* such tasks suppose the knowledge of desired contact forces and accelerations in a "convenient" frame - what convenient means is up to the user. However, analytical derivatives of multi-body dynamics algorithms are often expressed in local frames by construction, i.e. in frames attached at all instants to moving parts of the robot (following the efficiency of local formulation [9]). Yet from a user point of view, this representation is not the most intuitive and complicates the design of contact tasks. Indeed, think for instance of locomotion task where we would like to avoid slippage: the contact force could be regularized around a vertical force in the cost objective. As a matter of fact, doing so requires to express the residual of the contact force and its derivatives in a frame that is aligned with the world frame (i.e. with gravity) yet centered at all times with the robot foot. This kind of task is the motivation of our paper: expressing derivatives in such a custom frame, although done using basic spatial algebra tools, is not trivial.

In the literature, several contact models have been used in MPC. The rigid contact dynamics implemented in Pinocchio [10] is utilized for instance in [11]–[13]. In our previous work [11] the target force was expressed in a local frame whose alignment with the surface normal was part of the task. In [12], [13] contact force constraints are expressed in a surface-aligned frame but the contact forces are optimized as control variables and not explicitly tracked. Soft contact models are also exploited, e.g. in [14] where a visco-elastic force objective expressed in the surface coordinates, or in [15] where iLQG is used with Mujoco [16]. But Mujoco performs the Recursive Newton-Euler Algorithm only in local or world coordinates. Contact invariant approaches allowing to optimize control policies through contact have also been proposed in [7], [17], [18] but it is unclear in which coordinate frame they are expressed. Overall there seem to be little focus on this topic among the MPC community, and the most popular contact models are usually based on local formulations. MPC requires the explicit computation of the derivatives, hence an effort of the community to provide differentiable contact simulators [16], [19]–[21]. By basing our contact formulation on a differentiable contact solver [2], [22], we pursue the same objective and provide all necessary derivations to obtain the derivatives of the contact dynamics.

In this paper, we propose to derive the formulation of the rigid contact model, which includes the constrained joint acceleration and contact forces, in arbitrary frames of reference. We emphasize that this is a technical difficulty that needs to be overcome by most MPC practitioners. We present here a formal and generic way of dealing with rigid contact models of any dimensions in arbitrary frames. In particular, we focus on the generic spatial (6D) formulation from which all other formulations can be deduced. The calculations are not found explicitly in the literature, but from our experience they are tedious and error-prone. We hope that it can serve other researchers from the MPC community to design contact tasks. We also provide an open-source implementation of these derivatives on top of the Pinocchio [10] and Crocoddyl [2] libraries and we validate experimentally these developments in a nonlinear MPC framework on the torque-controlled humanoid robot Talos.

II. BACKGROUND

In this section we recall the rigid contact model [23], the constrained forward dynamics [22] and the derivation of their analytical derivatives [24]. Then we introduce the reference

[1]Tandon School of Engineering, New York University, Brooklyn, NY

[2]LAAS-CNRS, Université de Toulouse, CNRS, Toulouse

[3]Artificial and Natural Intelligence Toulouse Institute (ANITI), Toulouse

[4]Inria - Département d'Informatique de l'École Normale Supérieure, PSL Research University, Paris, France

This work was in part supported by the National Science Foundation (grants 1825993, 1932187, 1925079 and 2026479), Meta Platforms Inc., and ANR Dynamo-grade (ANR-21-LCV3-0002)

frame of interest in which we want to express contact tasks, as well as the spatial algebra tools used in the paper.

A. Constrained forward dynamics

A rigid contact is a pure kinematic constraint. The Lagrangian dynamics of a robot subject to rigid contacts is derived from Gauss' least action principle [23] stating that constrained accelerations are the closest to free accelerations

$$\min_{a_q} \ \frac{1}{2}\left\| a_q - M(q)^{-1}\left(\tau_q - b(q,v_q)\right)\right\|^2_{M(q)} \tag{1a}$$

$$\text{s.t. } f(q) = 0 \tag{1b}$$

where $q \in \mathbb{R}^{n_q}$ is the vector of joint positions, $f : \mathbb{R}^{n_q} \to \mathbb{R}^m$ is the contact constraint, $a_q \in \mathbb{R}^{n_v}$ is the vector of constrained joint accelerations, $v_q \in \mathbb{R}^{n_v}$ is the vector of joint velocities, $M(q) \in \mathbb{S}^{n_v}_+$ is the joint space inertia matrix, $b(q, v_q) \in \mathbb{R}^{n_v}$ is the vector of centrifugal, Coriolis and gravity forces, $\tau_q \in \mathbb{R}^{n_v}$ is the vector of joint torques. Differentiating twice (1b) leads to the quadratic program

$$\min_{a_q} \ \frac{1}{2}\left\| a_q - M(q)^{-1}\left(\tau_q - b(q,v_q)\right)\right\|^2_{M(q)} \tag{2a}$$

$$\text{s.t. } J(q)a_q + \underbrace{\dot{J}_c(q)v_q}_{\alpha_0(q,v_q)} = \alpha_* \tag{2b}$$

where $J(.) : \mathbb{R}^{n_q} \to \mathbb{R}^{m \times n_v}$ is the Jacobian of the contact constraint, $\alpha(q, v_q) = J(q)a_q + \alpha_0(q, v_q)$ is the contact acceleration, $\alpha_0(q, v_q) \in \mathbb{R}^m$ is the contact acceleration drift, and $\alpha_* \in \mathbb{R}^m$ is the desired contact acceleration, typically a numerical stabilization term. This program can be solved efficiently using proximal methods [25]. Dropping the dependencies in q, v_q, the KKT conditions of (2) read

$$\underbrace{\begin{bmatrix} M & J^T \\ J & 0 \end{bmatrix}}_{K} \underbrace{\begin{bmatrix} a_q \\ -\lambda \end{bmatrix}}_{y} = \underbrace{\begin{bmatrix} \tau_q - b \\ \alpha_* - \alpha_0 \end{bmatrix}}_{\sigma} \tag{3}$$

