Robust Cartesian Kinematics Estimation for Task-Space Control Systems

Seyed Ali Baradaran Birjandi, Niels Dehio, Abderrahmane Kheddar, Sami Haddadin

Robust Cartesian Kinematics Estimation for Task-Space Control Systems

Seyed Ali Baradaran Birjandi[1,*], Niels Dehio[2], Abderrahmane Kheddar[2] and Sami Haddadin[1]

***Abstract*— We discuss a novel method for estimating task Cartesian position and velocity in robot manipulators. This is done by model-based fusion of inertial measurement units with motor encoders. The model is developed to robustly handle the uncertainties in the trajectory. Thus, not only the approach benefits from high fidelity and bandwidth thanks to multiple-sensory fusion, but it also enforces stability despite poorly formulated motions. This empowers the method to be utilized in complex closed-loop applications, where both task position and velocity information is required.**

I. Introduction and State of the Art

Task-space control algorithms are becoming more sophisticated. Being model-based, task-space control formulations require precise information of the robot and its surrounding, at each iteration. Moreover, various sensors, actuators, computational capabilities and software interfaces, shipped with robotics systems, facilitate the execution of complex tasks with minimum error and programming efforts. Besides, sensor measurements are also used in model-based state observers/estimators. The latter, provide information about the plant states, which is being controlled, as well as the environment surrounding the plant. Observers can also provide information about states that cannot be directly measured, e.g. [1]. Tracking performance in closed-loop control systems, including robotics systems, is closely dependant on the (measurable and immeasurable) system states. In task-space control, task kinematics are one of such states, that can be fed back for optimizing the control error.

For many reasons, caused for instance by modeling errors [2], uncontrolled degrees of freedom or geometric disturbances in the robot environment [3], kinematic uncertainties exist in the real-world scenarios. While, nominal kinematic values may not provide accurate information [4]. There exist different approaches to overcome these uncertainties both in open- and closed-loop architectures for various robotics systems. For example, Hyatt *et al.* [5] proposed a method, which parameterize kinematics models, in order to estimate the configuration of large-scale soft robots. Moreover, fusion of multiple sensors such as *inertial measurement unit*s (IMUs) and encoders can estimate the kinematics of a type of legged robots in [6]. Configuration estimation in continuum robots is explained in [7], [8]. Furthermore, kinematics estimation in floating base robots such as humanoids is crucial, as this information is generally the key to stabilize the system [9]. Raghavan *et al.* [10] present a method to robustly estimate a humanoid robot (WALK-MAN) base kinematic by fusion of LIDAR output with IMUs. Furthermore, a stable technique for estimating the orientation of humanoid robots is introduced in [11].

Kinematics estimation is not limited to robotics systems. In fact, estimation of human kinematic is relatively mature and widely used in biomechanics, rehabilitation, ergonomics and sports to commercialized levels [12]. These commercial sensor suits, which mostly consist of either or both IMUs and cameras, can be used to estimate human lower limb [13] and upper limb [14] kinematics.

In brief, kinematics estimation studies mainly focus on position and orientation. To the best of authors' knowledge, higher derivatives of the estimated kinematics are rarely considered in the literature. This is mainly due to the limitations of measurements such as, quantization errors, noise and sensors (mostly cameras) low sampling rate. Therefore, extra care should be taken when using these estimations in task-space control or more generally, in closed-loop systems.

In our previous works [15], [16] we proposed a method which made possible the estimations of higher derivatives of manipulators joint variables. We extend this work to estimate the task kinematics, including position and velocity, in this paper. For this, IMUs are fused with motor encoder measurements. Moreover, potential numerical instabilities in the previous works are resolved, using the information of the robot desired trajectory. Thus, the estimations can be safely used in any naturally-stable closed-loop scheme.

The remainder of the paper is organized as follows. In Section II, the problem of interest and the proposed solution are briefly explained. In Section III, we review a potential risk that state-of-the-art kinematics estimators may face. Subsequently, we propose a solution to resolve this issue. In Section IV and V, we validate the theory in simulation and experiment with a 7-DoF robot, respectively. Finally, the paper concludes in Section VI.

[1] Munich Institute of Robotics and Machine Intelligence, Technical University of Munich, Germany.

[2] CNRS-University of Montpellier, IDH, LIRMM, France.

[*] Corresponding author. E-mail address: ali.baradaran@tum.de

We greatly acknowledge the funding of this work by of the Lighthouse Initiative Geriatronics by StMWi Bayern (Project X, grant no. IUK-1807-0007// IUK582/001) and LongLeif GaPa gGmbH (Project Y). This work was also supported by the European Union's Horizon 2020 research and innovation programme as part of the project I.AM. under grant no. 871899 and the project Darko under grant no. 101017274. We also acknowledge the financial support by the Bavarian State Ministry for Economic Affairs, Regional Development and Energy (StMWi) for the Lighthouse Initiative KI.FABRIK, (Phase 1: Infrastructure as well as the research and development program under, grant no. DIK0249). The authors also wish to thank Mr. Erfan Shahriari for his generous support. Please also note that S. Haddadin has a potential conflict of interest as shareholder of Franka Emika GmbH.

II. Problem Statement and Contribution

Let us assume the reduced rigid body dynamics model of a serial-chain robot with n joints as [17]

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{\tau}_{\mathrm{m}} - \boldsymbol{\tau}_{\mathrm{f}}, \tag{1}$$

where $\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}} \in \mathbb{R}^n$ denote the link side joint position, velocity and acceleration, $\boldsymbol{M}(\boldsymbol{q}) \in \mathbb{R}^{n\times n}$ the symmetric and positive definite inertia matrix, $\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} \in \mathbb{R}^n$ the centripetal and Coriolis vector, $\boldsymbol{g}(\boldsymbol{q}) \in \mathbb{R}^n$ the gravity vector, $\boldsymbol{\tau}_{\mathrm{m}} \in \mathbb{R}^n$ the active motor torque and $\boldsymbol{\tau}_{\mathrm{f}} \in \mathbb{R}^n$ the torque caused by external effects (e.g., frictions).
Now, take an arbitrary position $\boldsymbol{x} \in \mathbb{R}^6$ defined in the robot base frame. This position can be defined anywhere on the robot structure or an attached object to its end-effector ($\boldsymbol{x}_{\mathrm{ee}} \in \mathbb{R}^6$). Here, we assume that this position is rigidly connected to the robot end-effector. Therefore, the Cartesian velocity $\dot{\boldsymbol{x}} \in \mathbb{R}^6$ of this position is given by

$$\dot{\boldsymbol{x}} = \boldsymbol{J}_{\mathrm{p}}\dot{\boldsymbol{x}}_{\mathrm{ee}}, \tag{2}$$

where $\dot{\boldsymbol{x}}_{\mathrm{ee}} \in \mathbb{R}^6$ is the end-effector Cartesian velocity and the Jacobian matrix $\boldsymbol{J}_{\mathrm{p}} \in \mathbb{R}^{6\times 6}$ is

$$\boldsymbol{J}_{\mathrm{p}} = \begin{bmatrix} \boldsymbol{I}_3 & [{}^{\mathrm{ee}}\boldsymbol{p}]_\times^T \\ \boldsymbol{0}_{3\times 3} & \boldsymbol{I}_3 \end{bmatrix}. \tag{3}$$

Here $\boldsymbol{I}_3 \in \mathbb{R}^{3\times 3}$ is the identity matrix. Also, ${}^{\mathrm{ee}}\boldsymbol{p} \in \mathbb{R}^3$ is the location of the position $\boldsymbol{x}$ in the end-effector frame $\{\mathrm{ee}\}$ and $[\cdot]_\times$ is the skew symmetric matrix representation of a vector. Moreover, the manipulator Jacobian matrix $\boldsymbol{J}_{\mathrm{ee}}(q) \in \mathbb{R}^{6\times n}$ is given by

$$\boldsymbol{J}_{\mathrm{ee}}(q) = \frac{\partial \boldsymbol{x}_{\mathrm{ee}}(\boldsymbol{q})}{\partial \boldsymbol{q}}. \tag{4}$$

Thus, we can map the joint velocity $\dot{\boldsymbol{q}}$ to Cartesian velocity $\dot{\boldsymbol{x}}$ of a given position $\boldsymbol{x}$ via

$$\dot{\boldsymbol{x}} = \boldsymbol{J}_{\mathrm{p}}\boldsymbol{J}_{\mathrm{ee}}(\boldsymbol{q})\dot{\boldsymbol{q}} = \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}}, \tag{5}$$

