Calibration-free IMU-based Kinematic State Estimation for Robotic Manipulators*

Michael Fennel, Lukas Driller, Antonio Zea, Uwe D. Hanebeck

***Abstract*— The precise knowledge of a robot manipulator's kinematic state including position, velocity, and acceleration is one of the base requirements for the application of advanced control algorithms. To obtain this information, encoder data could be differentiated numerically. However, the resulting velocity and acceleration estimates are either noisy or delayed as a result of low-pass filtering. Numerical differentiation can be circumvented by the utilization of gyroscopes and accelerometers, but these suffer from a variety of measurement errors and nonlinearity regarding the desired quantities. Therefore, we present a novel, real-time capable kinematic state estimator based on the Extended Kalman filter with states for the effective sensor biases. This way, the handling of arbitrary inertial sensor setups is made possible without calibration on manipulators composed of revolute and prismatic joints. Simulation experiments show that the proposed estimator is robust towards various error sources and that it outperforms competing approaches. Moreover, the practical relevance is demonstrated using a real manipulator with two joints.**

I. Introduction

In theory, estimating a manipulator's kinematic state consisting of joint position, velocity, and acceleration is a straightforward task when joint position measurements are available. In practice, however, noise and discretization of the joint encoders will result in poor signal quality and delayed signals after the numerical differentiation required for velocity and acceleration estimates. To compensate for the noise introduced, the signals can be filtered. However, this introduces phase delays for higher frequencies resulting in further delayed estimates. While this is not a serious issue in classical industrial robot applications, it represents a challenge in the context of collaborative robots, where an accurate and non-delayed kinematic state estimation is among the necessary information for fast collision detection [1]. Beyond that, kinematic state data is required in a variety of robotic disciplines, ranging from model identification [2] over trajectory tracking [3] to force control [4].

A promising option to overcome the limitations of pure encoder sensor setups regarding the estimation of velocities and accelerations is the addition of inertial measurement units (IMUs), consisting of gyroscopes and accelerometers, to the manipulator. Driven by reduced costs and the increased availability, the integration and utilization of such sensors has become an actively researched topic in the last two decades: In [5], Zhu et al. presented two fusion approaches that utilize frequency weighting to combine acceleration measurements with encoder signals to estimate the end effector velocity of a manipulator in 3D space. In a similar paper, Hedberg et al. [6] presented an algorithm to fuse angular rate and linear acceleration measurements from end effector mounted sensors with the pose estimate from forward kinematics to improve information about the end effector trajectory. Although both authors utilize acceleration measurements, they do not provide smoothed acceleration estimates beyond the raw, noisy acceleration measurements. Furthermore, all estimates are provided in Cartesian space instead of joint space, even though the latter is more universal due to the unambiguity of forward kinematics. This limitation is compensated in the work of Chen et al. [7], where an end effector accelerometer is used to estimate the joint states of a robot with elastic actuators that do not possess load side encoders. For this, a dynamic model of the robot is paired with an optimization problem that has an acceleration constraint and a set of decoupled Kalman filters. It is obvious to deploy multiple IMUs if an estimation in the joint space instead of the Cartesian space is desired. For this reason, Vihonen et al. [8] proposed a sensor configuration where one triaxial gyroscope is attached to each link. The authors also show how the joint acceleration can be determined by adding one additional triaxial and three single-axis accelerometers. Similar to this, Rotella et al. introduced a setup in [9] with one triaxial gyroscope and two triaxial accelerometers attached to each link to calculate joint velocities and accelerations directly from raw measurements in a first step. In a second step, two Kalman filters are used to smooth the data and to ensure consistency between the position data and its derivatives.

Although a large number of sensors can result in improved accuracy, it can be beneficial to reduce the number of sensors for economical and constructive reasons. For this reason, Birjandi et al. presented an estimator in [10], [11] that utilizes one accelerometer and optionally one gyroscope per link in combination with an extended Kalman filter (EKF) to estimate all joint velocities and accelerations.

All solutions up to this point share the disadvantage of requiring an accurate sensor calibration regarding placement, bias, and scale factor, which is hard to achieve, especially in industrial environments. In contrast, the method in [12] uses one uncalibrated accelerometer per link to estimate a bias-polluted joint acceleration vector. Thereafter, the smoothened (and therefore delayed), but bias-free accelerations calculated from encoder data are used to compensate for the current acceleration bias. Although this method works well in

*This work was supported by the ROBDEKON project of the German Federal Ministry of Education and Research.

All authors are with the Intelligent Sensor-Actuator-Systems Laboratory (ISAS), Institute for Anthropomatics and Robotics, Karlsruhe Institute of Technology (KIT), Germany michael.fennel@kit.edu, lukas.driller@student.kit.edu, antonio.zea@kit.edu, uwe.hanebeck@kit.edu.

the given acceleration control application, it can neither incorporate gyroscope data nor provide velocity estimates beyond filtered position derivatives. This becomes a practical limitation if the last accelerometer is situated close to the revolute axis of the last joint.

This paper aims to overcome the presented limitations of the current state of the art by contributing a novel, unified, EKF-based framework for estimating the kinematic state of robot manipulators, which

- is hard real-time capable (i.e., execution frequency $\geq 1\,\mathrm{kHz}$),
- does not depend on accurate sensor calibration information, such as exact pose, scale factor error, bias, and cross-axis-sensitivity,
- can be used with arbitrary inertial sensor setups, and
- provides good estimation quality regarding all kinematic quantities over a high bandwidth.

The remainder of this paper is structured as follows: Based on the notation from Section II, the problem is formalized in Section III. In Section IV, the state estimation including system and measurement models is introduced. Finally, a detailed evaluation based on simulations and experiments on real hardware is presented in Section V and Section VI, respectively, before the paper is concluded in Section VII.

II. Notation

Throughout this paper, vectors are printed underlined and matrices are printed in bold. Positions, linear velocities and linear accelerations are denoted with $\underline{x}$, $\underline{v}$, and $\underline{a}$, respectively. For rotations, rotation matrices $\mathbf{C}$ are used in combination with angular velocities $\underline{\omega}$ and angular accelerations $\underline{\alpha}$. Furthermore, $\underline{q}$ is the generalized joint configuration vector. The unit or zero vectors $\underline{r}$ and $\underline{t}$ define the rotational and translational axes of joints, respectively. This means that $\underline{r} = \underline{0}_3$ holds for prismatic joints and $\underline{t} = \underline{0}_3$ for revolute joints. Superscripts denote the resolving coordinate frame in which a quantity is expressed. The subscripts denote the reference frame (if applicable) and the object frame. For example, $\underline{x}^A_{BC}$ represents the position of point C relative to frame B, given in coordinates of frame A. Similarly, $\mathbf{R}^A_B$ describes the orientation of frame B relative to frame A, resulting in a set of mathematical rules that are explained in depth in [13]. For brevity, the cross product $\underline{\omega} \times \underline{x}$ can be written as $\mathbf{\Omega}\,\underline{x}$, where the uppercase $\mathbf{\Omega} = [\underline{\omega}\times]$ is the skew-symmetric matrix of the lowercase $\underline{\omega}$. This definition is also valid for other vectors, e.g., $\underline{r}$. To reference the i-th element of a vector, $[i]$ is appended to the subscript. Likewise, the notation $[:i]$ can be used to address the first to i-th element of a vector. All quantities are given in SI units unless otherwise specified.

