2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Geometric Slosh-Free Tracking for Robotic Manipulators

Jon Arrizabalaga[1], Lukas Pries[1], Riddhiman Laha[2], Runkang Li[2], Sami Haddadin[2], Markus Ryll[1,2]

***Abstract*— This work focuses on the agile transportation of liquids with robotic manipulators. In contrast to existing methods that are either computationally heavy, system/container specific or dependant on a singularity-prone pendulum model, we present a real-time slosh-free tracking technique. This method solely requires the reference trajectory and the robot's kinematic constraints to output kinematically feasible joint space commands. The crucial element underlying this approach consists on mimicking the end-effector's motion through a virtual quadrotor, which is inherently slosh-free and differentially flat, thereby allowing us to calculate a slosh-free reference orientation. Through the utilization of a cascaded proportional-derivative (PD) controller, this slosh-free reference is transformed into task space acceleration commands, which, following the resolution of a Quadratic Program (QP) based on Resolved Acceleration Control (RAC), are translated into a feasible joint configuration. The validity of the proposed approach is demonstrated by simulated and real-world experiments on a 7 DoF Franka Emika Panda robot.**

Code: `https://github.com/jonarriza96/gsft`
Video: `https://youtu.be/4kitqYVS9n8`

I. Introduction

Slosh-free liquid handling is of great importance in various fields. For example, it plays a pivotal role in the design of earthquake-resistant watertanks or transportation systems, particularly when handling fuel in planes, propellants in spacecrafts or navigating through waves in ships that have tanks on their decks [1]. The use of slosh-free motions is also critical within industrial assembly lines or the healthcare sector, where robotic manipulators are often required to seamlessly mix, transport, or pour liquids [2].

In such manipulation scenarios achieving agile slosh-free maneuverability poses a significant challenge. Humans require exceptional skill to perform such motions effectively. For instance, only the most skilled waiters and waitresses can transport multiple dishes and glasses without spilling any food or liquid. Nevertheless, by utilizing advanced control methods and leveraging the enhanced agility of robotic systems, there exists potential to realize super-human slosh-free motions. Doing so, offers an appealing incentive to the aforementioned applications. Driven by this goal, in this work we focus on the problem of slosh-free tracking with robotic manipulators.

Starting with a purely-scientific interest that led to the formulation of the Navier-Stokes equations [3], and further motivated by the economic benefits of multiple industries [1], [2], slosh-free motion control is a well-studied and long-sought problem. In the field of robotics, the existing literature

[1]Autonomous Aerial Systems, School of Engineering and Design, Technical University of Munich, Germany. E-mail: `jon.arrizabalaga@tum.de` and `markus.ryll@tum.de`
[2]Munich Institute of Robotics and Machine Intelligence (MIRMI), Technical University of Munich

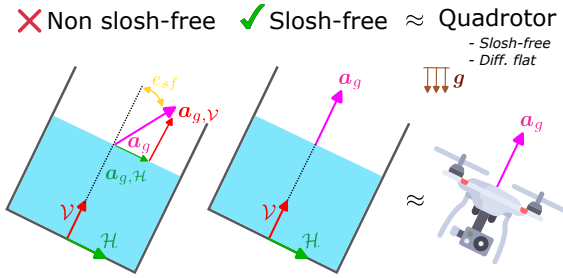

Fig. 1: A planar diagram showcasing the slosh-free condition and its equivalence to the quadrotor. Similar to the existing literature our method couples the longitudinal and rotational accelerations in the container to ensure that the resultant translational acceleration acting on the liquid (in magenta) is perpendicular to the liquid's surface. This implies that the slosh-free angle error e_{sf} (in yellow) is desired to be 0. For this purpose, we emulate the motion of the container with a virtual quadrotor, whose resultant acceleration aligns with the vertical component (in red), and thus, guarantees to be slosh-free.

on slosh-free tracking methods with robotic manipulators can be divided into four categories.

The first category relates to the white-box *fluid analysis / modelling* approach, in which the dynamics of the liquid are approximated by models created using complex methods such as Computational Fluid Dynamics (CFD) [4], [5] or particle-based approaches [6], [7]. Once generated, these models can be used for slosh-free trajectory planning and control [6], [8]. The computational demands involved in generating such detailed models not only hinder their immediate deployment but also render them case-specific. These models rely on various parameters, including liquid viscosity, density, and vessel size and shape. The second category takes the opposite black-box *learning based* approach, where instead of relying on physics to model the liquid's behavior, the slosh-free motions are Learned from Demonstrations (LfD) [9], [10]. These methods are also system-specific since they demand extensive training and are exclusively applicable to the system they were trained for.

The third category encompasses the *motion / smoothness minimization* methods. These rely on a first-principles model built from the Navier-Stokes equations, and aim to minimize the slosh-generating oscillations either by using polynomials to shape the input commands [11]–[13], designing smooth velocity profiles [14] or formulating filtering feedback controllers in the frequency domain [15]. Notice that the first two approaches are feedforward, making them vulnerable to unmodeled dynamics and disturbances, whereas the latter requires feedback from the motion of the liquid, thereby creating a dependency on highly specialized sensors designed to measure the state of the liquid. Additionally, these techniques exclusively pertain to translational movements and, as such, are unsuitable for agile motions, as they do not permit tilting of the vessel.

979-8-3503-8457-4/24/$31.00 ©2024 IEEE

The fourth category overcomes this limitation by coupling the container's translation and rotation by means of a *spherical pendulum model*. These methods cancel out the lateral accelerations acting on the container by emulating the liquid's motion as a virtual pendulum model. Since the original presentation of this technique in [16], [17], further studies have allowed for a better understanding on why the augmented pendulum model results in slosh-free motions [18], [19].