We can therefore, rewrite the dynamics model (1) in Cartesian space at the desired position $\boldsymbol{x}$ [18]

$$\boldsymbol{\Lambda}(\boldsymbol{q})\ddot{\boldsymbol{x}} + \boldsymbol{\mu}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{x}} + \boldsymbol{J}(\boldsymbol{q})^{-T}\boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{J}(\boldsymbol{q})^{-T}\boldsymbol{\tau}_{\mathrm{m}} - \boldsymbol{J}(\boldsymbol{q})^{-T}\boldsymbol{\tau}_{\mathrm{f}}, \tag{6}$$

with

$$\boldsymbol{\Lambda}(\boldsymbol{q}) = \boldsymbol{J}(\boldsymbol{q})^{-T}\boldsymbol{M}(\boldsymbol{q})\boldsymbol{J}(\boldsymbol{q})^{-1} \tag{7}$$

$$\boldsymbol{\mu}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \boldsymbol{J}(\boldsymbol{q})^{-T}\left(\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}}) - \boldsymbol{M}(\boldsymbol{q})\boldsymbol{J}(\boldsymbol{q})^{-1}\dot{\boldsymbol{J}}(\boldsymbol{q})\right)\boldsymbol{J}(\boldsymbol{q})^{-1}. \tag{8}$$

In general, in model-based Cartesian control algorithms, either of dynamics models (1) or (6) may be utilized. Subsequently, forward and inverse Kinematics are used to project the robot/object dynamics and the acting forces into Cartesian or joint space, interchangeably. Each approach benefits/suffers from some advantages/disadvantages. However, irrespective of the model of choice, the desired trajectory or objective is expressed primarily in Cartesian space in task-space control methods. Therefore task and robot Kinematics should be available at any given time with high numerical stability and precision. This in practice requires fusion of multiple measurements together with careful design.

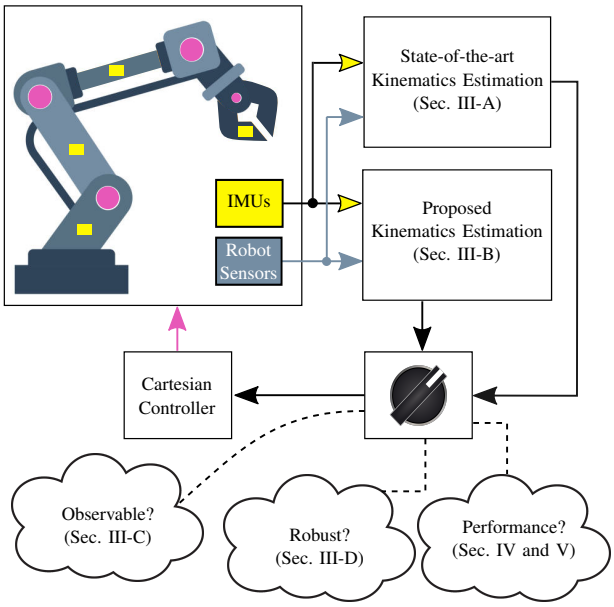

Fig. 1: Summary of the paper research questions. We initially investigate whether or not state-of-the-art methods for estimating kinematics are robust enough to be used in closed-loop systems. Subsequently, we introduce a new method for estimating robot/object Kinematics and finally examine the performance of this method.

Therefore, the problem that this paper strives the resolve is to estimate Cartesian velocity of a given position, which is rigidly attached to the robot manipulator, with minimal instability and high precision. This is done by fusing the robot proprioceptive measurements with easy-to-use and cheap inertial measurement units data. IMUs are installed on robot structure and report the Cartesian acceleration and angular velocity of their frame. The contributions of this work are listed in the followings.

- First, we briefly review some state-of-the-art joint variables robot estimators, and point out its potential drawback in closed-loop schemes.
- Subsequently, we propose an approach for robustly estimating joint variables. We also investigate the properties of the system, i.e. its steady-state behaviour and its observability. These properties are specifically important when the estimated variables are supposed to be used in closed-loop control. Moreover, forward kinematics approach for estimating Cartesian space variables based on the estimated joint variables, is briefly explained.
- Finally, we evaluate our findings in closed-loop scheme in simulations and in open-loop scheme in experiments with a robot manipulator.

Figure 1 summarizes the research questions that this paper tries to answer and the corresponding contributions.

III. Cartesian Velocity Estimation

A. State-of-the-Art Method for Estimating Joint Velocity and Acceleration

In order to be able to obtain the joint i variables by fusing IMU m with the robot link-side measurements, an estimator

is utilized with the dynamics model [16]

$$\dot{\hat{\mathbf{x}}}_{\mathrm{o1,i}} = \boldsymbol{f}_{\mathrm{o1,i}}(\hat{\mathbf{x}}_{\mathrm{o1,i}}) = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \hat{q}_{\mathrm{i}} \\ \hat{\dot{q}}_{\mathrm{i}} \\ \hat{\ddot{q}}_{\mathrm{i}} \\ \hat{\dddot{q}}_{\mathrm{i}} \end{pmatrix} + \boldsymbol{w}_{\mathrm{o1,i}},$$

$$\hat{\mathbf{y}}_{\mathrm{o1,i}} = \boldsymbol{h}_{\mathrm{o1,i}}(\hat{\mathbf{x}}_{\mathrm{o1,i}}(1,2,3)) = \begin{pmatrix} \hat{q}_{\mathrm{i}} \\ {}^{\mathrm{m}}\boldsymbol{a}(\hat{q}_{\mathrm{i}}, \hat{\dot{q}}_{\mathrm{i}}, \hat{\ddot{q}}_{\mathrm{i}}) \\ {}^{\mathrm{m}}\boldsymbol{\omega}(\hat{q}_{\mathrm{i}}, \hat{\dot{q}}_{\mathrm{i}}) \end{pmatrix} + \boldsymbol{v}_{\mathrm{o1}}, \quad (9)$$

with $\mathbf{x}_{\mathrm{o1,i}} = [q_{\mathrm{i}}\ \dot{q}_{\mathrm{i}}\ \ddot{q}_{\mathrm{i}}\ \dddot{q}_{\mathrm{i}}]^T \in \mathbb{R}^4$ and $\hat{\mathbf{x}}_{\mathrm{o1,i}} = [\hat{q}_{\mathrm{i}}\ \hat{\dot{q}}_{\mathrm{i}}\ \hat{\ddot{q}}_{\mathrm{i}}\ \hat{\dddot{q}}_{\mathrm{i}}]^T \in \mathbb{R}^4$ being the system states and the estimated system states, respectively. Moreover, $\boldsymbol{f}_{\mathrm{o1,i}} : \mathbb{R}^4 \to \mathbb{R}^4$ is the system transition function and $\boldsymbol{h}_{\mathrm{o1,i}} : \mathbb{R}^3 \to \mathbb{R}^7$ is the measurement fucntion. $\boldsymbol{w}_{\mathrm{o1,i}} \in \mathbb{R}^3$ and $\boldsymbol{v}_{\mathrm{o1,i}} \in \mathbb{R}^7$ are process and measurement noise, respectively. Furthermore, IMU m is installed on link i and outputs Cartesian acceleration ${}^{\mathrm{m}}\boldsymbol{a}(q_{\mathrm{i}}, \dot{q}_{\mathrm{i}}, \ddot{q}_{\mathrm{i}}) \in \mathbb{R}^3$ and angular velocity ${}^{\mathrm{m}}\boldsymbol{\omega}(q_{\mathrm{i}}, \dot{q}_{\mathrm{i}}) \in \mathbb{R}^3$ in frame $\{\mathrm{m}\}$. According to the kinematical equations for rigid bodies [19]

$${}^{\mathrm{i}}\boldsymbol{a}_{\mathrm{m}} = {}^{\mathrm{i}}\boldsymbol{a}_l + {}^{i}\dot{\boldsymbol{\omega}}_{\mathrm{i}} \times {}^{\mathrm{i}}\boldsymbol{x}_{S_{\mathrm{m}}} + {}^{\mathrm{i}}\boldsymbol{\omega}_{\mathrm{i}} \times \left({}^{\mathrm{i}}\boldsymbol{\omega}_{\mathrm{i}} \times {}^{\mathrm{i}}\boldsymbol{x}_{S_{\mathrm{m}}}\right), \quad (10)$$