In the remainder, the coordinate frames as illustrated in Fig. 1 are used. The world frame W is considered an inertial frame and marks the stationary base of the robot. Each joint i of the robot is located between the *end frame* of the previous link ϵ_{i-1} and the *begin frame* of the next link β_i.

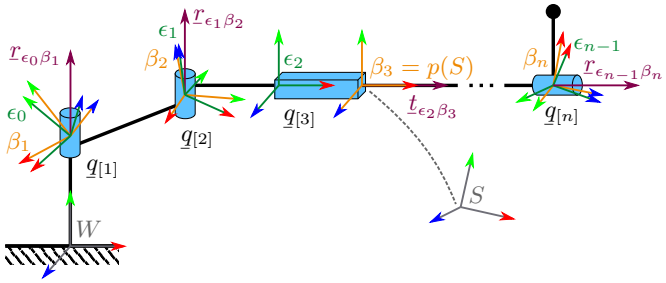

Fig. 1: Definition of the frames on the considered kinematic chain with the link begin frames, the link end frames, and the joint axes. An inertial sensor S is attached to the link after joint 3 in this example.

III. Problem Formulation

In this paper, a serial kinematic chain with n joints, which are either prismatic or revolute joints, is considered. It is assumed that the forward kinematics is known precisely and that each joint provides noisy (load-side) position measurements from an encoder. Along the kinematic chain, $n_\mathrm{a} \geq 0$ triaxial accelerometers $A = \{A_1, \ldots, A_{n_\mathrm{a}}\}$ and $n_\mathrm{g} \geq 0$ triaxial gyroscopes $G = \{G_1, \ldots, G_{n_\mathrm{g}}\}$ are distributed in an arbitrary fashion.[1] For each sensor $S \in G \cup A$, the mounting pose of the sensing frame relative to the parent link begin frame $p(S)$, i.e., $\underline{x}^{p(S)}_{p(S)S}$ and $\mathbf{C}^{p(S)}_S$, is known within typical manufacturing tolerances. However, the exact calibration of the sensors, including bias, scale-factor error, cross-axis-sensitivity, nonlinearity, and pose offset (e.g., originating from soldering), is not known.

The goal in this setup is to find accurate and undelayed estimates of the real joint positions $\underline{q} \in \mathbb{R}^n$, velocities $\dot{\underline{q}} \in \mathbb{R}^n$, and accelerations $\ddot{\underline{q}} \in \mathbb{R}^n$ based on the raw measurements from encoders $\tilde{\underline{q}} \in \mathbb{R}^n$, gyroscopes $\tilde{\underline{\omega}}^{G_1}_{WG_1}, \ldots, \tilde{\underline{\omega}}^{G_{n_\mathrm{g}}}_{WG_{n_\mathrm{g}}} \in \mathbb{R}^3$, and accelerometers $\tilde{\underline{a}}^{A_1}_{WA_1}, \ldots, \tilde{\underline{a}}^{A_{n_\mathrm{a}}}_{WA_{n_\mathrm{a}}} \in \mathbb{R}^3$. Here, the term undelayed encompasses not only phase delays but also the calculation under real-time constraints.

IV. Estimator Design

In this section, the estimator that combines all available measurements into a single kinematic state is explained. We chose an EKF architecture for its ability to handle nonlinear measurement functions, avoiding the introduction of sensor placement constraints as seen in [8] and [9]. Furthermore, the Kalman filter facilitates intuitive parameter tuning.

A. System Model

For the filter, a linear constant jerk motion model including all joints at once is combined with a constant bias model

[1] Although the distribution can be arbitrary, it is preferred to have a configuration that effectively permits the observation of joint velocities and accelerations from the inertial sensor measurements.

resulting in the time-discrete system model

$$\begin{pmatrix} \underline{q} \\ \dot{\underline{q}} \\ \ddot{\underline{q}} \\ \dddot{\underline{q}} \\ \underline{b}_{\mathrm{g}} \\ \underline{b}_{\mathrm{a}} \end{pmatrix}_{k+1} = \begin{pmatrix} \mathbf{I} & \Delta t\mathbf{I} & \frac{1}{2}\Delta t^2\mathbf{I} & \frac{1}{6}\Delta t^3\mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \Delta t\mathbf{I} & \frac{1}{2}\Delta t^2\mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \Delta t\mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} \end{pmatrix} \begin{pmatrix} \underline{q} \\ \dot{\underline{q}} \\ \ddot{\underline{q}} \\ \dddot{\underline{q}} \\ \underline{b}_{\mathrm{g}} \\ \underline{b}_{\mathrm{a}} \end{pmatrix}_{k} + \underline{w}_k \quad (1)$$

with the sampling time Δt. Here, the stacked biases

$$\underline{b}_{\mathrm{g}} = (\underline{b}_{G_1}^{\mathsf{T}} \cdots \underline{b}_{G_{n_{\mathrm{g}}}}^{\mathsf{T}})^{\mathsf{T}} \in \mathbb{R}^{3n_{\mathrm{g}}} \quad \text{and} \quad (2)$$

$$\underline{b}_{\mathrm{a}} = (\underline{b}_{A_1}^{\mathsf{T}} \cdots \underline{b}_{A_{n_{\mathrm{a}}}}^{\mathsf{T}})^{\mathsf{T}} \in \mathbb{R}^{3n_{\mathrm{a}}} \quad (3)$$

describe the current bias values of each triaxial gyroscope and accelerometer, respectively, which are equal to the actual inertial sensor biases under ideal conditions. Under real conditions, $\underline{b}_{\mathrm{g}}$ and $\underline{b}_{\mathrm{a}}$ shall approximate the current total of all measurement errors originating from the inertial sensors, allowing the estimator to be calibration-free. The zero-mean, Gaussian process noise $\underline{w}_k$ is assumed to exclusively affect the joint jerk and the bias states, i.e.,

$$\mathrm{Cov}(\underline{w}_k) = \mathrm{diag}\big(\underline{0}_n^{\mathsf{T}}\ \underline{0}_n^{\mathsf{T}}\ \underline{0}_n^{\mathsf{T}}\ \sigma_{\dddot{q}}^2\underline{1}_n^{\mathsf{T}}\ \sigma_{b_g}^2\underline{1}_{3n_{\mathrm{g}}}^{\mathsf{T}}\ \sigma_{b_a}^2\underline{1}_{3n_{\mathrm{a}}}^{\mathsf{T}}\big)\,, \quad (4)$$

where $\sigma_{\square}$ is the respective standard deviation.