The Lagrange multipliers $\lambda \in \mathbb{R}^m$ correspond to the contact forces. The *constrained forward dynamics* is the solution map of the system (3)

$$\text{FD} : \begin{cases} \mathbb{R}^{n_q} \times \mathbb{R}^{n_v} \times \mathbb{R}^{n_v} & \to \mathbb{R}^{n_v} \times \mathbb{R}^m \\ (q, v_q, \tau_q) & \mapsto y = K^{-1}\sigma \end{cases} \tag{4}$$

Given position, velocities and torques, it computes the constrained joint acceleration and contact forces. We are interested in computing $\frac{\partial \text{FD}}{\partial q}, \frac{\partial \text{FD}}{\partial v_q}, \frac{\partial \text{FD}}{\partial \tau_q}$ in a particular reference frame introduced later on. We also define the *inverse dynamics* function

$$\text{ID} : \begin{cases} \mathbb{R}^{n_q} \times \mathbb{R}^{2n_v} \times \mathbb{R}^m & \to \mathbb{R}^{n_v} \\ (q, v_q, a_q, \lambda) & \mapsto \tau_q = Ma_q + b - J^T\lambda \end{cases} \tag{5}$$

B. Differentiating the constrained dynamics

Let us now recall how the derivatives of FD can be expressed in terms of the derivatives of ID and of the contact acceleration [24]. Without loss of generality, we assume $\alpha_* = 0$. Let z be any of the variables q, v_q, τ_q. The derivatives of (4) are given by the following proposition

Proposition 1:

$$\frac{\partial \text{FD}}{\partial z} = \begin{cases} -K^{-1}\begin{bmatrix} \frac{\partial}{\partial z}\text{ID}(q, v_q, a_q, \lambda) \\ \frac{\partial \alpha}{\partial z} \end{bmatrix} & \text{if } z \equiv q \text{ or } v_q \\ -K^{-1}\begin{bmatrix} -I_{n_v} \\ 0 \end{bmatrix} & \text{if } z \equiv \tau_q \end{cases} \tag{6}$$

Proof: We simply differentiate $y = \text{FD}(q, v_q, \tau_q)$. Let $'$ denote the differentiation operator w.r.t. z.

$$y' = -K^{-1}K'K^{-1}\sigma + K^{-1}\sigma' \tag{7}$$

$$y' = -K^{-1}(K'y - \sigma') \tag{8}$$

$$y' = -K^{-1}\left(K'\begin{bmatrix} a_q \\ -\lambda \end{bmatrix} - \begin{bmatrix} \tau_q' - b' \\ -\alpha_0' \end{bmatrix}\right) \tag{9}$$

$$y' = -K^{-1}\left(\begin{bmatrix} M' & J'^T \\ J' & 0 \end{bmatrix}\begin{bmatrix} a_q \\ -\lambda \end{bmatrix} - \begin{bmatrix} \tau_q' - b \\ -\alpha_0' \end{bmatrix}\right) \tag{10}$$

$$y' = -K^{-1}\begin{bmatrix} M'a_q - J'^T\lambda + b' - \tau_q' \\ J'a_q + \alpha_0' \end{bmatrix} \tag{11}$$

When $z \equiv \tau_q$, the lower part vanishes and the upper part is reduced to $-I_{n_v}$. When $\tau_q \equiv q$ or v_q, we recognize the derivatives of (5) and (2b) w.r.t. q and v_q. ■

C. Spatial algebra tools

We remind here important definitions of spatial algebra[1]. A placement is an element $\boldsymbol{X}$ of the special Euclidean group $\mathbb{SE}(3)$. Given two frames $\mathcal{A}, \mathcal{B}$, the transformation of motions from $\mathcal{B}$ to $\mathcal{A}$ is characterized by ${}^{\mathcal{A}}\boldsymbol{X}_{\mathcal{B}}$

$${}^{\mathcal{A}}\boldsymbol{X}_{\mathcal{B}} \triangleq \begin{bmatrix} {}^{\mathcal{A}}R_{\mathcal{B}} & 0 \\ -{}^{\mathcal{A}}R_{\mathcal{B}}\,{}^{\mathcal{A}}p_{\mathcal{B}}^{\times} & {}^{\mathcal{A}}R_{\mathcal{B}} \end{bmatrix} \tag{12}$$

where ${}^{\mathcal{A}}R_{\mathcal{B}} \in \mathbb{SO}(3), {}^{\mathcal{A}}p_{\mathcal{B}} \in \mathbb{R}^3$ are the rotation and translation from frame $\mathcal{B}$ to frame $\mathcal{A}$ respectively, and $^{\times}$ transforms a vector in $\mathbb{R}^3$ into its associated skew-symmetric matrix. Forces are transformed with

$${}^{\mathcal{A}}\boldsymbol{X}^*_{\mathcal{B}} \triangleq \left({}^{\mathcal{A}}\boldsymbol{X}_{\mathcal{B}}\right)^{-T} \tag{13}$$