$${}^{\mathrm{m}}\boldsymbol{a} = {}^{\mathrm{m}}\boldsymbol{R}_{\mathrm{i}}\, {}^{\mathrm{i}}\boldsymbol{a}_{\mathrm{m}}, \quad (11)$$

$$\begin{pmatrix} 0 \\ 0 \\ \dot{q}_i \end{pmatrix} = {}^{\mathrm{i}}\boldsymbol{\omega}_i - {}^{i}\boldsymbol{\omega}_{i-1}, \ \begin{pmatrix} 0 \\ 0 \\ \ddot{q}_{\mathrm{i}} \end{pmatrix} = {}^{\mathrm{i}}\dot{\boldsymbol{\omega}}_{\mathrm{i}} - {}^{\mathrm{i}}\dot{\boldsymbol{\omega}}_{\mathrm{i-1}}, \quad (12)$$

$${}^{\mathrm{m}}\boldsymbol{\omega} = {}^{\mathrm{i}}\boldsymbol{\omega}_{\mathrm{i}} \quad (13)$$

hold. ${}^{\mathrm{i}}\boldsymbol{a}_{\mathrm{m}} \in \mathbb{R}^3$ and ${}^{\mathrm{i}}\dot{\boldsymbol{\omega}}_{\mathrm{i}} \in \mathbb{R}^3$ are the Cartesian acceleration and angular velocity at sensor location ${}^{\mathrm{i}}\boldsymbol{x}_{S_{\mathrm{m}}} \in \mathbb{R}^3$ in frame $\{\mathrm{i}\}$, respectively. Moreover, ${}^{\mathrm{m}}\boldsymbol{R}_{\mathrm{i}} \in \mathbb{R}^{3\times3}$ is the rotation matrix from frame $\{\mathrm{i}\}$ to frame $\{\mathrm{m}\}$. Since angular velocity of a point in rigid bodies is independent of the location of measurements, (13) holds. According to (12), for an n-link robot manipulator, the joint variables are estimated recursively from the first link $\mathrm{i} = 1$ to the last link $\mathrm{i} = n$.

Constant jerk in (9) is in general not accurate. However, in the absence of the desired trajectory information, constant jerk assumption will impose less numerical instability compared to constant acceleration or velocity assumptions. Moreover, we have shown that observer (9) is computationally light. Also, the estimated variables have higher bandwidth and accuracy compared to the state-of-the-art methods [15], [16]. Furthermore, the method is modular, meaning that it can be inserted to a given setup with minimum connection, tuning effort and computation costs as it does not depend on the robot dynamics, controller or the desired trajectory. Therefore, observer (9) can be used in applications such as collision detection, where quick responses are desirable [16], [20].

Nevertheless, observer (9) has been used in open-loop applications only. Besides the advantages of minimum-connection requirement, it may cause some difficulty especially in closed-loop schemes. Let us exemplify this by deriving the observer (9) error dynamics as

$$\begin{aligned} \dot{\boldsymbol{e}}_{\mathrm{o1,i}} &= \dot{\mathbf{x}}_{\mathrm{o1,i}} - \boldsymbol{f}_{\mathrm{o1,i}}(\hat{\mathbf{x}}_{\mathrm{o1,i}}) \\ &\quad + \boldsymbol{K}_{\mathrm{o1,i}}(t)\left(\boldsymbol{y} - \boldsymbol{h}_{\mathrm{o1,i}}(\hat{\mathbf{x}}_{\mathrm{o1,i}}(1,2,3))\right), \end{aligned} \quad (14)$$

$$\text{where } \boldsymbol{e}_{\mathrm{o1,i}} = \mathbf{x}_{\mathrm{o1,i}} - \hat{\mathbf{x}}_{\mathrm{o1,i}} \in \mathbb{R}^4, \quad (15)$$

with $\boldsymbol{K}_{\mathrm{o1,i}}(t) \in \mathbb{R}^{4\times7}$ being the observer gain and $\boldsymbol{y} \in \mathbb{R}^7$ the measurements. Consider that the robot is at rest and the observer (9) at the steady state, meaning that

$$\begin{aligned} \dot{\mathbf{x}}_{\mathrm{o1,i}} &= \mathbf{0}, \\ \dot{\hat{\mathbf{x}}}_{\mathrm{o1,i}} &\simeq \mathbf{0}, \\ \dot{\overline{\boldsymbol{K}}}_{\mathrm{o1,i}} &\simeq \mathbf{0}, \\ \dot{\overline{\boldsymbol{e}}}_{\mathrm{o1,i}} &\simeq \mathbf{0}. \end{aligned}$$

Here, $\overline{\boldsymbol{K}}_{\mathrm{o1,i}}$ is the observer gain at the steady state. Moreover, the observer steady-state error $\overline{\boldsymbol{e}}_{\mathrm{o1,i}}$ and the changes in the estimated state are at minimum. Therefore, we can linearize (14) around the equilibrium point $\overline{\boldsymbol{e}}_{\mathrm{o1,i}}$, as

$$\dot{\boldsymbol{e}}_{\mathrm{o1,i}} = \boldsymbol{A}_{\mathrm{o1,i}}\boldsymbol{e}_{\mathrm{o1,i}} + \boldsymbol{\xi}_{\mathrm{o1,i}}, \quad (16)$$

$$\begin{aligned} \text{where } \boldsymbol{A}_{\mathrm{o1,i}} &= \frac{\partial}{\partial \boldsymbol{e}_{\mathrm{o1,i}}} \boldsymbol{f}_{\mathrm{o1,i}}(\boldsymbol{e}_{\mathrm{o1,i}})|_{\overline{\boldsymbol{e}}_{\mathrm{o1,i}}} \\ &\quad + \overline{\boldsymbol{K}}_{\mathrm{o1,i}} \frac{\partial}{\partial \boldsymbol{e}_{\mathrm{o1,i}}} \boldsymbol{h}_{\mathrm{o1,i}}(\boldsymbol{e}_{\mathrm{o1,i}}(1,2,3))|_{\overline{\boldsymbol{e}}_{\mathrm{o1,i}}} \end{aligned} \quad (17)$$

$$\text{and } \boldsymbol{\xi}_{\mathrm{o1,i}} = \boldsymbol{e}_{\mathrm{o1,i}}(t_0) \in \mathbb{R}^4. \quad (18)$$

We assume that the observer is stable. In other words the steady state gain $\overline{\boldsymbol{K}}_{\mathrm{o1,i}}$ is chosen such that the eigenvalues of $\boldsymbol{A}_{\mathrm{o1,i}}$ have negative real values. According to linearized model (16), the error dynamics are a set of ordinary differential equations with a stable equilibrium at the origin. Actuating the robot at t_0 in this case, introduces large errors $\boldsymbol{\xi}_{\mathrm{o1,i}}$ to the observer, as it defies the observer assumption of constant jerk ($\dddot{q} = 0$ in (9)):

$$\begin{aligned} \boldsymbol{\xi}_{\mathrm{o1,i}}(3) &= \cancelto{\neq 0}{\mathbf{x}_{\mathrm{o1,i}}(3)} - \cancelto{\approx 0}{\hat{\mathbf{x}}_{\mathrm{o1,i}}(3)} &&\neq 0, \\ \boldsymbol{\xi}_{\mathrm{o1,i}}(4) &= \cancelto{\infty}{\mathbf{x}_{\mathrm{o1,i}}(4)} - \cancelto{\approx 0}{\hat{\mathbf{x}}_{\mathrm{o1,i}}(4)} &&\neq 0. \end{aligned}$$

This error, which we call *observer initial error*, appears every time there is a discontinuity in the robot trajectory, while the observer is at the steady state ($\boldsymbol{e}_{\mathrm{o1,i}} = \overline{\boldsymbol{e}}_{\mathrm{o1,i}} \simeq \mathbf{0}$). Note that the initial error will eventually decay to negligible values (the equilibrium point at $\overline{\boldsymbol{e}}_{\mathrm{o1,i}}$) as system (16) or more generally, system (14) are assumed to be stable. However, depending on the observer convergence rate, $\boldsymbol{\xi}_{\mathrm{o1,i}}$ affects the closed-loop system differently. If for example, the observer error dynamics are considerably slower than the closed-loop system ones, the closed-loop system may become unstable. As (14) suggests, the observer error dynamics depend on several factors such as the type of the estimator, estimator tuning, gain update law, etc. One can derive the closed loop system dynamics and subsequently tune the observer such that its initial error remains within the controller bounds. This may be difficult based on the controller. However, we propose a new method for estimating the joint variables with minimum initial error. This can be done by improving the observer knowledge of the desired trajectory.