Out of the extensive literature that falls under the fourth category, two works have shown very appealing results. First, in [20] a two-stage plug-in allows to modify the task space commands before being fed to the joint space controller: After passing the desired motion commands through a filter that ensures its smoothness and continuity – similar to the third category–, a spherical pendulum model is used for calculating the vessel's desired orientation, allowing for compensation of the slosh-generating lateral accelerations. In a similar manner, the authors in the second work [21] linearize the spherical pendulum model in its lower equilibrium part, resulting in an optimization-based real-time task-space controller for planar (2D) trajectories. Despite these achievements, these two works share three main drawbacks: First, the analytical simplicity of the pendulum's angular parameterization comes at the expense of singularities, preventing it from being applicable to all $\mathrm{SO}(3)$ [22]. Second, the tracking quality is dependant on the controller design, i.e., the damping-frequency in the first work and the pendulum's rod-length in the second, rendering the overall performance very sensitive to parameter selection. Third, these methods do not tackle the joint space control, leaving it up to inverse kinematics, which is computationally expensive.

This raises the question on how to design a slosh-free tracking method that is (i) singularity-free within all SE3, i.e., spatial (3D) end-effector trajectories in $\mathbb{R}^3$ with a continuous coverage of SO3, (ii) outputs joint commands that fulfill the robot's constraints, (iii) runs online in real-time and (iv) is generic with respect to the liquid properties, cup shape or robot specific parameters.

To address this question, we present a slosh-free tracking algorithm that, similar to [20], [21], leverages the tilt compensation mechanism. However, instead of relying on the spherical pendulum model, we emulate the end-effector's motion by utilizing the model of a quadrotor. When doing so, we can leverage the differential flatness property of a quadrotor to compute a slosh-free reference pose, i.e. position and orientation. This reference is converted to the joint space by two consecutive controllers, which are not only computationally lightweight but also enforce the robot's kinematic constraints. To the best of the authors' knowledge this is the first method that enables real-time slosh-free tracking of any 3D trajectory with robotic manipulators.

To achieve this, our method comprises three main ingredients: 1) Using the quadrotor's differential flatness [23], we compute a slosh-free translation and orientation reference. 2) Subsequently, we implement a cascaded proportional derivative (PD) controller to track the desired reference in the task space [24]. 3) Last but not least, we utilize Resolved-Acceleration Control (RAC) [25] to formulate a convex Quadratic Program (QP) that maps the desired task space accelerations to joint space while satisfying the robot's kinematic constraints.

More in detail, we make the following contributions:

1) We propose a novel perspective on the problem of slosh-free tracking by identifying the appropriateness of differential-flatness based trajectories from the domain of mobile robotics.
2) We formulate an entire pipeline for slosh-free tracking from a desired reference to joint space commands that is (i) capable of tracking any 3D references – even if infeasible –, (ii) deployable in real-time, (iii) system agnostic and (iv) compliant with the robot's kinematic constraints.

The reminder of this paper is organized as follows: Section II formally introduces the slosh-free tracking problem tackled in this work and, subsequently, Section III presents our solution by delving deeper into all three ingredients. Experimental results are shown in Section IV before Section V presents the conclusions.

II. The Slosh-Free Tracking Problem

A. Robotic manipulator model

The forward kinematics of a robotic manipulator are given by a nonlinear mapping between the joint space $\mathcal{J}$ and the task space $\mathcal{T}$, expressed as:

$$\mathrm{T}_e(t) = \begin{bmatrix} \boldsymbol{p}_e(t) & \mathrm{R}_e(t) \\ \mathbf{0} & 1 \end{bmatrix} = \Upsilon(\boldsymbol{q}(t)) \,. \tag{1}$$

Here, $\mathcal{J} \coloneqq \boldsymbol{q}(t) \in \mathcal{Q} \in \mathbb{R}^n$ is the vector of the joint angles, where n is the number of joints, $\mathcal{Q}$ represents the set of kinematically feasible joint configurations and $\mathcal{T} \coloneqq \mathrm{T}_e(t) \in \mathbb{R}^{4x4} \in \mathrm{SE3}$ refers to the homogeneous transformation matrix that represents the end-effector's pose. Its position and orientation are denoted as $\boldsymbol{p}_e(t) \in \mathbb{R}^3$ and $\mathrm{R}_e(t) \in \mathrm{SO3}$, respectively. The nonlinear mapping $\Upsilon(\boldsymbol{q}(t)) : \mathbb{R}^n \mapsto \mathrm{SE3}$ can either be computed from Denavit-Hartenberg parameters or from elementary transform sequences [26].

Derivating (1) on time allows to compute the end effector's longitudinal and angular velocities $\{\boldsymbol{v}_e(t), \boldsymbol{\omega}_e(t)\} \in \mathbb{R}^3$ as:

$$\boldsymbol{\nu}_e(t) = [\boldsymbol{v}_e(t), \boldsymbol{\omega}_e(t)] = J(\boldsymbol{q}(t))\dot{\boldsymbol{q}}(t) \,, \tag{2}$$

where $J(\boldsymbol{q}(t)) \in \mathbb{R}^{6\times n}$ is the robot's jacobian. Further derivating (2) on time leads to the end-effector's longitudinal and angular accelerations $\{\boldsymbol{a}_e(t), \dot{\boldsymbol{\omega}}_e(t)\} \in \mathbb{R}^3$:

$$\boldsymbol{\alpha}_e(t) = \begin{bmatrix} \boldsymbol{a}_e(t) \\ \dot{\boldsymbol{\omega}}_e(t) \end{bmatrix} = J(\boldsymbol{q}(t))\ddot{\boldsymbol{q}}(t) + \dot{\boldsymbol{q}}(t) \otimes H(\boldsymbol{q}(t))\dot{\boldsymbol{q}}(t) \,, \tag{3}$$

where $H(\boldsymbol{q}(t)) \in \mathbb{R}^{n\times 6\times n}$ is the robot's hessian and $\otimes$ denotes the tensor contraction.