Spatial velocities and accelerations are motions in $\mathbb{M} \simeq \mathbb{R}^6$ denoted by $\boldsymbol{\nu} \in \mathbb{R}^6$, $\boldsymbol{\alpha} \in \mathbb{R}^6$

$$\boldsymbol{\nu} = \begin{bmatrix} \boldsymbol{\nu}_\omega \\ \boldsymbol{\nu}_v \end{bmatrix}, \quad \boldsymbol{\alpha} = \begin{bmatrix} \boldsymbol{\alpha}_\omega \\ \boldsymbol{\alpha}_v \end{bmatrix} \tag{14}$$

where $\boldsymbol{\nu}_\omega, \boldsymbol{\nu}_v \in \mathbb{R}^3$ represent the angular and linear velocities, $\boldsymbol{\alpha}_\omega, \boldsymbol{\alpha}_v \in \mathbb{R}^3$ are the angular and linear components of the spatial acceleration. The spatial velocity (resp. acceleration) of frame $\mathcal{B}$ expressed in frame $\mathcal{A}$ is denoted by ${}^{\mathcal{A}}\boldsymbol{\nu}_{\mathcal{B}}$ (resp. ${}^{\mathcal{A}}\boldsymbol{\alpha}_{\mathcal{B}}$). The spatial force in $\mathbb{F} \simeq \mathbb{R}^6$ is denoted by $\boldsymbol{\lambda}$

$$\boldsymbol{\lambda} = \begin{bmatrix} \boldsymbol{\lambda}_\eta \\ \boldsymbol{\lambda}_f \end{bmatrix} \tag{15}$$

[1]For a more detailed treatment the reader can refer to [9], Section 2.9.

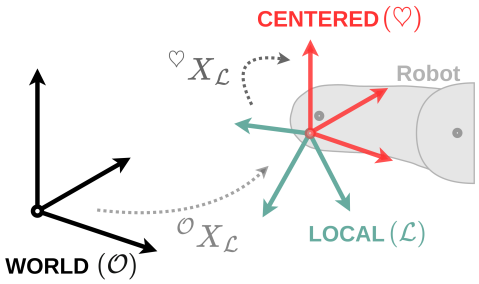


Fig. 1: Representation of the 3 reference frames

where $\boldsymbol{\lambda}_\eta, \boldsymbol{\lambda}_f \in \mathbb{R}^3$ represent the couple and linear force. The force acting at the center of frame $\mathcal{B}$ expressed in $\mathcal{A}$ by ${}^{\mathcal{A}}\boldsymbol{\lambda}_{\mathcal{B}}$. The subscript $\mathcal{B}$ will be omitted from spatial motions and forces when the frame of interest is not ambiguous. The spatial cross product operator and its adjoint are defined as

$$\boldsymbol{\nu}\times \triangleq \begin{bmatrix} \boldsymbol{\nu}_\omega^\times & 0 \\ \boldsymbol{\nu}_v^\times & \boldsymbol{\nu}_\omega^\times \end{bmatrix} \tag{16}$$

$$\boldsymbol{\nu}\times^* \triangleq -\boldsymbol{\nu}\times^T \tag{17}$$

We recall now the time derivative of a rigid transformation

Proposition 2: Let $\mathcal{A}, \mathcal{B}$ be arbitrary frames.

$$\frac{\partial}{\partial t}\left({}^{\mathcal{B}}\boldsymbol{X}_{\mathcal{A}}\right) = {}^{\mathcal{B}}(\boldsymbol{\nu}_{\mathcal{A}} - \boldsymbol{\nu}_{\mathcal{B}}) \times {}^{\mathcal{B}}\boldsymbol{X}_{\mathcal{A}} \tag{18}$$

The proof can be found in [9]. This proposition will be used to derive the derivatives of interest in the next section.

D. Reference frames convention

We define the following reference frames and their short names (see Figure 1). On purpose, we follow the conventions established in the software Pinocchio [10] where all the equations recalled in this section are efficiently implemented.

- WORLD ($\mathcal{O}$) : inertial frame, fixed w.r.t. the observer
- LOCAL ($\mathcal{L}$) : attached to a moving part of interest, typically to a joint of the robot (a.k.a. "body" frame)
- CENTERED (♡): centered at the local frame, but with axes oriented like WORLD at all times

The ♡ reference frame is the one in which we would like to express contact tasks. It is related to frames $\mathcal{O}$, $\mathcal{L}$ as follow

$${}^{\mathcal{O}}\boldsymbol{X}_{\heartsuit}(q) = \begin{bmatrix} I & 0 \\ -p^\times & I \end{bmatrix}, \quad {}^{\heartsuit}\boldsymbol{X}_{\mathcal{L}}(q) = \begin{bmatrix} R & 0 \\ 0 & R \end{bmatrix} \tag{19}$$

where $R = {}^{\mathcal{O}}R_{\mathcal{L}}(q)$ is the rotation from $\mathcal{L}$ to $\mathcal{O}$ and $p = {}^{\mathcal{O}}p_{\mathcal{L}}(q)$ is the translation from $\mathcal{L}$ to $\mathcal{O}$.

III. DERIVATIVES OF THE CONSTRAINED DYNAMICS IN THE ♡ FRAME

Note that Proposition 1 is agnostic to the reference frame. In this section, we show how the analytical derivatives of inverse dynamics and contact acceleration used in (6) can be derived in the ♡ frame by using their counterparts known in $\mathcal{L}$ [24]. Our analysis focuses on the generic case of a single $6D$ contact ($m = 6$)[2] such that the m-dimensional vectors α, λ introduced in II-A correspond to spatial quantities $\boldsymbol{\alpha}, \boldsymbol{\lambda}$.

A. Derivative of the spatial acceleration in ♡

The following proposition provides the derivatives of ${}^{\heartsuit}\boldsymbol{\alpha}$ (spatial acceleration of $\mathcal{L}$ expressed in ♡) in terms of the derivatives of ${}^{\mathcal{L}}\boldsymbol{\alpha}$ (spatial acceleration of $\mathcal{L}$ expressed in $\mathcal{L}$).