B. Proposed Method for Estimating Joint Velocity and Acceleration

We propose the new observer dynamics as

$$\hat{\dot{\mathbf{x}}}_{\mathrm{o2,i}} = \boldsymbol{f}_{\mathrm{o2,i}}(\hat{\mathbf{x}}_{\mathrm{o2,i}}, u_{\mathrm{o2,i}}) = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \hat{q}_{\mathrm{i}} \\ \hat{\dot{q}}_{\mathrm{i}} \\ \hat{\ddot{q}}_{\mathrm{i}} \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} u_{\mathrm{o2,i}} + \boldsymbol{w}_{\mathrm{o2,i}},$$

$$\hat{\mathbf{y}}_{\mathrm{o2,i}} = \boldsymbol{h}_{\mathrm{o1,i}}(\hat{\mathbf{x}}_{\mathrm{o2,i}}), \tag{19}$$

$$\text{with } u_{\mathrm{o2,i}} = \frac{\mathrm{d}}{\mathrm{d}t}\ddot{q}_{\mathrm{d,i}} \tag{20}$$

with $\mathbf{x}_{\mathrm{o2,i}} = [q_{\mathrm{i}}\ \dot{q}_{\mathrm{i}}\ \ddot{q}_{\mathrm{i}}]^T \in \mathbb{R}^3$ and $\hat{\mathbf{x}}_{\mathrm{o2,i}} = [\hat{q}_{\mathrm{i}}\ \hat{\dot{q}}_{\mathrm{i}}\ \hat{\ddot{q}}_{\mathrm{i}}]^T \in \mathbb{R}^3$ being the new states and their corresponding estimation, respectively. $\ddot{q}_{\mathrm{d,i}} \in \mathbb{R}$ is the reference acceleration trajectory for i-th link (output of the task-space controller) and its derivative is the input to the new dynamics model (19). Moreover, $\boldsymbol{w}_{\mathrm{o2,i}} \in \mathbb{R}^3$ is the process noise for the new observer dynamics model. Also, $\boldsymbol{f}_{\mathrm{o2,i}} : \mathbb{R}^3 \to \mathbb{R}^3$ is the system transition function. Note that dynamics models (9) and (19) share the same measurement function $\boldsymbol{h}_{\mathrm{o1.i}}(\cdot)$. Since the observer is proposed to be utilized in closed-loop schemes, its properties should be examined carefully. Therefore, we inspect the observability of (19) next.

C. Observability

Observability of a system indicates whether or not we can estimate the initial state (or in general, all states from $t = 0$) of the system, given the available measurements. In other words, if we prove that system (19) is observable, we can assume that the observer can converge to the steady state at $\overline{\boldsymbol{e}}_{\mathrm{o2,i}} = \mathbf{0}$.

Specifically, local weak observability is a notion that is commonly used for observability investigation of nonlinear systems. Roughly speaking, it states that the system is locally weakly observable as long as every pair of states in a closed neighborhood around the initial state are distinguishable [21].

a) Definition 1: The observation space for an input nonlinear system of the form

$$\begin{aligned} \dot{\mathbf{x}} &= \boldsymbol{f}\left(\mathbf{x}(t), \mathbf{u}(t)\right) \\ \mathbf{y} &= \boldsymbol{h}\left(\mathbf{x}(t)\right) \end{aligned} \tag{21}$$

with $\mathbf{x} \in \mathbb{R}^p$ being the system states, is defined as the smallest real vector space of $\mathcal{C}^\infty$ functions containing the components of the measurement function $\boldsymbol{h}(\cdot) : \mathbb{R}^p \to \mathbb{R}^q$, while it is closed under the Lie derivation w.r.t the vector field $\boldsymbol{f}(\cdot) : \mathbb{R}^p \times \mathbb{R}^m \to \mathbb{R}^p$ for any constant input $\mathbf{u} \in \mathbb{R}^m$ [22]. The Lie derivative of a $\mathcal{C}^\infty$ function $\boldsymbol{h}(\mathbf{x})$ w.r.t $\boldsymbol{f}(\mathbf{x}, \mathbf{u})$ is given by [23]

$$\mathcal{L}_f(\boldsymbol{h}(\mathbf{x})) = \frac{\partial \boldsymbol{h}(\mathbf{x})}{\partial \mathbf{x}} \boldsymbol{f}(\mathbf{x}, \mathbf{u}) \tag{22}$$

$$\mathcal{L}_f^{\mathrm{m}}(\boldsymbol{h}(\mathbf{x})) = \mathcal{L}_f\left(\mathcal{L}_f^{\mathrm{m}-1}(\boldsymbol{h}(\mathbf{x}))\right),\ \mathrm{m} \geq 1 \tag{23}$$

$$\text{with } \mathcal{L}_f^0(\boldsymbol{h}(\mathbf{x})) = \boldsymbol{h}(\mathbf{x}). \tag{24}$$

Moreover, the observability matrix for nonlinear systems is given by [24]

$$\mathcal{O} = \begin{pmatrix} \frac{\partial}{\partial \mathbf{x}} \mathcal{L}_f^0(\boldsymbol{h}(\mathbf{x})) \\ \frac{\partial}{\partial \mathbf{x}} \mathcal{L}_f^1(\boldsymbol{h}(\mathbf{x})) \\ \vdots \\ \frac{\partial}{\partial \mathbf{x}} \mathcal{L}_f^{\mathrm{p}-1}(\boldsymbol{h}(\mathbf{x})) \end{pmatrix} \tag{25}$$

b) Definition 2: A system is locally weakly observable, when the derivative of its observation vector space w.r.t to the states is of full rank. Equivalently, when the observability matrix $\mathcal{O}$ has full rank (p), the system (21) is locally weakly observable [25].

With Def. 1 and 2 we can investigate the observability of the system (19). The observability matrix can be obtained based on (10)-(13). This matrix has full rank ($= 3$) and therefore, system (19) is locally weakly observable. Note that due to large size of the observability matrix, we omit the results from this section.

D. Robustness

In the previous section, we showed that the observer error reaches the steady state. Now, if we prove that disturbances, i.e. discontinuities in the desired trajectory[1], can not deflect observer (19) from the steady state, we conclude that the observer is robust. Similar to Sec. III-A, we derive the linearized error dynamics for observer (19) when the observer is at the steady state, as

$$\dot{\boldsymbol{e}}_{\mathrm{o2,i}} = \boldsymbol{A}_{\mathrm{o2,i}} \boldsymbol{e}_{\mathrm{o2,i}} + \boldsymbol{\xi}_{\mathrm{o2,i}}, \tag{26}$$

$$\text{where } \boldsymbol{e}_{\mathrm{o2,i}} = \mathbf{x}_{\mathrm{o2,i}} - \hat{\mathbf{x}}_{\mathrm{o2,i}} \in \mathbb{R}^3, \tag{27}$$

$$\text{and } \boldsymbol{\xi}_{\mathrm{o2,i}} = \boldsymbol{e}_{\mathrm{o2,i}}(t_0) \in \mathbb{R}^3. \tag{28}$$

In case of small tracking error, when there is a discontinuity in the desired trajectory at t_0, the initial error remains negligible, as

$$\begin{aligned} \boldsymbol{\xi}_{\mathrm{o2,i}}(1) &= \cancel{\mathbf{x}_{\mathrm{o2,i}}(1)} - \cancel{\hat{\mathbf{x}}_{\mathrm{o2,i}}(1)} \overset{q_{i,0}}{\approx} q_{i,0} &&\approx 0, \\ \boldsymbol{\xi}_{\mathrm{o2,i}}(2) &= \cancel{\mathbf{x}_{\mathrm{o2,i}}(2)} - \cancel{\hat{\mathbf{x}}_{\mathrm{o2,i}}(2)} \overset{\dot{q}_{i,0}}{\approx} \dot{q}_{i,0} &&\approx 0, \\ \boldsymbol{\xi}_{\mathrm{o2,i}}(3) &= \cancel{\mathbf{x}_{\mathrm{o2,i}}(3)} - \cancel{\hat{\mathbf{x}}_{\mathrm{o2,i}}(3)} \overset{\approx \ddot{q}_{\mathrm{d,i}}}{\approx} \ddot{q}_{\mathrm{d,i}} &&\approx 0, \end{aligned}$$

hold. $q_{i,0}$ and $\dot{q}_{i,0}$ are the robot initial configuration and joint velocity, respectively. As the observer is assumed to be at the steady state, these variables are known. Thus, at t_0 error $\boldsymbol{e}_{\mathrm{o2,i}}$ remain at the origin.