B. Reference trajectory

Let Γ refer to a geometric reference whose Cartesian coordinates are given by a time-based reference $\boldsymbol{p}_r : \mathbb{R} \mapsto \mathbb{R}^3$ that is, at least, $\mathcal{C}^4$:

$$\Gamma = \{t \in [t_0, t_f] \subseteq \mathbb{R} \mapsto \boldsymbol{p}_r(t) \in \mathbb{R}^3\} \,. \tag{4}$$

The position difference between the end-effector and the reference will be denoted as *position error* $e_p(t) \in \mathbb{R}$:

$$e_p(t) = ||\boldsymbol{p}_r(t) - \boldsymbol{p}_e(t)|| \,. \tag{5}$$

C. Problem statement

As initially proposed in [16], [17] and further discussed in [18], [19], sloshing can be prevented by suppressing horizontal accelerations acting on the container. This can be achieved by maintaining the surface of the liquid perpendicular to the resultant acceleration at all times. Notice that this resultant acceleration $\boldsymbol{a}_g(t) \in \mathbb{R}^3$ includes all accelerations acting on the liquid, and thus the gravity $\boldsymbol{g} \in \mathbb{R}^3$ also needs to be considered, i.e., $\boldsymbol{a}_g(t) = \boldsymbol{a}_e(t) + \boldsymbol{g} = \boldsymbol{a}_{g,\mathcal{H}}(t) + \boldsymbol{a}_{g,\mathcal{V}}(t)$, where $\boldsymbol{a}_{g,\mathcal{V}}(t)$ and $\boldsymbol{a}_{g,\mathcal{H}}(t)$ refer to the vertical and horizontal components, respectively. In addition, to quantify the slosh-freeness of a given acceleration, we define the *slosh-free error angle*, $e_{sf} \in \mathbb{R}$, as

$$\tan e_{sf}(t) = \frac{\boldsymbol{a}_{g,\mathcal{H}}(t)}{\boldsymbol{a}_{g,\mathcal{V}}(t)} \,. \tag{6}$$

For a better understanding of these terms, please see the illustrative planar diagram in Fig. 1.

Having defined the robotic manipulator in (1), the geometric reference Γ in (4), as well as the respective position and slosh-free angle errors in eqs. (5) and (6), we can now formally state the problem addressed in this paper:

Problem 1 (Slosh-free tracking for robotic manipulators): *Given a robotic manipulator whose forward kinematics are expressed in* (1)*, and the geometric reference* Γ *in* (4)*, formulate a control method that achieves:*

- P1.1 ***Reference tracking:*** *The end-effector tracks the moving reference within a desired tolerance* ϵ_p*, i.e.,* $e_p(t) \leq \epsilon_p \,; \forall t \in [t_0, t_f]$.
- P1.2 ***Slosh-free motions:*** *The tracking motions are conducted in a slosh-free manner, i.e.,* $|e_{sf}(t)| \leq \epsilon_{sf} \,; \forall t \in [t_0, t_f]$*, where* ϵ_{sf} *is sufficiently small not to generate any slosh.*
- P1.3 ***Constraint satisfaction:*** *The kinematic constraints on the robotic manipulator's joints are satisfied for all times, i.e.,* $\{q(t), \dot{q}(t), \ddot{q}(t), \dddot{q}(t)\} \in \mathcal{Q} \,; \forall t \in [t_0, t_f]$.

III. Methodology

The slosh-free tracking scheme presented in this paper leverages (i) a quadrotor differential flatness based reference generator, (ii) a cascaded proportional derivative task space controller and (iii) a joint space controller based on a convex quadratic programme. These three ingredients are the main building blocks of the proposed methodology. In this section, we present further details in each of them.

A. Quadrotor inspired slosh-free reference generation

As mentioned earlier, in the last few years, a significant amount of the existing slosh-free control literature has focused on formulating novel architectures that leverage the dynamics of the spherical pendulum model. However, as we have already mentioned in Section I, this choice has two main drawbacks: First, the commonly chosen angular parameterization of this model is not applicable to all $\mathrm{SO}(3)$, and second, the pendulum's rod-length introduces a sensitive parameter, leading to a trade-off between tracking quality and slosh-freeness.

In this work, we take a step back and replace the underlying spherical pendulum model with a virtual quadrotor, a system whose configuration inherently fulfills the slosh-free condition presented in Section II. The benefits of this change are fourfold: First, the quadrotor's differential flatness allows us to compute slosh-free orientation references for any 3D trajectory, while also being singularity-free and continuous. Second, the dependency on the pendulum's rod-length is dropped. Third, given that the differential flatness mappings are expressed by closed-form non-linear equations, the calculation of a slosh-free reference does not add any computational burden. Fourth, the capacity to map back and forth from the flat outputs to the quadrotor states allows for computing the reference's longitudinal and angular velocity and acceleration, which might render very appealing for the design of task space controllers.

In the following, we explain how the quadrotor's differential flatness enables the calculation of slosh-free motion references. More specifically, assuming that the end-effector exactly tracks the geometric reference Γ in (4), $e_p(t) = 0$, while mimicking the motions of a (virtual) quadrotor, we proceed to derive the expressions that facilitate the expansion of the position-reference $\boldsymbol{p}_r(t) \in \mathbb{R}^3$ from Γ into a full slosh-free pose-reference $\mathrm{T}_{sf,r} = [\boldsymbol{p}_r(t), \boldsymbol{R}_r(t)] \in \mathrm{SE3}$.