Proposition 3:

$$\frac{\partial {}^{\heartsuit}\boldsymbol{\alpha}}{\partial \tau_q} = 0 \tag{20}$$

$$\frac{\partial {}^{\heartsuit}\boldsymbol{\alpha}}{\partial v_q} = {}^{\heartsuit}\boldsymbol{X}_{\mathcal{L}} \frac{\partial {}^{\mathcal{L}}\boldsymbol{\alpha}}{\partial v_q} \tag{21}$$

$$\frac{\partial {}^{\heartsuit}\boldsymbol{\alpha}}{\partial q} = {}^{\heartsuit}\boldsymbol{X}_{\mathcal{L}} \frac{\partial {}^{\mathcal{L}}\boldsymbol{\alpha}}{\partial q} - \begin{bmatrix} {}^{\heartsuit}\boldsymbol{\alpha}_\omega^\times R^{\mathcal{L}} J_\omega \\ {}^{\heartsuit}\boldsymbol{\alpha}_v^\times R^{\mathcal{L}} J_\omega \end{bmatrix} \tag{22}$$

where ${}^{\mathcal{L}}J_\omega$ represents the angular part (i.e. top 3 rows) of the local contact Jacobian.

Proof: The derivatives w.r.t. τ_q is trivially 0 since $\boldsymbol{\alpha}$ doesn't depend on τ_q. For q, v_q we differentiate

$${}^{\heartsuit}\boldsymbol{\alpha} = {}^{\heartsuit}\boldsymbol{X}_{\mathcal{L}}(q)\,{}^{\mathcal{L}}\boldsymbol{\alpha} \tag{23}$$

$${}^{\heartsuit}\boldsymbol{\alpha}' = {}^{\heartsuit}\boldsymbol{X}_{\mathcal{L}}(q)\,{}^{\mathcal{L}}\boldsymbol{\alpha}' + {}^{\heartsuit}\boldsymbol{X}_{\mathcal{L}}(q)'\,{}^{\mathcal{L}}\boldsymbol{\alpha} \tag{24}$$

${}^{\heartsuit}\boldsymbol{X}_{\mathcal{L}}(q)$ does not depend on v_q so the second equality follows. For q, the second term involves a tensor-vector product that can be evaluated directly using Proposition 2:

$$\frac{\partial}{\partial t}\left({}^{\heartsuit}\boldsymbol{X}_{\mathcal{L}}\right) = {}^{\heartsuit}(\boldsymbol{\nu} - \boldsymbol{\nu}_{\heartsuit}) \times {}^{\heartsuit}\boldsymbol{X}_{\mathcal{L}} \tag{25}$$

where $\boldsymbol{\nu}, \boldsymbol{\nu}_{\heartsuit}$ are the spatial velocities of frames $\mathcal{L}$ and ♡ respectively. Moreover, by definition of the frame ♡

$${}^{\heartsuit}\boldsymbol{\nu} = \begin{bmatrix} {}^{\mathcal{O}}\boldsymbol{\nu}_\omega \\ R^{\mathcal{L}}\boldsymbol{\nu}_v \end{bmatrix}, \quad {}^{\heartsuit}\boldsymbol{\nu}_{\heartsuit} = \begin{bmatrix} 0 \\ R^{\mathcal{L}}\boldsymbol{\nu}_v \end{bmatrix} \tag{26}$$

Injecting Eq. (26) in (25) and using the anti-commutativity of the spatial cross product

$$\frac{\partial}{\partial t}\left({}^{\heartsuit}\boldsymbol{X}_{\mathcal{L}}\right){}^{\mathcal{L}}\boldsymbol{\alpha} = \begin{bmatrix} {}^{\mathcal{O}}\boldsymbol{\nu}_\omega \\ 0 \end{bmatrix} \times {}^{\heartsuit}\boldsymbol{X}_{\mathcal{L}}\,{}^{\mathcal{L}}\boldsymbol{\alpha} \tag{27}$$

$$= -{}^{\heartsuit}\boldsymbol{\alpha} \times \begin{bmatrix} {}^{\mathcal{O}}\boldsymbol{\nu}_\omega \\ 0 \end{bmatrix} \tag{28}$$

$$= \begin{bmatrix} -{}^{\heartsuit}\boldsymbol{\alpha}_\omega^\times R^{\mathcal{L}} J_\omega \\ -{}^{\heartsuit}\boldsymbol{\alpha}_v^\times R^{\mathcal{L}} J_\omega \end{bmatrix} v_q \tag{29}$$

We used ${}^{\mathcal{O}}\boldsymbol{\nu}_\omega = R^{\mathcal{L}}\boldsymbol{\nu}_\omega$ and the Jacobian definition ${}^{\mathcal{L}}\boldsymbol{\nu}_\omega = {}^{\mathcal{L}}J_\omega v_q$. Observing that $v_q = \frac{\partial q}{\partial t}$, the result follows. ■

B. Derivative of ID in ♡

The derivatives of (5) in ♡ are given in terms of the local ID derivatives by the following proposition.

[2]Adding more contacts is not an issue - they simply need to be stacked properly in the KKT system and in the derivatives. The case of lower dimensional constraints is discussed in Section III-C.