Note that other formulations of the system model, which have inputs based on controller setpoints or forward dynamics, might be more accurate due to the incorporation of additional information. However, this would impose non-trivial constraints on the robot architecture, violating the promise of a unified kinematic state estimation.

B. Measurement Model

The measurement vector

$$\tilde{\underline{y}} = \begin{pmatrix} \tilde{\underline{q}} \\ \tilde{\underline{\omega}}_{WG_1}^{G_1} \\ \vdots \\ \tilde{\underline{\omega}}_{WG_{n_{\mathrm{g}}}}^{G_{n_{\mathrm{g}}}} \\ \tilde{\underline{a}}_{WA_1}^{A_1} \\ \vdots \\ \tilde{\underline{a}}_{WA_{n_{\mathrm{a}}}}^{A_{n_{\mathrm{a}}}} \end{pmatrix} = \begin{pmatrix} \underline{q} \\ \underline{\omega}_{WG_1}^{G_1}(\underline{q}, \dot{\underline{q}}) + \underline{b}_{G_1} \\ \vdots \\ \underline{\omega}_{WG_{n_{\mathrm{g}}}}^{G_{n_{\mathrm{g}}}}(\underline{q}, \dot{\underline{q}}) + \underline{b}_{G_{n_{\mathrm{g}}}} \\ \underline{a}_{WA_1}^{A_1}(\underline{q}, \dot{\underline{q}}, \ddot{\underline{q}}) + \underline{b}_{A_1} \\ \vdots \\ \underline{a}_{WA_{n_{\mathrm{a}}}}^{A_{n_{\mathrm{a}}}}(\underline{q}, \dot{\underline{q}}, \ddot{\underline{q}}) + \underline{b}_{A_{n_{\mathrm{a}}}} \end{pmatrix} + \underline{v} \quad (5)$$

stacks the measurements from all available sensors. The measured angular rates $\tilde{\underline{\omega}}_{WG_i}^{G_i}$ are assumed to be the sum of the actual angular rates $\underline{\omega}_{WG_i}^{G_i}$, the current total bias $\underline{b}_{G_i}$, and the corresponding entries of the measurement noise $\underline{v} \in \mathbb{R}^{n+3n_{\mathrm{g}}+3n_{\mathrm{a}}}$. The composition of the measured linear accelerations $\tilde{\underline{a}}_{WA_i}^{A_i}$ follows the same scheme. With the assumption of uncorrelated sensor noise, the covariance matrix of $\underline{v}$ is diagonal and its values correspond to the actual noise intensity of the sensors.

In contrast to [10], this approach does neither rely on a minimum number of gyros and accelerometers per link nor are the joints considered in a decoupled fashion. As a result, the initially stated requirement for flexibility can be met. Furthermore, inertial sensors can be disabled at runtime, e.g., when a sensor is faulty or reaches its range limits.

C. Measurement Jacobian

The measurement model (5) is nonlinear. To apply the EKF in this case, the Jacobian of the measurement function $\tilde{\underline{y}}$ with respect to the state variables from (1) is required. A simple solution for this involves numerical derivatives, but the sought Jacobian is $\in \mathbb{R}^{(n+3n_{\mathrm{g}}+3n_{\mathrm{a}})\times(4n+3n_{\mathrm{g}}+3n_{\mathrm{a}})}$, which requires many computationally expensive calculations of the forward kinematics. Therefore, it is better to calculate the Jacobian analytically in form of a recursive expression. In [14], the required kinematic derivatives were introduced as a by-product. However, given that the underlying spatial algebra impedes mathematical accessibility and interpretability, the following recursive algorithm with complexity $\mathcal{O}(n^2)$ utilizing standard algebra was created.

The partial derivatives for $\underline{b}_{\mathrm{g}}$ and $\underline{b}_{\mathrm{a}}$ as well as $\ddot{\underline{q}}$ are trivial, because they appear linearly or not at all in the components of (5). As a first step for $\underline{\omega}_{WG_i}^{G_i}$, the angular velocity of the link begin frame β_j is described using the angular velocity of the previous link begin frame β_{j-1} and the movement of joint j using

$$\underline{\omega}_{W\beta_j}^{W} = \underline{\omega}_{W\beta_{j-1}}^{W} + \mathbf{C}_{\beta_{j-1}}^{W}\mathbf{C}_{\epsilon_{j-1}}^{\beta_{j-1}}\underline{r}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}}\dot{\underline{q}}_{[j]}\,. \quad (6)$$

Changing the resolving frame to β_j yields

$$\underline{\omega}_{W\beta_j}^{\beta_j} = \underbrace{\mathbf{C}_{\epsilon_{j-1}}^{\beta_j}}_{\underline{q}_{[j]}}\mathbf{C}_{\beta_{j-1}}^{\epsilon_{j-1}}\Big(\underbrace{\underline{\omega}_{W\beta_{j-1}}^{\beta_{j-1}}}_{\underline{q}_{[:j-1]},\dot{\underline{q}}_{[:j-1]}} + \mathbf{C}_{\epsilon_{j-1}}^{\beta_{j-1}}\underline{r}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}}\underbrace{\dot{\underline{q}}_{[j]}}_{\dot{\underline{q}}_{[j]}}\Big)\,. \quad (7)$$

Here, all dependencies on the state variables are listed under the curly brackets. This way, it is easy to see that

$$\frac{\partial\underline{\omega}_{W\beta_j}^{\beta_j}}{\partial\underline{q}_{[i]}} = \mathbf{C}_{\beta_{j-1}}^{\beta_j}\frac{\partial\underline{\omega}_{W\beta_{j-1}}^{\beta_{j-1}}}{\partial\underline{q}_{[i]}} \quad (8)$$

for $i < j$ and

$$\frac{\partial\underline{\omega}_{W\beta_j}^{\beta_j}}{\partial\underline{q}_{[j]}} = -\mathbf{R}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}}\underline{\omega}_{W\beta_j}^{\beta_j} \quad (9)$$

by applying $\frac{\partial\mathbf{C}_{\epsilon_{j-1}}^{\beta_j}}{\partial\underline{q}_{[j]}} = -\mathbf{R}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}}\mathbf{C}_{\epsilon_{j-1}}^{\beta_j}$. Consequently, the partial derivative of $\underline{\omega}_{W\beta_j}^{\beta_j}$ with respect to $\underline{q}_{[i]}$ is defined recursively as

$$\frac{\partial\underline{\omega}_{W\beta_j}^{\beta_j}}{\partial\underline{q}_{[i]}} = \begin{cases} \mathbf{C}_{\beta_{j-1}}^{\beta_j}\frac{\partial\underline{\omega}_{W\beta_{j-1}}^{\beta_{j-1}}}{\partial\underline{q}_{[i]}} & i < j \\ -\mathbf{R}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}}\underline{\omega}_{W\beta_j}^{\beta_j} & i = j \\ \underline{0} & i > j \end{cases}. \quad (10)$$