The quadrotor's differential flatness property implies that all its states can be written as algebraic functions of four flat outputs $\boldsymbol{\sigma}(t) = [\boldsymbol{p}_r(t), \psi(t)]$ and their time derivatives $\dot{\boldsymbol{\sigma}}(t)$, $\ddot{\boldsymbol{\sigma}}(t)$, $\dddot{\boldsymbol{\sigma}}(t)$, $\ddddot{\boldsymbol{\sigma}}(t)$. Here $\boldsymbol{p}_r(t)$ is the reference's position, which is obliged to emulate the motion of a (virtual) quadrotor, and $\psi(t)$ is the yaw angle, which is not relevant to the slosh-free problem, and thus, will be fixed to a constant value $\tilde{\psi}$. As a consequence, the flat outputs are fully defined by the geometric reference Γ, i.e., $\boldsymbol{\sigma}(t) = \left[\boldsymbol{p}_r(t), \tilde{\psi}\right] \equiv \Gamma$. This signifies that all the quadrotor or better named the end-effector states can be computed from reference Γ. Out of all these states, the successive task space controller solely requires position and orientation slosh-free states $[\boldsymbol{p}_r(t), \boldsymbol{R}_r(t)]$, and therefore, we will only show the derivations for these two states. For those interested in the computation of the remaining states, please refer to [23].

Commencing with the position reference $\boldsymbol{p}_r(t)$, it becomes evident that it inheretly serves as a flat output that is readily accessible from the reference Γ. Regarding the orientation reference $\boldsymbol{R}_r(t)$, as denoted in the previous paragraph, we parameterize it by a rotation matrix. Its derivation starts by aligning its third component with respect to the resultant acceleration acting on the liquid:

$$\boldsymbol{z}_r(t) = \frac{\boldsymbol{a}_g(t)}{||\boldsymbol{a}_g(t)||} \text{ with } \boldsymbol{a}_g = [\ddot{\sigma}_1(t), \ddot{\sigma}_2(t), \ddot{\sigma}_3(t)] + \boldsymbol{g}. \tag{7a}$$

Next, we define an auxiliary vector

$$\tilde{x}_r = [\cos\sigma_4, \sin\sigma_4, 0]^\mathsf{T} \,, \tag{7b}$$

that allows us to compute the first and second components of the rotation matrix by

$$\boldsymbol{y}_r = \frac{\boldsymbol{z}_r(t) \times \tilde{\boldsymbol{x}}_r}{||\boldsymbol{z}_r(t) \times \tilde{\boldsymbol{x}}_r||}\,, \quad \boldsymbol{x}_r(t) = \boldsymbol{y}_r(t) \times \boldsymbol{z}_r(t)\,.$$

If $\tilde{\boldsymbol{x}}_r(t) \times \boldsymbol{z}_r(t) \neq 0$, which can easily be ensured by choosing $\tilde{\psi}$ appropriately, we can stack all three components to finally obtain the reference's slosh-free rotation matrix:

$$\mathrm{R}_r(t) = [\boldsymbol{x}_r(t), \boldsymbol{y}_r(t), \boldsymbol{z}_r(t)]\,. \tag{7c}$$

From eqs. (7a) and (7b) it is apparent that this matrix exclusively depends on the flat outputs and its derivatives. This allows us to compute the reference's slosh-free orientation for the end-effector by simply evaluating the rotation matrix in (7).

It is worth to highlight that, as per (7a), the computed slosh-free orientation matrix is dependent on the reference's translational accelerations $\boldsymbol{a}_r(t)$. At the same time, the end-effector's orientation is defined by the forward kinematics in (1). Putting both facts together, it results that the reference's acceleration profile $\boldsymbol{a}_r(t)$, if tracked by the associated slosh-free orientation $\mathrm{R}_r(t)$ in (7), is correlated to the robotic manipulator's joint angles $q(t)$. Intuitively, or following the derivations in [23], the joint velocities $\dot{q}(t)$ correlate to the jerk profile $\boldsymbol{j}_r(t)$ of the reference, while the joint accelerations $\ddot{q}(t)$ correlate to the snap profile $\boldsymbol{s}_r(t)$.

B. Task space control: A Cascaded Proportional Derivative

The slosh-free pose-reference $\mathrm{T}_{sf,r} = [\boldsymbol{p}_r(t), \boldsymbol{R}_r(t)]$ derived in the previous subsection, is now used as the input for a task space controller that outputs longitudinal and angular acceleration commands, i.e., $\boldsymbol{u}_{\mathcal{T}}(t) = [\boldsymbol{a}_{\mathcal{T}}(t)\,, \boldsymbol{\omega}_{\mathcal{T}}(t)] \in \mathbb{R}^6$. This controller hinges on an error vector which represents the difference between the desired and current poses:

$$\boldsymbol{e}_{\mathcal{T}}(t) = [\boldsymbol{p}_r(t) - \boldsymbol{p}_e(t),\, e_{\mathrm{R}}(\mathrm{R}_r(t), \mathrm{R}_e(t))] \in \mathbb{R}^6\,, \tag{8}$$

where $e_{\mathrm{R}}(\mathrm{R}_r(t), \mathrm{R}_e(t))$ is the error function between the reference and current orientation. To express this function, in this work we adopt the same approach as in [26]. Alternatively, one could also consider utilizing the more compact variant proposed in [27]. As mentioned before, a cascaded PD controller determines the task space acceleration commands. First, given the pose-error $\boldsymbol{e}_{\mathcal{T}}(t)$ defined in (8), the outer loop computes the desired longitudinal and angular end-effector velocities:

$$\boldsymbol{\nu}_{\mathcal{T}}(t) = [\boldsymbol{v}_{\mathcal{T}}(t), \boldsymbol{\omega}_{\mathcal{T}}(t)] = \boldsymbol{k}_{\mathcal{T}} \odot \boldsymbol{e}_{\mathcal{T}}(t) \in \mathbb{R}^6\,, \tag{9a}$$

where $\odot$ refers to the element-wise product and $\boldsymbol{k}_{\mathcal{T}} \in \mathbb{R}^6$ is a proportional gain that controls the convergence rate in the task space. Subsequently, the computed velocities are fed to the inner loop, which, in conjunction with the feedback acquired from the end-effector's longitudinal and angular velocity defined in $\boldsymbol{\nu}_e(t)$ (2), proceeds to compute the longitudinal and angular acceleration commands:

$$\boldsymbol{u}_{\mathcal{T}}(t) = [\boldsymbol{a}_{\mathcal{T}}(t)\,, \boldsymbol{\omega}_{\mathcal{T}}(t)] = \boldsymbol{k}_{\nu} \odot (\boldsymbol{\nu}_{\mathcal{T}}(t) - \boldsymbol{\nu}_e(t))\,, \tag{9b}$$

where $\boldsymbol{k}_{\nu} \in \mathbb{R}^6$ is also a proportional gain.

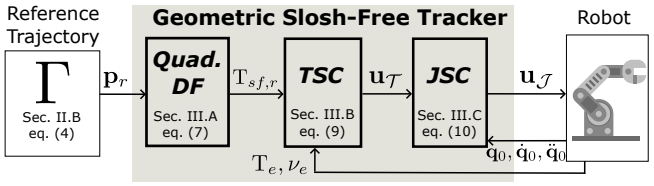

Fig. 2: Block diagram of the presented slosh-free tracking method. "Quad. DF." stands for Quadrotor based differential flatness, "TSC" for task space control and "JSC" for joint space control

C. Joint space control: Optimal Resolved Acceleration

The joint space controller maps the task space acceleration commands $\boldsymbol{u}_{\mathcal{T}}(t)$ to the joint space, $\boldsymbol{u}_{\mathcal{J}}(t) = [\boldsymbol{q}(t), \dot{\boldsymbol{q}}(t), \ddot{\boldsymbol{q}}(t)]$. For this purpose, we rely on a well-established method denoted as Resolved-Acceleration Control (RAC) [25], a direct application of the second order differential equation in (3). Unlike the conventional RAC approach, which computes joint accelerations $\ddot{\boldsymbol{q}}(t)$ via a nonlinear inversion of (3) and leaves the solution's feasibility unattended, we solve a constrained optimal control problem at each iteration, ensuring the kinematic feasibility of the obtained joint configuration.

When formulating the optimal control problem, two design choices are particularly noteworthy due to their pivotal roles in enhancing the methodology's robustness and real-time applicability: First, inspired by [28], we ensure that the optimal control problem always remains feasible by adding slack variables, $\boldsymbol{\delta} \in \mathbb{R}^6$, to the accelerations commanded by the task space controller. These are penalized in the cost function and therefore only get activated if the acceleration commanded by the task space controller is infeasible with respect to the kinematic constraints. Second, to facilitate real-time implementation, we make an approximation by assuming that the Jacobian in (2) and the second-order term of (3) remain constant. It's worth noting that this assumption holds true when the optimization problem is solved at a high rate, which is the underlying rationale for this approximation. Following this simplification, the optimal control problem becomes a convex Quadratic Programme (QP), and thus, can be very efficiently solved [29].

In particular, given an initial joint space configuration $[\boldsymbol{q}_0, \dot{\boldsymbol{q}}_0, \ddot{\boldsymbol{q}}_0]$, acceleration commands from the task space controller $\boldsymbol{u}_{\tau}$ and the robot's kinematic constraints, the QP that we solve at every iteration is:

$$\min_{\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}, \boldsymbol{\delta}} [\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}, \boldsymbol{\delta}]\, P\, [\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}, \boldsymbol{\delta}]^{\mathsf{T}} \tag{10a}$$

$$\text{s.t.} \quad \boldsymbol{q} = \boldsymbol{q}_0 + \dot{\boldsymbol{q}}\Delta t\,, \tag{10b}$$

$$\dot{\boldsymbol{q}} = \dot{\boldsymbol{q}}_0 + \ddot{\boldsymbol{q}}\Delta t\,, \tag{10c}$$

$$\boldsymbol{u}_{\mathcal{T}} + \boldsymbol{\delta} = J_0 \ddot{\boldsymbol{q}} + \dot{\boldsymbol{q}}_0 \otimes H_0\, \dot{\boldsymbol{q}}_0\,, \tag{10d}$$

$$\left[\underline{\boldsymbol{q}}, \underline{\dot{\boldsymbol{q}}}, \underline{\ddot{\boldsymbol{q}}}\right] \leq [\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}] \leq \left[\overline{\boldsymbol{q}}, \overline{\dot{\boldsymbol{q}}}, \overline{\ddot{\boldsymbol{q}}}\right]\,, \tag{10e}$$

$$\underline{\ddot{\boldsymbol{q}}} \leq (\ddot{\boldsymbol{q}} - \ddot{\boldsymbol{q}}_0)/\Delta t \leq \overline{\ddot{\boldsymbol{q}}}\,, \tag{10f}$$

where $P \in \mathbb{R}^{27\times 27}$ is a diagonal positive definite matrix, Δt is the time-step, $J_0 = J(\boldsymbol{q_0})$ and $H_0 = H(\boldsymbol{q_0})$. It's worth noting that the resulting QP is convex, with just 27 variables and all constraints being linear. This illustrates the problem's lightweight nature, and thereby, its real-time applicability.