Since system (19) is both observable and robust, we propose to compute the Cartesian velocity $\hat{\dot{\boldsymbol{x}}}$ via the corresponding estimated joint variables, as

$$\hat{\dot{\boldsymbol{x}}} = \boldsymbol{J}(\hat{\boldsymbol{q}})\hat{\dot{\boldsymbol{q}}}. \tag{29}$$

[1]Note that disturbances which correspond to a failure in the system, such as collisions, sensor or actuator failures, should destabilize the observer. As a result, fault detection units can detect these failures. This paper does not account for these types of disturbances, which potentially destabilize system (19).

Here, $\hat{\boldsymbol{q}} \in \mathbb{R}^n$ and $\hat{\dot{\boldsymbol{q}}} \in \mathbb{R}^n$ are the recursively estimated joint angle and velocity using (19) for all links $\mathrm{i} = 1$ to $\mathrm{i} = n$. If the desired trajectory is expressed in the task space, the input to the observer (19) is given by

$$\boldsymbol{u}_{\mathrm{o}2} = \frac{\mathrm{d}}{\mathrm{d}t}\left(\boldsymbol{J}^{-1}(\hat{\boldsymbol{q}})\left(\dot{\boldsymbol{J}}(\hat{\boldsymbol{q}},\hat{\dot{\boldsymbol{q}}})\hat{\dot{\boldsymbol{q}}} - \ddot{\boldsymbol{x}}_{\mathrm{d}}\right)\right), \tag{30}$$

$$\text{with } \boldsymbol{u}_{\mathrm{o}2} = \left[u_{\mathrm{o}2,1}, \cdots, u_{\mathrm{o}2,\mathrm{i}}, \cdots, u_{\mathrm{o}2,\mathrm{n}}\right]^T, \tag{31}$$

where $\ddot{\boldsymbol{x}}_{\mathrm{d}} \in \mathbb{R}^6$ is the desired acceleration trajectory in task space. It can be assumed that the trajectory planner respects all joint and torque limits of the robot manipulator. Therefore, the pseudo inverse of Jacobian matrix $\boldsymbol{J}^{-1}(\hat{\boldsymbol{q}})$, which is normally non-square, is always bounded.

IV. Simulation Experiments

Initially, the accuracy of Cartesian velocity and acceleration, estimated with the proposed method is examined in this section. Subsequently, the estimated variables are used in *quadratic programming* (QP) control scheme. The simulation results are based on a 7-DoF robot manipulator[2]. In order to give the results scientific validity, different real world effects are implemented in the simulations. Before reporting the results, these effects, listed in table I, are detailed in the followings. The linear viscous friction model is

TABLE I: List of parasitic effects and model errors present in simulations

Real world effects
Motor-side viscous friction
Motor encoder quantization
Motor encoder noise
Motor encoder LPF
Torque measurement noise
IMU quantization IMU noise
IMU bias
IMU LPF
IMU sensitivity and bias change due to temperature

$$\boldsymbol{\tau}_{\mathrm{f}} = \boldsymbol{F}_{\theta}\dot{\boldsymbol{\theta}}, \tag{32}$$

with $\boldsymbol{F}_{\theta}$ being the diagonal motor side viscous coefficients matrix. The simulated encoder is a 12-bit digital sensor whose 3 *least significant bit*s (LSBs) are affected by noise. The quantization effects as well as an LPF are also modeled for the simulated encoder. The LPF has a cutoff frequency of 300 Hz and truncates the sensor noise effects. The IMUs are modeled based on Bosch BMI055 [27] with all parasitic effects such as noise, bias, quantization and sensitivity and bias change due to temperature. The temperature profile swings between 15° to 40° in a sinusoidal wave and affects the sensor drift and sensitivity in the simulation, based on the statistical relationship provided in the datasheet. Moreover, according to the datasheet, noise power spectral density is 150 $\mu\mathrm{g}/\sqrt{\mathrm{Hz}}$ for accelerometers and $0.014^{\circ}/\mathrm{s}/\sqrt{\mathrm{Hz}}$ for gyroscopes. The full scale is chosen 1 g for the accelerometer and 1000 $^{\circ}/\mathrm{s}$ for the gyroscope. Both sensors report the measurements in 12 bits. Each link is equipped with one IMU. The location of each IMU is ${}^{\mathrm{i}}\boldsymbol{x}_{S_{\mathrm{m}}} = [0\ 0.05\ 0]^T\mathrm{m}$ w.r.t their corresponding joint frame $\{\mathrm{i}\}$.

[2]The dynamics model of the simulated 7-DoF manipulator is taken from [26] and accessible from http://diag.uniroma1.it/∼gaz/panda2019.html.

In both sections the robot (all joints) is actuated to follow a 20-seconds excitation trajectory. This periodic Fourier-like trajectory [28]

$$q_{\mathrm{i}}(t) = q_{\mathrm{i},0} + \sum_{\mathrm{r}=1}^{R}\frac{a_{\mathrm{r,i}}}{r\omega}\sin(\mathrm{r}\omega t) - \sum_{\mathrm{r}=1}^{R}\frac{b_{\mathrm{r,i}}}{r\omega}\cos(\mathrm{r}\omega t), \tag{33}$$

respects all robot joint limitations from the datasheet. The number of harmonics in this trajectory is $R = 5$ and the base frequency is $\omega = 0.1\pi$. Table II lists the coefficients $a_{\mathrm{r,i}}$ and $b_{\mathrm{r,i}}$ and also the offset $q_{\mathrm{i},0}$ in rad. These parameter are obtained via an optimization routine, where robot joint and torque limits are provided as constraints. These coefficients are optimized for minimal uncertainty in identifying robot base parameters [28]. Therefore, this trajectory persistently excites the system along the whole bandwidth and subsequently evaluates the estimator (9) performance. Moreover, position $\boldsymbol{x}$ is considered at the robot end-effector. Figure 2 (a) depicts the reference trajectory $\dot{\boldsymbol{x}}$ at the end-effector. For the sake of clarity, we include only the translational trajectory in the simulations and experimental evaluations.

TABLE II: Excitation trajectory parameters

joint i	1	2	3	4	5	6	7
$a_{1,\mathrm{i}}$	0.38	-0.11	0.42	-0.06	-0.17	0.02	0.08
$a_{2,\mathrm{i}}$	-0.26	-0.20	-0.29	0.28	0.72	0.26	0.22
$a_{3,\mathrm{i}}$	-0.77	0.10	0.91	-0.03	-0.39	-0.32	0.89
$a_{4,\mathrm{i}}$	-0.78	0.22	-0.09	0.40	-0.90	0.02	0.17
$a_{5,\mathrm{i}}$	-0.02	-0.08	-0.43	-0.63	-0.29	-0.34	0.14
$b_{1,\mathrm{i}}$	-0.28	0.08	0.08	0.02	-0.06	-0.21	-0.03
$b_{2,\mathrm{i}}$	0.10	-0.23	-0.27	-0.06	0.05	-0.46	0.12
$b_{3,\mathrm{i}}$	-0.37	0.25	0.37	0.03	-0.47	0.53	-0.17
$b_{4,\mathrm{i}}$	0.61	0.05	-0.29	0.22	0.11	-0.52	-0.32
$b_{5,\mathrm{i}}$	-0.59	0.05	-0.37	0.93	-0.55	0.28	0.21
$q_{\mathrm{i},0}$	0.02	-0.76	-0.02	-2.34	-0.03	1.54	0.0