With the same argumentation, the partial derivative with respect to $\dot{\underline{q}}_{[i]}$

$$\frac{\partial\underline{\omega}_{W\beta_j}^{\beta_j}}{\partial\dot{\underline{q}}_{[i]}} = \begin{cases} \mathbf{C}_{\beta_{j-1}}^{\beta_j}\frac{\partial\underline{\omega}_{W\beta_{j-1}}^{\beta_{j-1}}}{\partial\dot{\underline{q}}_{[i]}} & i < j \\ \mathbf{C}_{\epsilon_{j-1}}^{\beta_j}\underline{r}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} & i = j \\ \underline{0} & i > j \end{cases} \quad (11)$$

is calculated recursively. The angular velocity of the gyroscope G_i is determined from the angular velocity of the

parent link begin frame $k = p(G_i)$ via

$$\underline{\omega}_{WG_i}^{G_i} = \mathbf{C}_{\beta_k}^{G_i} \underline{\omega}_{W\beta_k}^{\beta_k} . \tag{12}$$

Finally, the desired partial derivatives of the gyroscope measurement with respect to the state variables are

$$\frac{\partial \underline{\omega}_{WG_i}^{G_i}}{\partial \underline{q}} = \mathbf{C}_{\beta_k}^{G_i} \frac{\partial \underline{\omega}_{W\beta_k}^{\beta_k}}{\partial \underline{q}} \quad \text{and} \tag{13}$$

$$\frac{\partial \underline{\omega}_{WG_i}^{G_i}}{\partial \dot{\underline{q}}} = \mathbf{C}_{\beta_k}^{G_i} \frac{\partial \underline{\omega}_{W\beta_k}^{\beta_k}}{\partial \dot{\underline{q}}} . \tag{14}$$

This scheme can be applied to linear accelerations as well. Unfortunately, the resulting formulas are lengthy and we refer the reader to the supplemental material[2] for a detailed derivation. For this reason, we just outline the key formulas at this point. The recursive formula for the position of the link begin frame β_j in the world frame is

$$\underline{x}_{W\beta_j}^{W} = \underline{x}_{W\beta_{j-1}}^{W} + \mathbf{C}_{\beta_{j-1}}^{W} \left(\underline{x}_{\beta_{j-1}\epsilon_{j-1}}^{\beta_{j-1}} + \mathbf{C}_{\epsilon_{j-1}}^{\beta_{j-1}} \underline{t}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} \underbrace{q_{[j]}}_{\underline{q}_{[:j-1]}, \dot{\underline{q}}_{[:j-1]}} \right) . \tag{15}$$

If this is differentiated twice with respect to time and then resolved in the link begin frame β_i,

$$\begin{aligned} \underline{a}_{W\beta_j}^{\beta_j} = {} & \underbrace{\mathbf{C}_{\epsilon_{j-1}}^{\beta_j}}_{\underline{q}_{[j]}} \mathbf{C}_{\beta_{j-1}}^{\epsilon_{j-1}} \Bigg[2 \underbrace{\mathbf{\Omega}_{W\beta_{j-1}}^{\beta_{j-1}}}_{\underline{q}_{[:j-1]}, \dot{\underline{q}}_{[:j-1]}} \mathbf{C}_{\epsilon_{j-1}}^{\beta_{j-1}} \underline{t}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} \underbrace{\dot{q}_{[j]}}_{\dot{\underline{q}}_{[j]}} \\ & + \mathbf{C}_{\epsilon_{j-1}}^{\beta_{j-1}} \underline{t}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} \underbrace{\ddot{q}_{[j]}}_{\ddot{\underline{q}}_{[j]}} + \Big(\underbrace{[\underline{\alpha}_{W\beta_{j-1}}^{\beta_{j-1}} \times]}_{\underline{q}_{[:j-1]}, \dot{\underline{q}}_{[:j-1]}, \ddot{\underline{q}}_{[:j-1]}} + \underbrace{\mathbf{\Omega}_{W\beta_{j-1}}^{\beta_{j-1}} \mathbf{\Omega}_{W\beta_{j-1}}^{\beta_{j-1}}}_{\underline{q}_{[:j-1]}, \dot{\underline{q}}_{[:j-1]}} \Big) \\ & \left(\underline{x}_{\beta_{j-1}\epsilon_{j-1}}^{\beta_{j-1}} + \mathbf{C}_{\epsilon_{j-1}}^{\beta_{j-1}} \underline{t}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} \underbrace{q_{[j]}}_{\underline{q}_{[j]}} \right) + \underbrace{\underline{a}_{W\beta_{j-1}}^{\beta_{j-1}}}_{\underline{q}_{[:j-1]}, \dot{\underline{q}}_{[:j-1]}, \ddot{\underline{q}}_{[:j-1]}} \Bigg] \end{aligned} \tag{16}$$

is obtained. This expression can be differentiated recursively as demonstrated before. The occurring angular accelerations and their derivatives are calculated with the help of

$$\begin{aligned} \underline{\alpha}_{W\beta_j}^{\beta_j} = {} & \underbrace{\mathbf{C}_{\epsilon_{j-1}}^{\beta_j}}_{\underline{q}_{[j]}} \mathbf{C}_{\beta_{j-1}}^{\epsilon_{j-1}} \Big(\underbrace{\underline{\alpha}_{W\beta_{j-1}}^{\beta_{j-1}}}_{\underline{q}_{[:j-1]}, \dot{\underline{q}}_{[:j-1]}, \ddot{\underline{q}}_{[:j-1]}} + \mathbf{C}_{\epsilon_{j-1}}^{\beta_{j-1}} r_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} \underbrace{\ddot{q}_{[j]}}_{\ddot{\underline{q}}_{[j]}} \\ & + \underbrace{\mathbf{\Omega}_{W\beta_{j-1}}^{\beta_{j-1}}}_{\underline{q}_{[:j-1]}, \dot{\underline{q}}_{[:j-1]}} \mathbf{C}_{\epsilon_{j-1}}^{\beta_{j-1}} \underline{r}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} \underbrace{\dot{q}_{[j]}}_{\dot{\underline{q}}_{[j]}} \Big) , \end{aligned} \tag{17}$$

which in turn is obtained by differentiating (6) with respect to time and changing the resolving frame to β_j. Note, that the effect of gravity is already fully covered if $\underline{a}_{W\beta_0}^{\beta_0}$ is set appropriately. In the last step, the Jacobian of the accelerometer A_i mounted on link $k = p(A_i)$ is calculated using the partial derivatives of

$$\underline{a}_{WA_i}^{A_i} = \mathbf{C}_{\beta_k}^{A_i} \left(\underline{a}_{W\beta_k}^{\beta_k} + \left([\underline{\alpha}_{W\beta_k}^{\beta_k} \times] + \mathbf{\Omega}_{W\beta_k}^{\beta_k} \mathbf{\Omega}_{W\beta_k}^{\beta_k} \right) \underline{x}_{\beta_k A_i}^{\beta_k} \right), \tag{18}$$

which is a modified version of (16).