An overview of the full slosh-free tracking scheme is shown in Fig. 2, where each block within the highlighted

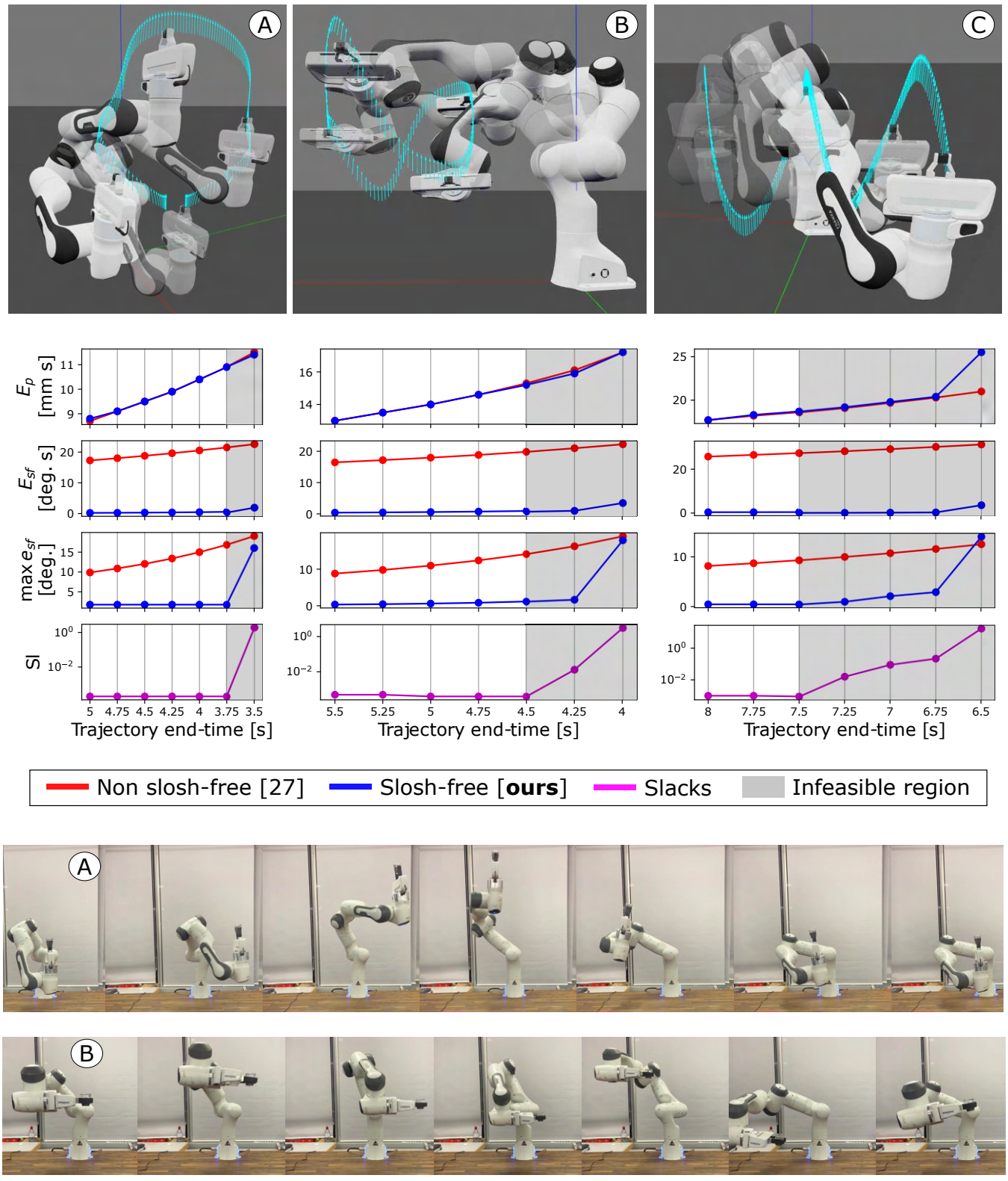

Fig. 3: *Top (above legend box):* A comparison between the presented geometric slosh-free tracker (blue) against a non-slosh-free standard tracker (red) with a Franka Emika Panda robot for three different case-studies: A) Loop (left col.), B) Lissajous (middle col.), C) Helix (right col.). The motions of the robot along these trajectories are shown in the first row. The cyan arrows result from the differential flatness based reference generation and refer to the required acceleration's direction to ensure the motion to be slosh-free. Each case-study has been evaluated for different trajectory execution times. The respective position errors E_p, slosh-free angle errors E_{sf} and maximum slosh-free angle errors $\max e_{sf}$ are depicted in the second, third and fourth rows respectively. The slacks resulting from the slosh-free motions are shown in the fifth row. The gray areas refer to the infeasible regions, i.e, the cases where QP (10) would fail to find a solution if slacks would not be activated. *Bottom (below legend box):* The motions (left to right) resulting from running the proposed slosh-free tracker in a real Franka Emika Panda robot for trajectories (A) and (B) with a cup filled of water attached to its end-effector.

area represents a different component of the proposed slosh-free tracking scheme.

IV. Experiments

To assess the validity of our methodology, we conduct simulated and real-world experiments on a 7 DoF Franka Emika Panda robot arm [30].