A. Open-loop estimation

In the first part of the simulations, where the controller performance is not of our concern, the system is controlled via inverse dynamics control. However, any other suitable controller may be used as well. The control error in this section is considered to be minimal. In order to have a fair comparison the proposed method, introduced in Sec. III-B, is compared against three other methods. The first method is the state-of-the-art approach, which is explained in Sec. III-A. In Fig. 2, this approach is denoted by $\dot{\boldsymbol{x}}_1$. For the next method we use only IMU m, which is installed on the last link. The Cartesian acceleration at the end-effector can then be computed via

$$\ddot{\boldsymbol{x}} = {}^{0}\boldsymbol{R}_{7}\Big({}^{7}\boldsymbol{R}_{\mathrm{m}}\,{}^{\mathrm{m}}\boldsymbol{a} + \frac{\Delta^{\mathrm{m}}\boldsymbol{\omega}}{\Delta t}\times\left({}^{7}\boldsymbol{x}_{\mathrm{ee}} - {}^{7}\boldsymbol{x}_{S_{\mathrm{m}}}\right) + {}^{\mathrm{m}}\boldsymbol{\omega}\times{}^{\mathrm{m}}\boldsymbol{\omega}\times\left({}^{7}\boldsymbol{x}_{\mathrm{ee}} - {}^{7}\boldsymbol{x}_{S_{\mathrm{m}}}\right)\Big), \tag{34}$$

where $\frac{\Delta^{\mathrm{m}}\boldsymbol{\omega}}{\Delta t}$ is numerical differentiation of the measured angular velocity, ${}^{7}\boldsymbol{x}_{\mathrm{ee}}$ the coordinates of the end-effector in the 7-th joint frame and ${}^{0}\boldsymbol{R}_{7}$ the rotation matrix from joint seven to the robot base. Given that the robot is initially at rest, the Cartesian velocity can be computed by integrating $\ddot{\boldsymbol{x}}$ in (34), as

$$\dot{\boldsymbol{x}} = \int \ddot{\boldsymbol{x}}\ \mathrm{d}t. \tag{35}$$

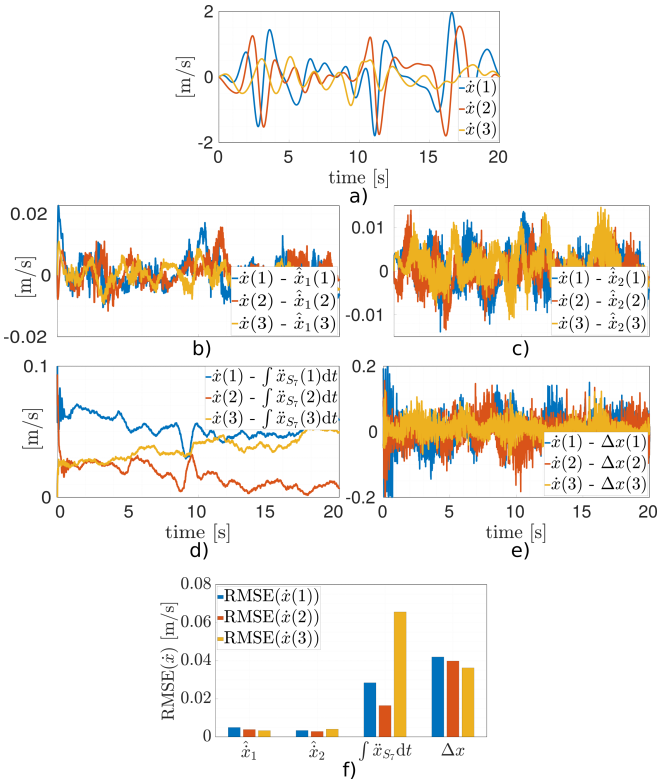

Fig. 2: Comparison of different methods for computing/estimating Cartesian velocity at the robot end-effector. (a) Reference Cartesian velocity profile. (b) Error of estimated Cartesian velocity using observer (9) and forward kinematics (29). (c) Error of estimated Cartesian velocity using observer (19) and forward kinematics (29). (d) Error of computed Cartesian velocity using (34) and (35). (e) Error of computed Cartesian velocity using numerical differentiation (36) and filtering. (f) RMSE($\dot{\boldsymbol{x}}$) for different methods.

In Fig. 2, this approach is denoted by $\int \ddot{\boldsymbol{x}}_{S_7} \mathrm{d}t$. In the next method, symbolized by $\Delta \boldsymbol{x}$, numerical differentiation is utilized to compute the end-effector Cartesian velocity, i.e.

$$\frac{\Delta \boldsymbol{x}}{\Delta t} = \boldsymbol{J}(\boldsymbol{q}) \frac{\Delta \boldsymbol{q}}{\Delta t}. \tag{36}$$

Moreover, signal $\frac{\Delta \boldsymbol{x}}{\Delta t}$ is filtered via a second-order low-pass filter with cutoff frequency 1 Hz to truncate the noise and quantization errors.

The initial observer error is apparent in the state-of-the-art algorithm error $\dot{\boldsymbol{x}} - \hat{\dot{\boldsymbol{x}}}_1$ (see Fig 2 (b)). While, the proposed method eliminates this error adequately (see Fig 2 (c)). Furthermore, projecting the measured IMU signal to the desired coordinates via (34) is not a practical solution (see Fig 2 (d)). Even though, the effect of noise is substantially reduced by integration in (35). The main culprit is the drift in the IMU measurements. Since we fuse the IMU output with the robot proprioceptive measurements in observer (19), the measurement drifts can be resolved relatively suitably. This can be seen in Fig 2 (b) and (c), where the estimation error fluctuates around zero. With more than 20 cm/s error, Fig. 2 (e) shows that computing the velocity via (36) is not numerically stable. Fig. 2 (f) compares the *root mean square error* (RMSE) of the estimated/computed Cartesian velocity over 100 seconds. This figure shows that our proposed algorithm ($\dot{\boldsymbol{x}}_2$), discussed in Sec. III-A, has the best performance, which is slightly better than the state-of-the-art $\dot{\boldsymbol{x}}_1$.

B. Closed-loop estimation

In this part of the simulations we try to utilize the estimated velocity signal in a standard task space controller. For this, we picked QP controller, which is widely used to control a large number of robot manipulators as well as humanoid robots [29]. Let us define the joint and torque constraints as

$$\ddot{\boldsymbol{q}}_{\min} < \ddot{\boldsymbol{q}} < \ddot{\boldsymbol{q}}_{\max}, \tag{37}$$

$$\boldsymbol{\tau}_{\mathrm{m,min}} < \boldsymbol{\tau}_{\mathrm{m}} < \boldsymbol{\tau}_{\mathrm{m,max}}. \tag{38}$$

With the estimated Cartesian velocity, we can define the quadratic objective function

$$\begin{bmatrix} \ddot{\boldsymbol{q}}^* \\ \boldsymbol{\tau}_{\mathrm{m}}^* \end{bmatrix} = \min_{\ddot{\boldsymbol{q}}, \boldsymbol{\tau}_{\mathrm{m}}} \| \begin{bmatrix} \boldsymbol{J}(\hat{\boldsymbol{q}}), & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{q}} \\ \boldsymbol{\tau}_{\mathrm{m}} \end{bmatrix} - \left(\ddot{\boldsymbol{x}}_{\mathrm{d}} - \dot{\boldsymbol{J}}(\hat{\boldsymbol{q}}, \hat{\dot{\boldsymbol{q}}}) \hat{\dot{\boldsymbol{q}}} \right) \|$$
$$\text{s.t. } (1), (37), (38) \tag{39}$$

which determines at each iteration the optimal input $(\boldsymbol{\tau}_{\mathrm{m}}^*, \ddot{\boldsymbol{q}}^*)$ for system (1) based on the desired trajectory $\ddot{\boldsymbol{x}}_{\mathrm{d}}$. In the proposed closed-loop scheme, the optimal solution $\ddot{q}_{\mathrm{i}}^* \in \ddot{\boldsymbol{q}}^*$ is the desired trajectory $\ddot{q}_{\mathrm{d,i}}$ in (20). In order to stabilize the closed loop system, we can use the stabilizing state feedback [29]

$$\ddot{\boldsymbol{x}}_{\mathrm{d}} = \ddot{\boldsymbol{x}}_{\mathrm{ref}} + \boldsymbol{K}_D \left(\dot{\boldsymbol{x}}_{\mathrm{ref}} - \hat{\dot{\boldsymbol{x}}} \right) + \boldsymbol{K}_P \left(\boldsymbol{x}_{\mathrm{ref}} - \hat{\boldsymbol{x}} \right), \tag{40}$$

with $\boldsymbol{K}_D = \mathrm{diag}(K_{D,1}, \cdots, K_{D,6}) \in \mathbb{R}^{6\times 6}$ and $\boldsymbol{K}_P = \mathrm{diag}(K_{P,1}, \cdots, K_{P,6}) \in \mathbb{R}^{6\times 6}$ being the feedback gains. These gains are selected such that matrix

$$\begin{bmatrix} \boldsymbol{0}_{6\times 6} & \boldsymbol{I}_6 \\ \boldsymbol{K}_P & \boldsymbol{K}_D \end{bmatrix}$$

is Hurwitz. Also, $\ddot{\boldsymbol{x}}_{\mathrm{ref}}$ is the reference trajectory. This trajectory can intrinsically represent force or joint space trajectory, described in task space Cartesian acceleration. Figure 3 depicts the closed loop system block diagram with QP controller. For the sake of clarity we assume that robot dynamics model variables are numerically available at each iteration, as most modern robot manipulators provide this information through their interface. Figure 4 depicts the controller tracking error

$$\boldsymbol{e}_{\mathrm{qp}} = \dot{\boldsymbol{x}}_{\mathrm{ref}} - \dot{\boldsymbol{x}}. \tag{41}$$