Proposition 4:

$$\frac{\partial}{\partial \tau_q}\left({}^{\heartsuit}\mathrm{ID}\right) = 0 \tag{30}$$

$$\frac{\partial}{\partial v_q}\left({}^{\heartsuit}\mathrm{ID}\right) = \frac{\partial}{\partial v_q}\left({}^{\mathcal{L}}\mathrm{ID}\right) \tag{31}$$

$$\frac{\partial}{\partial q}\left({}^{\heartsuit}\mathrm{ID}\right) = \frac{\partial}{\partial q}\left({}^{\mathcal{L}}\mathrm{ID}\right) - {}^{\mathcal{L}}J^T \begin{bmatrix} {}^{\mathcal{L}}\boldsymbol{\lambda}_\eta^{\times^{\mathcal{L}}} J_\omega \\ {}^{\mathcal{L}}\boldsymbol{\lambda}_f^{\times^{\mathcal{L}}} J_\omega \end{bmatrix} \tag{32}$$

Proof: By definition, ID in $\heartsuit$ and its derivative are given by

$${}^{\heartsuit}\mathrm{ID}(q, v_q, a_q, {}^{\heartsuit}\boldsymbol{\lambda}) = Ma_q + b - {}^{\heartsuit}J^T {}^{\heartsuit}\boldsymbol{\lambda} \tag{33}$$

$${}^{\heartsuit}\mathrm{ID}'(q, v_q, a_q, {}^{\heartsuit}\boldsymbol{\lambda}) = M'a_q + b' - \left({}^{\heartsuit}J^T\right)' {}^{\heartsuit}\boldsymbol{\lambda} \tag{34}$$

where a_q, ${}^{\heartsuit}\boldsymbol{\lambda}$ are considered **constant** during the differentiation. The differential w.r.t. τ_q is trivially 0. For q, v_q, we observe that the only terms depending on the reference frame are the contact wrench $\boldsymbol{\lambda}$ and the contact Jacobian. Using that ${}^{\heartsuit}J = {}^{\heartsuit}\boldsymbol{X}_{\mathcal{L}}\, {}^{\mathcal{L}}J$, we have

$$\left({}^{\heartsuit}J^T\right)' {}^{\heartsuit}\boldsymbol{\lambda} = \left({}^{\mathcal{L}}J^T {}^{\heartsuit}\boldsymbol{X}_{\mathcal{L}}^T\right)' {}^{\heartsuit}\boldsymbol{\lambda} \tag{35}$$

$$= {}^{\mathcal{L}}\left(J^T\right)' {}^{\heartsuit}\boldsymbol{X}_{\mathcal{L}}^T {}^{\heartsuit}\boldsymbol{\lambda} + {}^{\mathcal{L}}J^T \left({}^{\heartsuit}\boldsymbol{X}_{\mathcal{L}}^T\right)' {}^{\heartsuit}\boldsymbol{\lambda} \tag{36}$$

The first term is computed naturally with the local derivatives of ID. Since ${}^{\heartsuit}\boldsymbol{X}_{\mathcal{L}}^T$ in the second term doesn't depend on v_q, the second equality follows. For q, we observe that ${}^{\heartsuit}\boldsymbol{X}_{\mathcal{L}}^T = {}^{\mathcal{L}}\boldsymbol{X}_{\heartsuit}$. Let us use Proposition 2 for the tensor-vector product

$$\frac{\partial}{\partial t}\left({}^{\heartsuit}\boldsymbol{X}_{\mathcal{L}}^T\right) = \frac{\partial}{\partial t}\left({}^{\mathcal{L}}\boldsymbol{X}_{\heartsuit}\right) \tag{37}$$

$$= {}^{\mathcal{L}}(\boldsymbol{\nu}_{\heartsuit} - \boldsymbol{\nu}) \times {}^{\mathcal{L}}\boldsymbol{X}_{\heartsuit} \tag{38}$$

The spatial velocities of $\heartsuit$ and $\mathcal{L}$ expressed in $\mathcal{L}$ are

$${}^{\mathcal{L}}\boldsymbol{\nu}_{\heartsuit} = \begin{bmatrix} 0 \\ {}^{\mathcal{L}}\boldsymbol{\nu}_v \end{bmatrix}, \quad {}^{\mathcal{L}}\boldsymbol{\nu} = \begin{bmatrix} {}^{\mathcal{L}}\boldsymbol{\nu}_\omega \\ {}^{\mathcal{L}}\boldsymbol{\nu}_v \end{bmatrix} \tag{39}$$

Plugging expressions (39) in (38)

$$\frac{\partial}{\partial t}\left({}^{\heartsuit}\boldsymbol{X}_{\mathcal{L}}^T\right) {}^{\heartsuit}\boldsymbol{\lambda} = \begin{bmatrix} -{}^{\mathcal{L}}\boldsymbol{\nu}_\omega \\ 0 \end{bmatrix} \times {}^{\mathcal{L}}\boldsymbol{X}_{\heartsuit}\, {}^{\heartsuit}\boldsymbol{\lambda} \tag{40}$$

Let us notice that ${}^{\mathcal{L}}\boldsymbol{X}_{\heartsuit} = {}^{\mathcal{L}}\boldsymbol{X}_{\heartsuit}^*$ since ${}^{\mathcal{L}}\boldsymbol{X}_{\heartsuit}$ is orthogonal. So ${}^{\mathcal{L}}\boldsymbol{X}_{\heartsuit}\, {}^{\heartsuit}\boldsymbol{\lambda} = {}^{\mathcal{L}}\boldsymbol{X}_{\heartsuit}^*\, {}^{\heartsuit}\boldsymbol{\lambda} = {}^{\mathcal{L}}\boldsymbol{\lambda}$ and

$$\frac{\partial}{\partial t}\left({}^{\heartsuit}\boldsymbol{X}_{\mathcal{L}}^T\right) {}^{\heartsuit}\boldsymbol{\lambda} = \begin{bmatrix} -{}^{\mathcal{L}}\boldsymbol{\nu}_\omega \\ 0 \end{bmatrix} \times {}^{\mathcal{L}}\boldsymbol{\lambda} \tag{41}$$

$$= {}^{\mathcal{L}}\boldsymbol{\lambda} \times \begin{bmatrix} {}^{\mathcal{L}}\boldsymbol{\nu}_\omega \\ 0 \end{bmatrix} \tag{42}$$

$$= \begin{bmatrix} {}^{\mathcal{L}}\boldsymbol{\lambda}_\eta^{\times^{\mathcal{L}}} J_\omega \\ {}^{\mathcal{L}}\boldsymbol{\lambda}_f^{\times^{\mathcal{L}}} J_\omega \end{bmatrix} v_q \tag{43}$$