A. Simulated analysis

Aiming to understand the performance of our *slosh-free* tracker, we start by comparing it against a *non slosh-free* variant similar to [28], where the quadrotor-based differential flatness part is deactivated, and thus, the resulting end-effector's motions are not slosh-free. The comparison is conducted according to four criterion: First, we evaluate the tracking performance by computing the area below position error $E_p = \int_{t_0}^{t_f} \boldsymbol{e}_p(t)\, dt \in \mathbb{R}$. Second, we analyze the motion's overall slosh-freeness by computing the area below the slosh-free angle error $E_{sf} = \int_{t_0}^{t_f} e_{sf}(t)\, dt \in \mathbb{R}$. Third, we study the extreme slosh-prone cases by also computing the maximum slosh-free angle error $\max \boldsymbol{e}_{sf}(t) \in \mathbb{R};\, \forall t \in [t_0, t_f]$. Fourth, we evaluate the feasibility of the motions by computing the sum of the areas below all slack variables $Sl = \sum_{i=0}^{6} \int_{t_0}^{t_f} \boldsymbol{\delta}_i(t)\, dt \in \mathbb{R}$.

For a better understanding of the solution's performance, we generalize the evaluation by running the comparison in three different case-studies: A) A *loop*-alike trajectory that widely expands the XY plane, B) a *Lissajous* trajectory that requires fast end-effector motions and C) a *helix* that fully exploits the reachable task space. Their differences on shape and scale allow for testing the versatility of the presented method. Moreover, to observe the behavior of the solution in different dynamic regimes, each case-study is evaluated for different navigation times, i.e., starting with low and feasible durations, up to infeasible ones. This allows us to evaluate the performance of the presented slosh-free tracker in slow-moving, as well as in agile and aggressive maneuvers.

To conduct the numerical simulations, we use the Robotics Toolbox for Python [31], which provides a convenient interface to the Franka Emika Panda robot. As a solver for the QP in (10) we use quadprog[1], an open-source software that efficiently solves convex QPs by relying on the Goldfarb/Idnani dual algorithm [32]. Regarding the selection of gains and weights, in the task space control we heuristically choose $\boldsymbol{k}_{\mathcal{T}} = [10, 10, 10, 10, 10, 10]$ and $\boldsymbol{k}_{\nu} = 10\boldsymbol{k}_{\mathcal{T}}$. In the joint space control, we define the weights as $P = \text{blkdiag}(1\text{e}{-8}\, I_7, I_7, 1\text{e}{-8}\, I_7, 1\text{e}3, I_6)$. All of these values are held constant across all the upcoming evaluations.

The robot motions and the respective benchmarks resulting from running the presented slosh-free tracking method are shown in Fig. 3. The first row depicts the evolution of the robot when navigating along the reference trajectories. The cyan arrows refer to the third component of the slosh-free orientation obtained from the quadrotor-based differential flatness. The second, third and fourth rows depict the aforementioned benchmarking metrics E_p, E_{sf}, Sl evaluated for different navigation times. The lower the navigation time, the more aggressive the trajectory. The gray areas within these graphs refer to the infeasible region, where the reference's excessive agility implies that if the slacks would not have been included, the accelerations commanded by the task space controller $\boldsymbol{u}_{\mathcal{T}}(t)$ in (10d) would be infeasible with respect to the kinematic constraints in (10e) and (10f), thereby, causing QP (10) to fail.

The computed motions exhibit three noteworthy characteristics. Firstly, both methods demonstrate equivalent tracking accuracy across all navigation times and case studies. However, the slosh-free variant stands out by delivering a substantial enhancement in the elimination of sloshing. Secondly, in all three cases, the slosh-free variant consistently maintains its slosh-free nature, as indicated by the conditions $e_sf \approx 0$ and $\max e_sf < \epsilon_{sf}$. This holds true unless the reference trajectory becomes excessively infeasible. Thirdly, when the navigation time is exceptionally short, implying an overly agile reference trajectory, adjustments in the form of significantly increased slacks are necessary to render the QP (10) feasible. These adjustments, in turn, result in a substantial modification of the accelerations generated by the task space accelerator, ultimately leading to the generation of sloshing in the motion.

B. Real-world validation

Having understood the behavior of the presented slosh-free tracking method, we proceed in performing real-world experiments. For this purpose, we attach a cup of water to the robotic arm's end-effector and test our solution on two of the three case-studies introduced in the simulations: A) the *loop*-alike trajectory evaluated with a navigation time of 3.75 s and B) the *Lissajous* trajectory with a navigation time of 4.5 s. The remaining *helix* trajectory is not compatible with the laboratory's specific robot setup[2], and thus, will be dropped. The conducted motions are depicted at the lower part of Fig. 3 by image sequences. For a better insight, we encourage readers to view the accompanying video for this paper. As shown by these, and akin to the simulation results, water spillage is successfully avoided.

V. Conclusion

In this work we presented a real-time slosh-free tracker for robotic manipulators. Within the proposed framework, the emulation of the end-effector's motion by a virtual quadrotor has allowed for efficiently calculating a slosh-free reference, which, after going through task and joint space controllers, is converted into a feasible joint space configuration. To validate our findings, first, we have conducted a simulation analysis. The results indicate that our solution approximates slosh-free behaviour without compromising tracking performance, even when dealing with infeasible trajectories. Second, we have run some real-world experiments by attaching a cup of water to a Franka Emika Panda robot and showing that the resulting end-effector motions are spill-free.

[1]quadprog is available in `https://github.com/quadprog/quadprog.git`

[2]The *helix* trajectory requires moving the end-effector below its base's height, which would cause a collision with the large table that holds the robot

REFERENCES

[1] H. Abramson, "Dynamic behavior of liquids in moving containers with applications to propellants in space vehicle fuel tanks," in *Technical Report*, vol. 36, no. 1, 1966, pp. 44–67.

[2] S. a. d. Wiesche, "Computational slosh dynamics: theory and industrial application," *Computational mechanics*, vol. 30, no. 5, pp. 374–387, 2003.