[2] https://isas.iar.kit.edu/media/pdf/Fennel_MFI22_SupplementalMaterial.pdf

D. Implementation

The resulting EKF and the presented algorithm for the efficient calculation of the measurement Jacobian were implemented in C++. To be robot-agnostic, the robot model can be configured at runtime by providing a URDF-file and a list of available sensors. The necessary forward kinematic calculations as well as the parsing of the URDF-file are carried out by the pinocchio-library [15].

V. Evaluation in Simulation

This section covers various simulation experiments to characterize the behavior of the proposed estimator.

A. Simulation Environment

For the following simulations, performed with a sampling frequency of 1 kHz, the manipulator as depicted in Fig. 2 with 2 prismatic joints, 6 revolute joints, and 8 IMUs was used. The corresponding Denavit–Hartenberg (DH) parameters are listed in Table I. For the test movements, all joints follow a pre-defined sinusoidal trajectory with a given frequency. To ensure a steady state at the beginning and the end as in real scenarios, the sinusoidal signal is modulated with a cosine window of length 10 s. The position amplitudes are chosen so that a maximum acceleration of 20 $\mathrm{m\,s^{-2}}$ or $\mathrm{rad\,s^{-2}}$ occurs, as long as the necessary position amplitude of 30° or 0.52 m is not exceeded. To cover different circumstances, 30 initial configurations with $|\underline{q}_{[i]}| < 0.52$ were drawn once from an uniform distribution. Furthermore, each initial configuration is associated with a set of random phase offsets for the sinusoidal signals.

For the simulation of the encoder data, the groundtruth position signals are distorted with zero-mean, white Gaussian noise with standard deviation 4.0×10^{-4} and discretized with resolution 1.2×10^{-5} (19-bit encoders). The simulation model of the gyroscope is depicted in Fig. 3. It comprises the scale-factor error and cross-axis sensitivity matrix $\mathbf{S}_{G_i}$, the additive Gaussian noise $\underline{v}_{G_i}$, a constant bias $\underline{b}_{G_i,\mathrm{const}}$, a temperature sensitive bias with coefficient $\underline{c}_{G_i}$, and a quantization. If not stated differently, the standard deviation of the noise is $0.32\,^\circ\,\mathrm{s^{-1}}$. All other parameters are drawn once from uniform random distributions, whose limits are derived from the typical values in the datasheet of an InvenSense MPU-9250 [16]. Additionally, a fixed pose calibration error is introduced for each gyroscope, for which translation errors are drawn uniformly from $[-2, 2]$ mm and Euler orientation errors are drawn uniformly from $[-2, 2]\,^\circ$ before the first simulation. The simulation model of the accelerometer follows the same scheme, whereas the noise standard deviation is $9.5 \times 10^{-3}\,\mathrm{m\,s^{-2}}$. For the temperature T, a deterministic sinusoidal trajectory with a frequency of 0.1 Hz and amplitude of 5 °C is assumed.

B. Estimator Performance

The proposed estimator was tuned for the presented simulation scenario. For this, the measurement noise covariance was set according to the sensor specifications. The process noise covariance was adjusted manually to

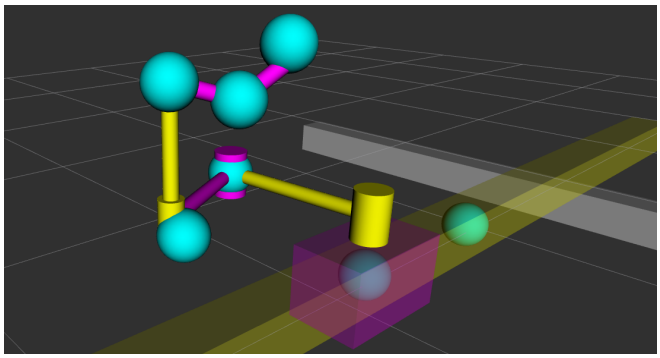

Fig. 2: An illustration of the manipulator used during simulation. The blue spheres mark the mounting points of the IMUs with triaxial gyroscopes and accelerometers at the end of each link. Note that the last sensor location is very close to the rotation axis of the last revolute joint.

Joint i	θ_i	d_i	a_i	α_i
1	0	$\underline{q}_{[1]}$	0	90°
2	90°	$\underline{q}_{[2]}$	0	90°
3	$\underline{q}_{[3]}$	0	0.7	0
4	$\underline{q}_{[4]} + 90°$	0	0.6	180°
5	$\underline{q}_{[5]}$	−0.5	0	90°
6	$\underline{q}_{[6]}$	0	−0.25	0
7	$\underline{q}_{[7]}$	0	0	90°
8	$\underline{q}_{[8]}$	0.25	0	0

TABLE I: DH parameters of the manipulator used during simulations.

$\mathrm{Cov}(\underline{w}_k) = \mathrm{diag}\big(\underline{0}_8^\mathsf{T}\ \underline{0}_8^\mathsf{T}\ \underline{0}_8^\mathsf{T}\ 12.5^2\underline{1}_8^\mathsf{T}\ 0.001^2\underline{1}_{24}^\mathsf{T}\ 0.01^2\underline{1}_{24}^\mathsf{T}\big)$. Here, the large jerk noise was chosen to facilitate rapid acceleration changes. As stated in Section IV-A, the bias states have to compensate for all inertial sensor errors. For this reason, the bias driving noise was selected much larger compared to the necessary noise for a *pure* sensor bias. Due to gravity, misalignments and scaling errors of the acceleration sensors are more severe for the measurement than for the gyroscopes. Therefore, $\sigma_{b_a} = 10\,\sigma_{b_g}$ was selected. The KF is initialized during standstill. This means the kinematic part of the state is accurately known at the beginning. The initially unknown bias states are zero-initialized with a high variance.

Fig. 4 shows the resulting RMS errors (RMSE) of the kinematic states for varying frequencies. In general, the position estimates have very low errors, that are even below the encoder noise for most of the considered frequencies. For very high frequencies, the estimation errors become larger than the RMS value of the excitation. However, this is only a minor limitation, as typical excitation amplitudes for high-frequency vibrations are very small. The velocity estimation is also performing well, especially in the practical

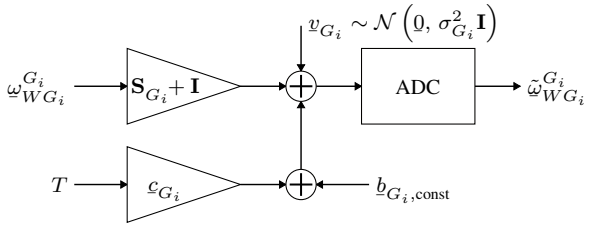

Fig. 3: Sensor model of the gyroscopes used during simulations.