Finally, the result follows from observing that $v_q = \frac{\partial q}{\partial t}$. ■

C. Projecting onto lower dimensional constraints

The case of an m-dimensional constraint for $m < 6$ can be deduced by projections from the $6D$ case. It can be useful, as illustrated in our experiments, to express tasks where the motion is constrained only in some directions. Let us define the set $I \subset \{1, ..., 6\}$ of m dimensions. The constrained forward dynamics read

$$\begin{bmatrix} a_q \\ -\boldsymbol{\lambda}_I \end{bmatrix} = K^{-1} \begin{bmatrix} \tau_q - b \\ \boldsymbol{\alpha}_{*,I} - \boldsymbol{\alpha}_{0,I} \end{bmatrix} \tag{44}$$

were the subscript $\boldsymbol{\lambda}_I$ means the m-dimensional sub-vector obtained by masking indices in $\boldsymbol{\lambda}$ that are not in I. Following this notation, the constrained forward dynamics function is

$$\mathrm{FD}_I(q, v_q, \tau_q) \triangleq \mathbb{P}_I\left(\mathrm{FD}(q, v_q, \tau_q)\right) \tag{45}$$

where $\mathbb{P}_I : \mathbb{R}^{n_v} \times \mathbb{R}^m \to \mathbb{R}^{n_v} \times \mathbb{R}^m$ represents the projection onto dimensions $i \in I$, i.e. $\mathbb{P}_I\left(a_q, \boldsymbol{\lambda}\right) \triangleq \left(a_q, \boldsymbol{\lambda}_I\right)$.

IV. Experimental results

This section demonstrates the relevance of our method on a polishing task with the torque-controlled humanoid Talos.

A. Experimental setup

The upper body (torso + right arm) is controlled in MPC while the lower body is maintained in a fixed posture using position control, so we have $n_q = n_v = 6$. The DDP implementation is used to solve the optimal control problem is available in Crocoddyl[3] and the rigid-body dynamics computations are performed in Pinocchio[4]. The analytical derivatives of the constrained forward dynamics in the $\heartsuit$ frame which were derived in Section III, are implemented in the open-source library Sobec[5]. The task is exert with the right wrist a constant normal force on a flat rigid surface while drawing a circle. This task is demonstrated on both horizontal and vertical surfaces. For this task, the desired contact force is naturally expressed in the $\heartsuit$ frame, as the local frame attached to the wrist of the robot is not necessarily aligned with the surface normal direction.

B. MPC formulation

We use the force feedback MPC formulated in [11], which solves the following Optimal Control Problem (OCP)

$$\min_{w(.), z(.)} \int_0^T L(z(t), w(t), \lambda(t), t)dt + L_T(z(T)) \tag{46}$$

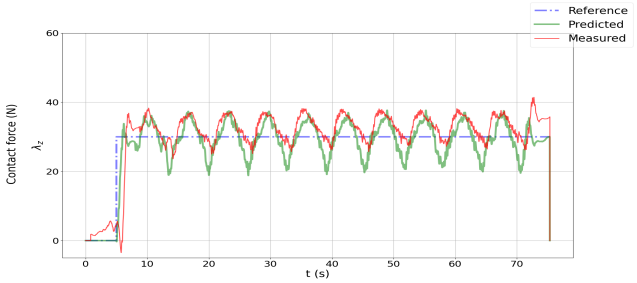
$$\text{s.t.} \begin{cases} \dot{z}(t), \lambda(t) & = F(z(t), w(t)) \\ z(0) & = z_0 \\ z(t) \in \mathcal{Z}, & w(t) \in \mathcal{W} \end{cases}$$

where $z \triangleq (q, v_q, \tau_q)$ is the state, w is the control (a.k.a computed) torque, z_0 is the initial state, L, L_T the running and terminal costs, $\mathcal{Z}, \mathcal{W}$ represent state and control constraints. The dynamics F gathers the constrained forward dynamics

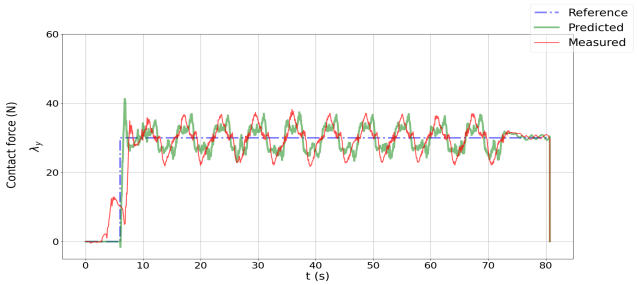
[3] https://github.com/loco-3d/crocoddyl

[4] https://github.com/stack-of-tasks/pinocchio

[5] https://github.com/MeMory-of-MOtion/sobec



(a) Contact force in the z direction of the (centered) ♡ frame for the sanding task on a horizontal table



(b) Contact force in the y direction of the (centered) ♡ frame for the sanding task on a vertical board