Note that this error ($\boldsymbol{e}_{\mathrm{qp}} \in \mathbb{R}^6$) is obtained based on the ground truth trajectory $\dot{\boldsymbol{x}}$. According to Fig. 4 (a) observer (9) and subsequently tracking performance suffers from the observer initial error. The tracking error is denoted by $e_{\mathrm{qp},o2}$ in this figure. The tracking errors in Fig. 4 (a)-(d) can be justified based on the estimation/computation errors, explained in the previous section (Sec. IV-A). According to Fig. 4 (e) the tracking error with observer (19) (denoted by $e_{\mathrm{qp},o1}$) is comparatively the smallest one.

V. Experimental Evaluation

In this section, we evaluate the observer (9) performance in an experiment with a 7-DoF robot arm. To the best of authors' knowledge, no available manipulator comes with installed IMUs thus far. Therefore, we manually install the sensor in this experiment. Note that all robot links have to

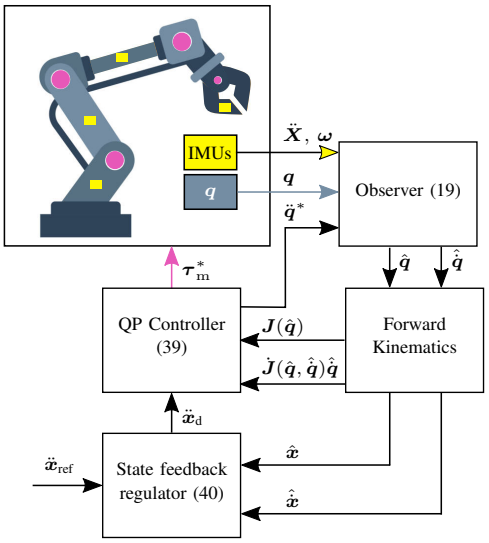

Fig. 3: Proposed closed-loop system block diagram

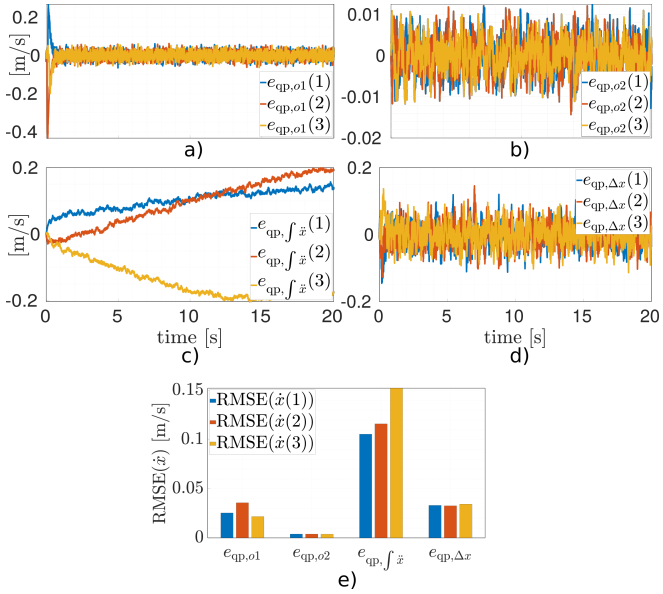

Fig. 4: Comparison of QP controller tracking error when the task velocity is computed/estimated via different methods. Specifically, tracking error when the Cartesian velocity is (a) estimated using observer (9) and forward kinematics (29), (b) estimated using observer (19) and forward kinematics (29), (c) computed using (34) and (35) and (d) computed using numerical differentiation (36) and filtering, (e) RMSE($\dot{x}$) for different methods.

be equipped with IMUs, in order to be able to fully actuate the robot manipulator and estimate joint variables. This assumption however, falls short in the current setup, where one link is considered to be actuated. More specifically, the fourth joint is actuated by a sinusoidal motion with different velocities. Different velocities let us evaluate our method bandwidth. Figure 5 depicts our experiment setup. Figure 6 (a) and (b) plots the ground truth trajectory in joint and care. In order to generate the ground truth trajectory, the motor-side position θ_4 and the joint torque τ_4 are measured. It is also assumed that the sensor transformation matrix from the sensor frame to the moving joint frame is fully known. This is done by careful measurements before the experiment. Therefore, ${}^4\boldsymbol{x}_{S_m}$ in (10) and ${}^m\boldsymbol{R}_4$ in (11) are available.

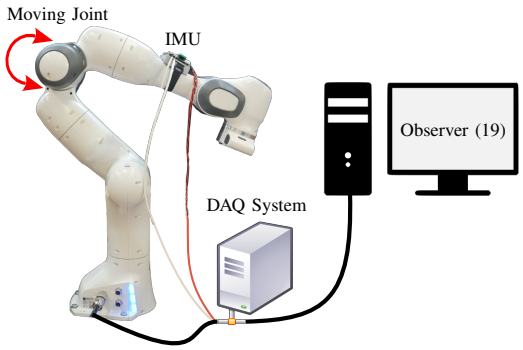

Fig. 5: Schematics of experimental setup. All measurements (IMU and robot joint position measurements) are collected in a *Data Acquisition* (DAQ) system and sent and processed in a PC, where the observer estimates the Cartesian velocity of the end-effector.

Moreover, EKF is used to estimate the Cartesian velocity at the end-effector. Figure 6 (c) shows the estimation error of state-of-the-art observer (9). The observer initial error is apparent in this plot. Figure 6 (d) depicts the error of the estimated Cartesian velocity using the proposed observer (19). This error is below 2 cm/s along the trajectory. Furthermore, the method in which, the accelerometer measurements are transformed and integrated using (34) and (35) respectively, is also examined in this experiment. The corresponding error is mainly caused by the sensor drift (see Fig. 6 (e)). For example, the second element of Cartesian velocity is zero in this trajectory. However, due to the measurement drift, the error grows up to 4 cm/s. Since the sensor signal is integrated over time in this method, the drift effect grows relatively quickly. Lastly, the Cartesian velocity is computed via numerical differentiation (according to (36)) and low-pass filtering with the cutoff frequency of 5 Hz. The error corresponding to this method is depicted in Fig. 6 (f). This error is majorly due to the filter phase-shift effects. Figure 6 (g) compares RMSE($\dot{x}$) for the aforementioned approaches. Accordingly, the proposed method outperforms the state-of-the-art solutions.

VI. Conclusions

We discussed the state-of-the-art method for estimating joint velocity and acceleration, which is prone to numerical instabilities, caused by discontinuities in the robot manipulator trajectory. A technique for overcoming the limitations of the stat-the-art method is introduced in this paper. Similar to the state-of-the-art algorithm, the proposed approach is based on the fusion of multiple sensors, namely IMUs and motor encoders. It also exploits the information of the robot desired trajectory. Although, the requirement of the latter information decreases the modularity of the proposed approach, it ensures stable and smooth outputs. As a result theses outputs can be safely utilized in state-feedback systems. Closed- and open-loop performance of the method in simulation and experiments respectively, support our claims. The closed-loop performance of this technique in real-world scenarios, which is compared to the state-of-the-art methods such as [30] and [31], is an interesting future-work topic. For this, we need to address wider task-space error functions and constraints close to e.g., [32].