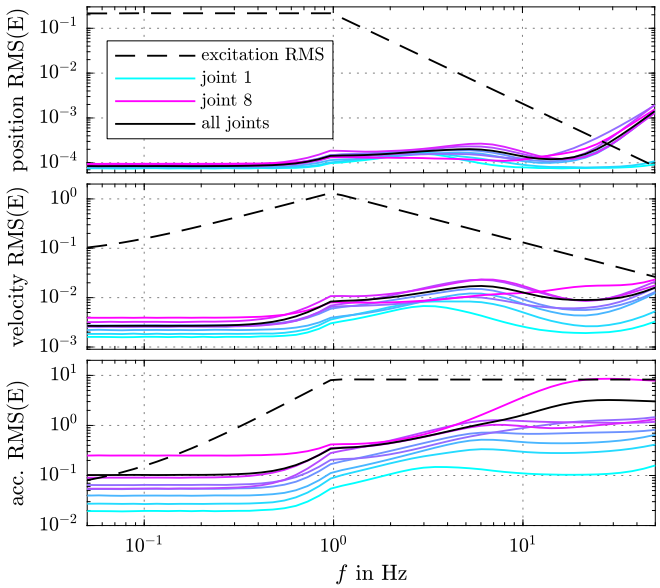

Fig. 4: RMSEs of the proposed estimator. The dashed lines represent the RMS values of the excitation signals.

relevant range between 0.5 Hz and 5 Hz, where the error is less than $1/20$ of the excitation. It should also be noted that the RMSE is in the range of the raw gyroscope noise standard deviation for very low frequencies. Similar to the velocity, the RMSE of the acceleration is mostly $1/10$ of the excitation RMS in the frequency range between 0.5 Hz and 5 Hz. For higher frequencies, the estimator performs worse, especially regarding the last joint. The reason for this is the reduced observability of the last joint's acceleration because the responsible acceleration sensor is mounted very close (50 mm) to the rotation axis and no further acceleration sensors are observing the last joint. For the same reason, the combined acceleration RMSE for all joints looks unsatisfying at the first glance for high frequencies. Nevertheless, this quantity facilitates a fast comparison of different estimators and is therefore used hereinafter.

In the simulations, a full iteration of the estimator took 560 µs on average on a standard laptop with an Intel Core i7-9750H CPU. From this, we can conclude that our estimator is real-time capable for manipulators with ≤ 8 joints and control frequencies ≤ 1 kHz. Furthermore, the evaluation of the Jacobian required 6 µs on average, proving that our proposed recursive algorithm from Section IV-C is highly efficient.

C. Influence of Environmental Variations

To test the influence of different environmental effects and increasing errors, the RMSEs over frequency were determined in the following setups:

1) Normal noise, but ideal sensor calibration and no bias.
2) Like 1, but with sensor bias terms.
3) Like 2, but with a scale factor error.
4) Reference setup from Section V-A with all errors.
5) Like 4, but with 5-fold increased inertial sensor noise.
6) Like 4, but only with sensors at every other link.

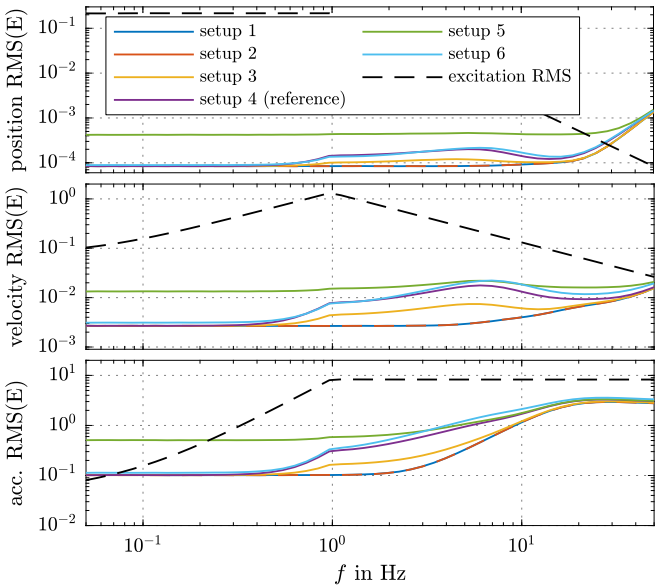

Fig. 5: RMSEs of the proposed estimator under different conditions. The dashed lines represent the RMS values of the excitation signals.

Accordingly, the curves of setup 1 in Fig. 5 represent a perfect sensor model and thus a lower bound for the estimation error with the given system model. The curves for setup 2 are coincident with setup 1, indicating that constant and slowly varying sensor biases are handled as expected. If scale factor errors are added as in setup 3, the estimation error of velocity and acceleration increases approximately by a factor of two in the range between 1 Hz and 10 Hz. A further doubling of these errors occurs when rotation and translation errors as in setup 4 are included in the simulation, yielding the previously discussed data from Fig. 4. Consequently, we deduce that the calibration of the inertial sensors impacts the overall system performance. Especially, the knowledge of scale factor errors, cross-axis sensitivity, and alignment errors has the potential to improve the estimation quality significantly. Nevertheless, we want to emphasize that the estimation also works without calibration as claimed in the introduction.

In real scenarios, the sensor noise might be increased. The resulting curves from setup 5 in Fig. 5 suggest that the proposed estimator is robust against such changes, as long as a proportional increase of velocity and acceleration errors for frequencies < 1 Hz is tolerated. Higher frequencies are less affected by the increased sensor noise.

With the last setup, the claimed modularity regarding the sensor configuration was tested. As one can see from setups 4 and 6 in Fig. 4, the missing sensors do not significantly diminish the estimation quality. In practice, however, the number of sensors cannot be reduced arbitrarily without performance losses because the full observability based on acceleration sensors will get lost at some point.

D. Comparison with other Estimators

In this section, a comparison of the proposed **f**ull **K**alman **F**ilter (*KF-F*) with four kinematic state estimators from the literature is carried out in simulation.

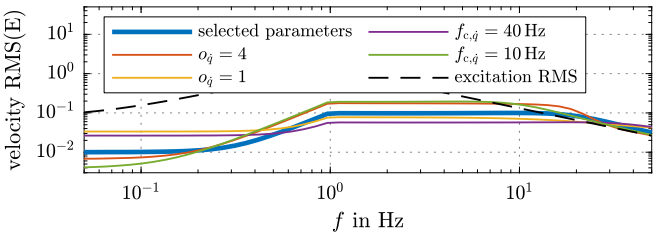

Fig. 6: RMSEs of a Butterworth low pass filter for velocity estimation with different parameters. None of the configurations clearly outperforms the selected parameters.