Fig. 2: Sanding task on a horizontal table (top) and on a vertical board (bottom). Analytical derivatives of the constrained forward dynamics in the ♡ frame enables to easily design and realize a contact task using nonlinear MPC.

and a 1^{st}-order low-pass filter on the torques with angular cut-off frequency ω_0, i.e. $\dot{z}(t), \lambda(t)$ are such that

$$\dot{q}(t) = v_q(t) \tag{47}$$

$$\dot{v}_q(t), \lambda(t) = \text{FD}_I\left(q(t), v_q(t), \tau_q(t)\right) \tag{48}$$

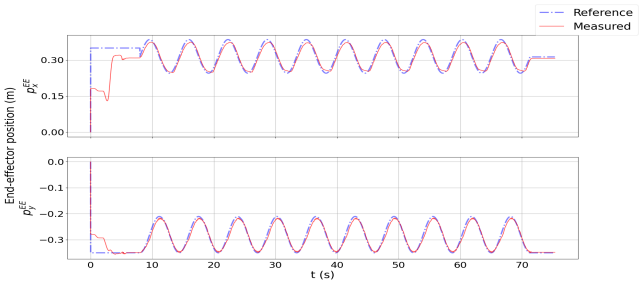
$$\dot{\tau}_q(t) = \omega_0(w(t) - \tau_q(t)) \tag{49}$$

The contact model used in this case is $1D$ ($m = 1$), i.e. the robot is free to move in the plane orthogonal to the surface normal. The definition of the index set I depends on the experiment: when the task is to maintain contact with a horizontal plane, then $I = \{3\}$ (constraint acting along the z-dimension in ♡), whereas for the vertical plane contact, $I = \{2\}$ (constraint acting along the y-dimension in ♡). The cost function describing the sanding task has the form

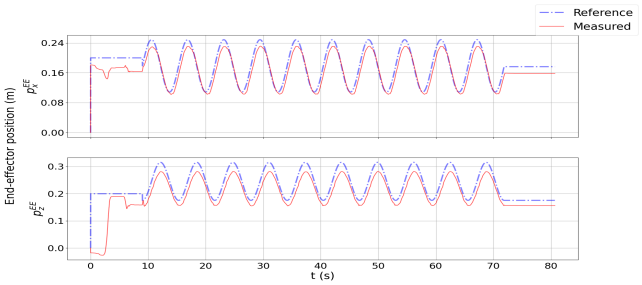
$$\begin{aligned} L(z, w, t) = \; & c_z \|A_z(z - z_0)\|^2 + c_z^{lim}\|B_z(z)\|^2 + \\ & c_\lambda \|A_\lambda({}^{\heartsuit}\lambda - {}^{\heartsuit}\bar{\lambda})\|^2 + c_R\|A_R(R(q) \ominus \bar{R})\|^2 + \\ & c_p\|A_p(p(q) - \bar{p}(t))\|^2 + \\ & c_w\|A_w w\|^2 + c_w^{lim}\|B_w(w)\|^2 \end{aligned} \tag{50}$$

where p, R are the end-effector frame $3D$ position and rotation respectively, $A_z, A_w, A_p, A_\lambda, A_R$ are activation weight matrices, $c_z, c_z^{lim}, c_w, c_w^{lim}, c_\lambda, c_p, c_R$ are scalar costs weights, $\ominus$ represents the difference in $\mathbb{SO}(3)$, B_z, B_w are weighted quadratic barriers. The reference force to exert on the contact surface is ${}^{\heartsuit}\bar{\lambda}$, expressed in the centered frame and the trajectory to track on the surface plane is $\bar{p}(t)$.

Fig. 3: Screenshot of the horizontal (top) and vertical (bottom) sanding tasks.



(a) End-effector position in the x, y-plane of the centered ♡ frame for the horizontal surface sanding



(b) End-effector position in the x, y-plane of the centered ♡ frame for the vertical surface sanding

Fig. 4: End-effector trajectories in the surface plane for horizontal (top) and vertical (bottom) sanding tasks

C. Results and discussion

The OCP discretization is set to $10\,\text{ms}$ and the MPC update rate to $100\,\text{Hz}$. The Riccati gains in position and velocity output by DDP are used to interpolate the solution at $2\,\text{kHz}$ [4]. The target force was set to $\bar{\lambda} = 30\,\text{N}$ and the reference trajectory of the end-effector is a circle of radius $7\,\text{cm}$ and angular velocity $1\,\text{rad}\,\text{s}^{-1}$. The estimated

contact force is shown Figures 2a and 2b and the end-effector position in Figures 4a, 4b. The contact forces are estimated from joint torques sensors, joint position encoder measurements and joint velocities computed through finite differences using the forward dynamics. We can see that the robot is able to track the forces expressed in the ♡ frame. It is important to note that without the mathematical developments presented in this paper, it would have been impossible to specify the task directly in this frame. We would have had to make the wrist orientation as *part of the main task*, i.e. to hard-code an operational frame that must remain oriented like the world frame at all times. This imposes an additional constraint on the robot's motion and makes the task more difficult to realize. By expressing the contact task directly in the appropriate frame, the controller is free to trade-off wrist orientation against force tracking.

D. Extension and implementation

The model covered in this paper is only considering the case where the contact occurs between a robot body and the world. It is implemented for the 3 proposed frames LOCAL, CENTERED and WORLD in Sobec[5]. It can be generalized to the universal case where the contact occurs between two frames $\mathcal{C}^1$ and $\mathcal{C}^2$ that can both be either attached to a moving body or fixed in the world (not limiting to one contact being fixed). We then recover the LOCAL formulation when $\mathcal{C}^1$ is attached to the robot and $\mathcal{C}^2$ is fixed, and the CENTERED formulation when $\mathcal{C}^2$ is attached to the robot and $\mathcal{C}^1$ is fixed. This extension has been chosen for the implementation of the constraint dynamics soon released in Pinocchio v3. The extension of the model proposed here to this universal description is handled in an extended version of this paper[6].