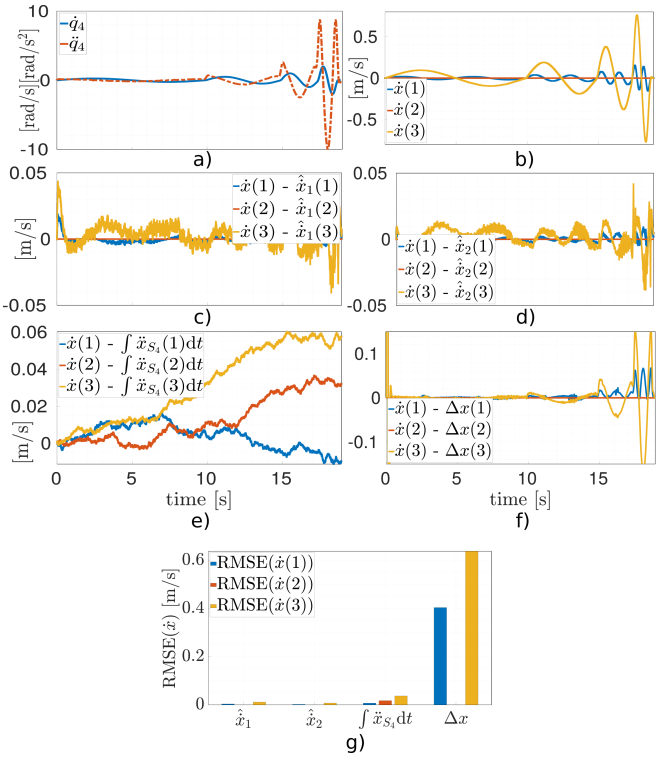

Fig. 6: Comparison of different methods for computing/estimating Cartesian velocity at the robot end-effector. (a) Reference joint velocity and acceleration. (b) Reference Cartesian velocity profile. (c) Error of estimated Cartesian velocity using the observer (9) and forward kinematics (29). (d) Error of estimated Cartesian velocity using the observer (19) and forward kinematics (29). (e) Error of computed Cartesian velocity using (34) and (35). (f) Error of computed Cartesian velocity using numerical differentiation (36) and filtering. (g) RMSE($\dot{\boldsymbol{x}}$) for different methods.

REFERENCES

[1] D. Simon, *Optimal state estimation: Kalman, H infinity, and nonlinear approaches*. John Wiley & Sons, 2006.

[2] R. Smits, T. De Laet, K. Claes, H. Bruyninckx, and J. De Schutter, "iTASC: a tool for multi-sensor integration in robot manipulation," in *2008 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*. IEEE, 2008, pp. 426–433.

[3] J. De Schutter, T. De Laet, J. Rutgeerts, W. Decré, R. Smits, E. Aertbeliën, K. Claes, and H. Bruyninckx, "Constraint-based task specification and estimation for sensor-based robot systems in the presence of geometric uncertainty," *The International Journal of Robotics Research*, vol. 26, no. 5, pp. 433–455, 2007.

[4] C. Yang, Y. Jiang, W. He, J. Na, Z. Li, and B. Xu, "Adaptive parameter estimation and control design for robot manipulators with finite-time convergence," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 10, pp. 8112–8123, 2018.

[5] P. Hyatt, D. Kraus, V. Sherrod, L. Rupert, N. Day, and M. D. Killpack, "Configuration estimation for accurate position control of large-scale soft robots," *IEEE/ASME Transactions on Mechatronics*, vol. 24, no. 1, pp. 88–99, 2018.

[6] R. Hartley, J. Mangelson, L. Gan, M. G. Jadidi, J. M. Walls, R. M. Eustice, and J. W. Grizzle, "Legged robot state-estimation through combined forward kinematic and preintegrated contact factors," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 4422–4429.

[7] H. Cheng, B. Lan, H. Liu, X. Wang, and B. Liang, "VLC-SE: Visual-lengthwise configuration self-estimator of continuum robots," in *2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)*. IEEE, 2021, pp. 1428–1434.

[8] L. Rupert, T. Duggan, and M. D. Killpack, "Improved continuum joint configuration estimation using a linear combination of length measurements and optimization of sensor placement," *Frontiers in Robotics and AI*, vol. 8, 2021.

[9] M. Benallegue, P. Gergondet, H. Audrerr, A. Mifsud, M. Morisawa, F. Lamiraux, A. Kheddar, and F. Kanehiro, "Model-based external force/moment estimation for humanoid robots with no torque measurement," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 3122–3129.

[10] V. S. Raghavan, D. Kanoulas, C. Zhou, D. G. Caldwell, and N. G. Tsagarakis, "A study on low-drift state estimation for humanoid locomotion, using LiDAR and kinematic-inertial data fusion," in *2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2018, pp. 1–8.

[11] R. Cisneros, A. Benallegue, Y. Chitour, M. Morisawa, F. Kanehiro *et al.*, "Lyapunov-stable orientation estimator for humanoid robots," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6371–6378, 2020.

[12] M. Schepers, M. Giuberti, G. Bellusci *et al.*, "Xsens MVN: Consistent tracking of human motion using inertial sensing," *Xsens Technol*, vol. 1, no. 8, 2018.

[13] L. Sy, M. Raitor, M. Del Rosario, H. Khamis, L. Kark, N. H. Lovell, and S. J. Redmond, "Estimating lower limb kinematics using a reduced wearable sensor count," *IEEE Transactions on Biomedical Engineering*, vol. 68, no. 4, pp. 1293–1304, 2020.

[14] A. Dash, A. Yadav, A. Chauhan, and U. Lahiri, "Kinect-assisted performance-sensitive upper limb exercise platform for post-stroke survivors," *Frontiers in Neuroscience*, vol. 13, p. 228, 2019.

[15] S. A. B. Birjandi, J. Kühn, and S. Haddadin, "Joint velocity and acceleration estimation in serial chain rigid body and flexible joint manipulators," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Nov 2019, pp. 7503–7509.

[16] S. A. B. Birjandi, J. Kühn, and S. Haddadin, "Observer-extended direct method for collision monitoring in robot manipulators using proprioception and imu sensing," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 954–961, April 2020.

[17] M. W. Spong, "Modeling and control of elastic joint robots," *Journal of dynamic systems, measurement, and control*, vol. 109, no. 4, pp. 310–318, 1987.

[18] C. Ott, *Cartesian impedance control of redundant and flexible-joint robots*. Springer, 2008.

[19] B. Siciliano, O. Khatib, and T. Kröger, *Springer handbook of robotics*. Springer, 2008, vol. 200.

[20] S. A. B. Birjandi and S. Haddadin, "Model-adaptive high-speed collision detection for serial-chain robot manipulators," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6544–6551, 2020.

[21] R. Hermann and A. Krener, "Nonlinear controllability and observability," *IEEE Transactions on automatic control*, vol. 22, no. 5, pp. 728–740, 1977.

[22] A. Isidori, *Nonlinear control systems: an introduction*. Springer, 1985.

[23] K. Yano, *The theory of Lie derivatives and its applications*. Courier Dover Publications, 2020.

[24] C. Letellier, L. A. Aguirre, and J. Maquet, "Relation between observability and differential embeddings for nonlinear dynamics," *Physical Review E*, vol. 71, no. 6, p. 066213, 2005.

[25] G. Besançon, *Nonlinear observers and applications*. Springer, 2007, vol. 363.

[26] C. R. Gaz, M. Cognetti, A. A. Oliva, P. R. Giordano, and A. De Luca, "Dynamic identification of the franka emika panda robot with retrieval of feasible parameters using penalty-based optimization," *IEEE Robotics and Automation Letters*, 2019.

[27] Bosch, *BMI055 small, versatile 6DoF sensor module*, Bosch, 7 2014, rev. 1.2.

[28] J. Swevers, C. Ganseman, D. B. Tukel, J. De Schutter, and H. Van Brussel, "Optimal robot excitation and identification," *IEEE transactions on robotics and automation*, vol. 13, no. 5, pp. 730–740, 1997.

[29] K. Bouyarmane and A. Kheddar, "On weight-prioritized multitask control of humanoid robots," *IEEE Transactions on Automatic Control*, vol. 63, no. 6, pp. 1632–1647, 2017.

[30] B. Olofsson, J. Antonsson, H. G. Kortier, B. Bernhardsson, A. Robertsson, and R. Johansson, "Sensor fusion for robotic workspace state estimation," *IEEE/ASME Transactions on Mechatronics*, vol. 21, no. 5, pp. 2236–2248, 2015.

[31] P. Axelsson, R. Karlsson, and M. Norrlöf, "Bayesian state estimation of a flexible industrial robot," *Control Engineering Practice*, vol. 20, no. 11, pp. 1220–1228, 2012.

[32] K. Bouyarmane, J. Vaillant, K. Chappellet, and A. Kheddar, "Multi-robot and task-space force control with quadratic programming," *IEEE Robotics and Automation Letters*, 2017.