1) Competitors: The following competitors were implemented in C++:

- *ND*: The first competitor calculates the **n**umerical **d**erivatives of the joint positions. The noisy velocity is then smoothed with a Butterworth filter of order $o_{\dot{q}}$ and cutoff frequency $f_{c,\dot{q}}$. Analogously, the acceleration is smoothed with the parameters $o_{\ddot{q}}$ and $f_{c,\ddot{q}}$.
- *PI*: In the second approach following [12], biased joint accelerations are calculated from accelerometer measurements based on a **p**seudo**i**nverse. At the same time, unbiased but delayed estimates are calculated using the previous method. In combination with the biased joint accelerations, which are artificially delayed by t_{delay}, and a Butterworth filter with parameters o_b and $f_{c,b}$, a bias is calculated to correct the current biased joint accelerations. Lastly, another Butterworth filter with parameters o_a and $f_{c,a}$ smoothes the final acceleration estimate.
- *KF-D*: The third competitor uses several **d**ecoupled **K**alman **F**ilters as presented in [11]. In contrast to our model, biases are completely neglected and each sensor is used exclusively for updating one joint.
- *KF-T*: The last competitor is the **t**rivial **K**alman **F**ilter which only processes encoder data with a constant jerk model. This means that no inertial data is incorporated into the estimate.

For KF-D and KF-T, appropriate parameters were directly derived from the parameters in Section V-B, as both can be considered as a functional subset of our filter. The parameters of ND were tuned manually to $o_{\dot{q}} = 2$, $o_{\ddot{q}} = 4$, and $f_{c,\dot{q}} = f_{c,\ddot{q}} = 20$ Hz. Fig. 6 demonstrates that the selected parameters for the velocity filter are a reasonable choice because neither a variation of the order nor of the cutoff frequency yield a global improvement. Unfortunately, no parameters are stated in the original publication of PI. For this reason, the filter parameters for the delayed velocity and acceleration estimates were adopted from ND. The remaining parameters were tuned manually to $t_{\text{delay}} = 22$ ms, $o_b = 2$, $o_a = 1$, $f_{c,b} = 5$ Hz, and $f_{c,a} = 200$ Hz, which is a good tradeoff as seen in Fig. 7.

2) Results: With Fig. 7, it is evident that the proposed estimator, KF-F, outperforms all considered approaches in a broad frequency range. Especially for the velocity and accelerations, a significant decrease of the estimation error can be expected confirming the effectiveness of our method. The plot also shows that KF-D performs much worse than KF-T or ND, which is caused by its sensitivity

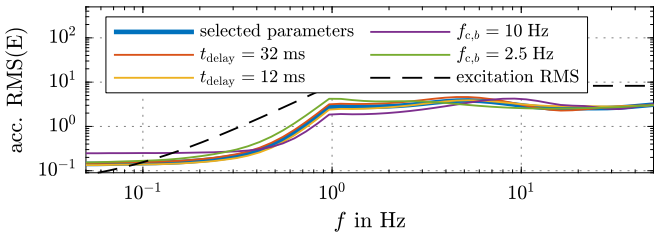

Fig. 7: RMSEs of PI for acceleration estimation with different parameters. The filter orders have similar effects as in Fig. 6. The parameter $f_{\mathrm{c},a}$ was omitted as it had very little effect.

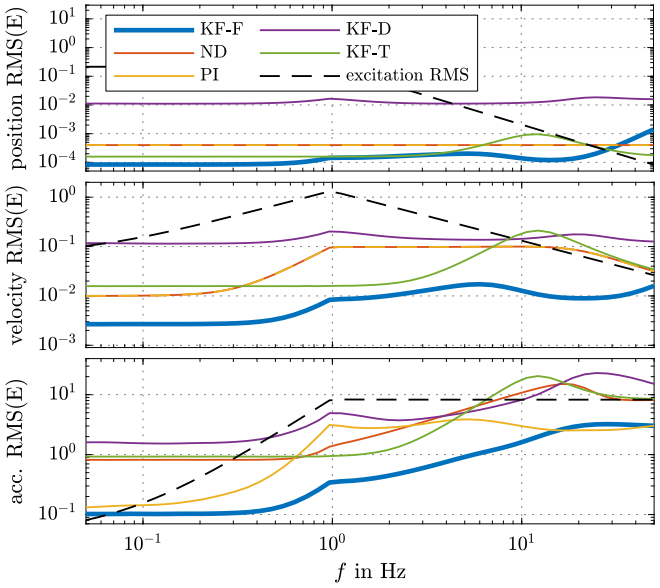

Fig. 8: Estimation error of the compared kinematic state estimators.

towards inaccurate calibration data. The acceleration estimates of PI are much less precise than those of KF-F in the medium frequency range, which is mostly owed to the bad observability of the last joint as the joint-wise RMS errors $\mathrm{RMSE}_{\mathrm{PI}}^{\mathsf{T}} = (0.14, 0.19, 0.42, 0.65, 0.91, 0.92, 0.76, 6.41)$ and $\mathrm{RMSE}_{\text{KF-F}}^{\mathsf{T}} = (0.10, 0.17, 0.21, 0.30, 0.26, 0.49, 0.44, 0.48)$ for 2 Hz suggest. A potential drawback of our method is the increased computational effort, as shown in Table II.

VI. Evaluation on Real Hardware

The practical applicability of the proposed estimator was demonstrated on real hardware.

A. Setup

The test setup is depicted in Fig. 9. A two joint SCARA manipulator is equipped with a 19-bit load-side encoder in every joint and an uncalibrated InvenSense MPU-9250 at the end effector. All calculations are carried out with a frequency of 1 kHz. The manipulator is programmed to perform a linear back-and-forth movement in Cartesian space with a maximum velocity of $0.5\,\mathrm{m\,s^{-1}}$ and a distance of $0.5\,\mathrm{m}$ between the reversal points. After 11 cycles in 25 s, the motion is stopped abruptly by triggering the mechanical brakes of the joints. The encoder noise was identified to be less serious than assumed in the simulations, while the IMU errors were increased

Estimator	Mean runtime in µs
KF-F	560
ND	< 1
PI	6
KF-D	113
KF-T	24

TABLE II: Processing times of the compared kinematic state estimators.