V. CONCLUSION

In this paper we derived derivatives of the contact acceleration and forces to perform contact tasks in MPC in an arbitrary frame. The main outcome of this study is that nonlinear skew terms need to be added to the rotated local derivatives. We showed experimentally that our calculations enable to exert forces in arbitrary directions. We hope this paper will help MPC practitioners to design contact tasks without having to go through these tedious calculations. It is also worth emphasizing that the centered frame can be in fact *any* frame deemed relevant for the task by the user (only the rotation matrix needs to be changed, not the maths). As future work we intend to include these derivatives more systematically in our software stack and propagate their use to locomotion and whole-body multi-contact problems. Another interesting lead can be to optimize the computation of the projected constraint derivatives.

ACKNOWLEDGMENT

The authors would like to thank Nahuel Villa, Côme Perrot and Pierre Fernbach for helping in the experiments and Guilhem Saurel for his amazing support in software development.

[6] https://hal.archives-ouvertes.fr/hal-03758989

REFERENCES

[1] D. Q. Mayne, J. B. Rawlings *et al.*, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, jun 2000.

[2] C. Mastalli, R. Budhiraja *et al.*, "Crocoddyl: An efficient and versatile framework for multi-contact optimal control," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[3] A. Meduri, P. Shah *et al.*, "Biconmp: A nonlinear model predictive control framework for whole body motion planning," 2022. [Online]. Available: https://arxiv.org/pdf/2201.07601

[4] E. Dantec, M. Taïx *et al.*, "First order approximation of model predictive control solutions for high frequency feedback," *IEEE Robotics and Automation Letters (RAL)*, vol. 7, no. 2, 2022.

[5] S. Kleff, A. Meduri *et al.*, "High-frequency nonlinear model predictive control of a manipulator," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[6] F. Farshidian, E. Jelavic *et al.*, "Real-time motion planning of legged robots: A model predictive control approach," in *IEEE-RAS International Conference on Humanoid Robotics (Humanoids)*, 2017.

[7] M. Neunert, M. Stäuble *et al.*, "Whole-body nonlinear model predictive control through contacts for quadrupeds," *IEEE Robotics and Automation Letters (RAL)*, vol. 3, no. 3, 2018.

[8] J. Marti-Saumell, J. Solà *et al.*, "Full-body torque-level non-linear model predictive control for aerial manipulation," 2021. [Online]. Available: https://arxiv.org/abs/2107.03722

[9] R. Featherstone, *Rigid Body Dynamics Algorithms*. Berlin, Heidelberg: Springer-Verlag, 2007.

[10] J. Carpentier, G. Saurel *et al.*, "The Pinocchio C++ library: A fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives," in *IEEE/SICE International Symposium on System Integration*, 2019.

[11] S. Kleff, E. Dantec *et al.*, "Introducing Force Feedback in Model Predictive Control," in *IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[12] J.-P. Sleiman, F. Farshidian *et al.*, "A unified mpc framework for whole-body dynamic locomotion and manipulation," *IEEE Robotics and Automation Letters*, vol. 6, pp. 4688–4695, 2021.

[13] M. Galliker, N. Csomay-Shanklin *et al.*, "Bipedal locomotion with nonlinear model predictive control: Online gait generation using whole-body dynamics," 2022. [Online]. Available: https://arxiv.org/pdf/2203.07429

[14] T. Gold, A. Völz *et al.*, "Model Predictive Interaction Control for Industrial Robots," *IFAC-PapersOnLine*, vol. 53, no. 2, 2020.

[15] T. Erez, K. Lowrey *et al.*, "An integrated system for real-time model predictive control of humanoid robots," in *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2013.

[16] E. Todorov, T. Erez *et al.*, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012, pp. 5026–5033.

[17] I. Mordatch, E. Todorov *et al.*, "Discovery of complex behaviors through contact-invariant optimization," *ACM Transactions on Graphics (TOG)*, vol. 31, pp. 1 – 8, 2012.

[18] S. L. Cleac'h, T. Howell *et al.*, "Fast contact-implicit model-predictive control," 2021. [Online]. Available: https://arxiv.org/abs/2107.05616

[19] K. Werling, D. Omens *et al.*, "Fast and feature-complete differentiable physics for articulated rigid bodies with contact," *ArXiv*, 2021. [Online]. Available: https://arxiv.org/abs/2103.16021

[20] T. A. Howell, S. L. Cleac'h *et al.*, "Dojo: A differentiable physics engine for robotics," 2022. [Online]. Available: https://arxiv.org/abs/2203.00806

[21] Q. Le Lidec, I. Kalevatykh *et al.*, "Differentiable simulation for physical system identification," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, 2021.

[22] R. Budhiraja, J. Carpentier *et al.*, "Differential dynamic programming for multi-phase rigid contact dynamics," in *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2018.

[23] F. E. Udwadia and R. E. Kalaba, "A new perspective on constrained motion," *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences*, vol. 439, no. 1906, pp. 407–410, 1992.

[24] J. Carpentier and N. Mansard, "Analytical derivatives of rigid body dynamics algorithms," *Robotics: Science and Systems*, 2018.

[25] J. Carpentier, R. Budhiraja *et al.*, "Proximal and sparse resolution of constrained dynamic equations," *Robotics: Science and Systems*, 2021.