[3] P. G. Lemarié-Rieusset, *The Navier-Stokes problem in the 21st century*. CRC press, 2018.

[4] M. Müller, D. Charypar, and M. Gross, "Particle-based fluid simulation for interactive applications," in *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*. Citeseer, 2003, pp. 154–159.

[5] M. H. Djavareshkian and M. Khalili, "Simulation of sloshing with the volume of fluid method," *Fluid Dynamics & Materials Processing*, vol. 2, no. 4, pp. 299–308, 2006.

[6] Z. Pan and D. Manocha, "Motion planning for fluid manipulation using simplified dynamics," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 4224–4231.

[7] J. Ichnowski, Y. Avigal, Y. Liu, and K. Goldberg, "Gomp-fit: Grasp-optimized motion planning for fast inertial transport," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 5255–5261.

[8] Y. Kuriyama, K. Yano, and M. Hamaguchi, "Trajectory planning for meal assist robot considering spilling avoidance," in *2008 IEEE International Conference on Control Applications*. IEEE, 2008, pp. 1220–1225.

[9] J. Hartz, "Adaptive pouring of liquids with a robotic arm," 2018.

[10] L. Rozo, P. Jiménez, and C. Torras, "Force-based robot learning of pouring skills using parametric hidden markov models," in *9th International Workshop on Robot Motion and Control*, 2013, pp. 227–232.

[11] M. Grundelius and B. Bernhardsson, "Motion control of open containers with slosh constraints," *IFAC Proceedings Volumes*, vol. 32, no. 2, pp. 6142–6147, 1999.

[12] A. Aboel-Hassan, M. Arafa, and A. Nassef, "Design and optimization of input shapers for liquid slosh suppression," *Journal of Sound and Vibration*, vol. 320, no. 1-2, pp. 1–15, 2009.

[13] B. Pridgen, K. Bai, and W. Singhose, "Shaping container motion for multimode and robust slosh suppression," *Journal of Spacecraft and Rockets*, vol. 50, no. 2, pp. 440–448, 2013.

[14] A. Y. S. Wan, Y. D. Soong, E. Foo, W. L. E. Wong, and W. S. M. Lau, "Waiter robots conveying drinks," *Technologies*, vol. 8, no. 3, p. 44, 2020.

[15] K. Yano, S. Higashikawa, and K. Terashima, "Liquid container transfer control on 3d transfer path by hybrid shaped approach," in *Proceedings of the 2001 IEEE International Conference on Control Applications (CCA'01)(Cat. No. 01CH37204)*. IEEE, 2001, pp. 1168–1173.

[16] M. W. Decker, "Active acceleration compensation for transport of delicate objects," Ph.D. dissertation, School of Civil and Environmental Engineering, Georgia Institute of Technology, 2000.

[17] A. X. Dang and I. Ebert-Uphoff, "Active acceleration compensation for transport vehicles carrying delicate objects," *IEEE transactions on robotics*, vol. 20, no. 5, pp. 830–839, 2004.

[18] J. Han, "A study on the coffee spilling phenomena in the low impulse regime," *Achievements in the Life Sciences*, vol. 10, no. 1, pp. 87–101, 2016.

[19] H. Guang, S. Bazzi, D. Sternad, and N. Hogan, "Dynamic primitives in human manipulation of non-rigid objects," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 3783–3789.

[20] L. Biagiotti, D. Chiaravalli, L. Moriello, and C. Melchiorri, "A plug-in feed-forward control for sloshing suppression in robotic teleoperation tasks," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 5855–5860.

[21] R. I. C. Muchacho, R. Laha, L. F. Figueredo, and S. Haddadin, "A solution to slosh-free robot trajectory optimization," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 223–230.

[22] T. Lee, M. Leok, and N. H. McClamroch, "Global formulations of lagrangian and hamiltonian dynamics on manifolds," *Springer*, vol. 13, p. 31, 2017.

[23] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *2011 IEEE international conference on robotics and automation*. IEEE, 2011, pp. 2520–2525.

[24] K. J. Åström and T. Hägglund, *Advanced PID control*. ISA-The Instrumentation, Systems and Automation Society, 2006.

[25] J. Luh, M. Walker, and R. Paul, "Resolved-acceleration control of mechanical manipulators," *IEEE Transactions on Automatic Control*, vol. 25, no. 3, pp. 468–474, 1980.

[26] J. Haviland and P. Corke, "Manipulator differential kinematics: Part i: Kinematics, velocity, and applications," *IEEE Robotics & Automation Magazine*, pp. 2–12, 2023.

[27] T. Lee, M. Leok, and N. H. McClamroch, "Geometric tracking control of a quadrotor uav on se (3)," in *49th IEEE conference on decision and control (CDC)*. IEEE, 2010, pp. 5420–5425.

[28] J. Haviland and P. Corke, "A purely-reactive manipulability-maximising motion controller," *arXiv preprint arXiv:2002.11901*, 2020.

[29] S. P. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[30] S. Haddadin, S. Parusel, L. Johannsmeier, S. Golz, S. Gabl, F. Walch, M. Sabaghian, C. Jähne, L. Hausperger, and S. Haddadin, "The franka emika robot: A reference platform for robotics research and education," *IEEE Robotics & Automation Magazine*, vol. 29, no. 2, pp. 46–64, 2022.

[31] P. Corke and J. Haviland, "Not your grandmother's toolbox–the robotics toolbox reinvented for python," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 11 357–11 363.

[32] D. Goldfarb and A. Idnani, "A numerically stable dual method for solving strictly convex quadratic programs," *Mathematical programming*, vol. 27, no. 1, pp. 1–33, 1983.