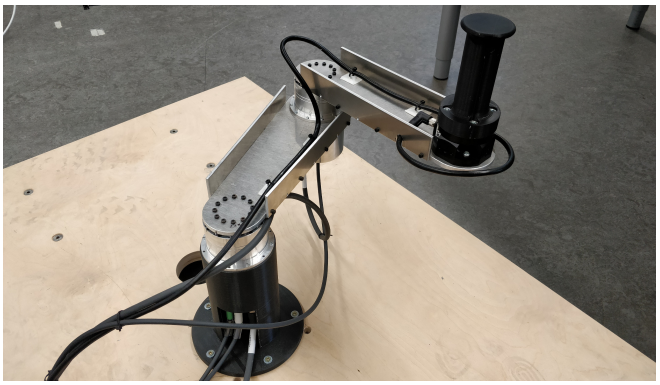

Fig. 9: A planar manipulator with 2 joints in SCARA configuration was used for the evaluation on real hardware.

significantly. Hence, the filter has been retuned with the parameters $\mathrm{Cov}(\underline{w}_k) = \mathrm{diag}\big(\underline{0}_6^{\mathsf{T}} \;\; 12.5^2\underline{1}_2^{\mathsf{T}} \;\; 0.005^2\underline{1}_3^{\mathsf{T}} \;\; 0.05^2\underline{1}_3^{\mathsf{T}}\big)$ and $\mathrm{Cov}(\underline{v}) = \mathrm{diag}\Big(\big(6{\times}10^{-5}\big)^2\underline{1}_2^{\mathsf{T}} \;\; 0.055^2\underline{1}_3^{\mathsf{T}} \;\; 0.188^2\underline{1}_3^{\mathsf{T}}\Big)$.

B. Results

Fig. 10 shows the velocity and acceleration estimates of our approach (KF-F) for both joints. The oscillations are not noise, but true mechanical oscillations originating from finite stiffness as reported in [12]. Before the estimation quality can be assessed, the reference signals must be discussed due to the lack of real groundtruth information in our setup.

It is obvious that numerical differentiation combined with an *acausal* smoothing step during post-processing can yield noise-free and undelayed estimates. However, the resulting signals are only reliable up to a specific frequency as the amplification of the encoder noise increases with frequency. Furthermore, the encoders have additional errors beyond additive noise and the manipulator's structural elements are not infinitely stiff, resulting in high-frequent movements not captured by the encoders. For this reason, a fair comparison without dedicated acceleration and velocity groundtruth sensors is limited to a comparatively low frequency range.

In our case, an analysis of the raw IMU data and the differentiated, but unfiltered encoder data suggests that a frequency limitation to 10 Hz is reasonable. This was implemented using a 4th-order Butterworth filter in a forward-backward setup to achieve delay-free results. The resulting low-pass filtered numerical derivatives (reference) and the low-pass filtered estimates of our approach (KF-F, band-limited) are included in Fig. 10. Since both are very close, the deviations, which can be interpreted as an estimation error, are plotted separately in Fig. 11. For the whole trajectory with an RMS of 1.253 for velocity and 5.727 for acceleration, the corresponding RMSEs are 0.005 and 0.430, respectively.

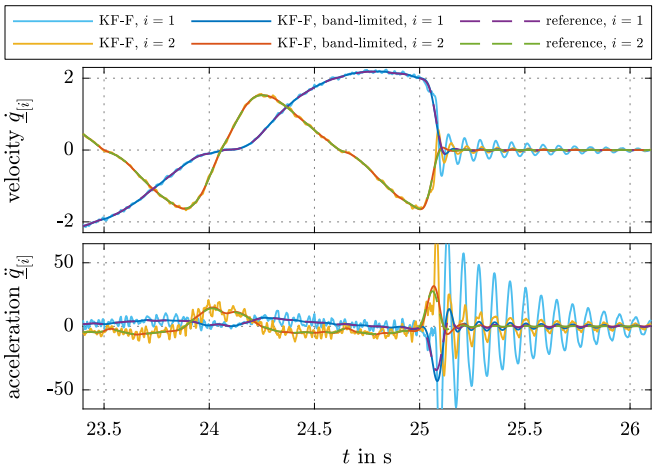

Fig. 10: Excerpt of the velocity and acceleration estimates of the proposed estimator over time in comparison with the band-limited numerical differentiation. The brake is actuated at $t \approx 25\,\mathrm{s}$.

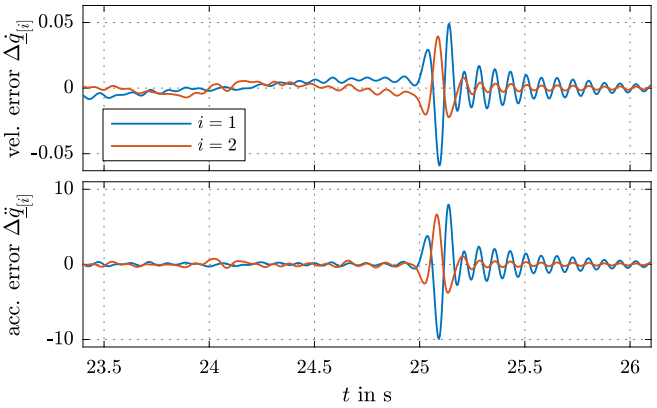

Fig. 11: Deviations between the proposed estimator and the reference from numerical differentiation over time, when a band-limit of 10 Hz is applied. The estimation errors are increased significantly when the brakes are triggered. Nevertheless, the error magnitude is kept at a fraction of the actual signal magnitude.

From this, it is evident that the proposed estimator also works well in real-life scenarios, even in situations where very abrupt acceleration and velocity changes occur.

VII. Conclusions

In this paper, a new calibration-free kinematic state estimator for robot manipulators with revolute and prismatic joints was presented. To achieve this, an EKF combines a constant jerk model for the movement with constant bias models for the inertial sensors. Although the resulting system model has a comparatively large state dimension, real-time capable and robot-agnostic execution is made possible by means of an efficient recursive expression for the Jacobian of the measurement function. As demonstrated in simulations and experiments with real hardware, the proposed estimator is able to incorporate nearly all information from a given sensor setup, while still being robust towards parameter changes. We outperformed competing approaches significantly in all aspects except for runtime.

Future work will involve runtime optimizations exploiting the sparsity of the system and measurement Jacobians during the prediction and the update step of the EKF. Thereafter, the application of the proposed estimator on more complex real robots is planned. This also raises the question of which sensor setups are optimal regarding the number of sensors and the observability. Furthermore, the generalization to setups where the joint position cannot be measured directly with sufficient precision, i.e., human motion tracking, is considered for future research.

References

[1] S. Haddadin, A. De Luca, and A. Albu-Schäffer, "Robot collisions: A survey on detection, isolation, and identification," *IEEE Transactions on Robotics*, vol. 33, no. 6, pp. 1292–1312, Dec. 2017.

[2] Q. Leboutet, J. Roux, A. Janot, J. R. Guadarrama-Olvera, and G. Cheng, "Inertial parameter identification in robotics: A survey," *Applied Sciences*, vol. 11, no. 9, May 2021.

[3] W. L. Xu and J. D. Han, "Joint cceleration feedback control for robots: analysis, sensing and experiments," *Robotics and Computer-Integrated Manufacturing*, vol. 16, no. 5, pp. 307–320, Oct. 2000.

[4] A. Calanca and P. Fiorini, "A rationale for acceleration feedback in force control of series elastic actuators," *IEEE Transactions on Robotics*, vol. 34, no. 1, pp. 48–61, Feb. 2018.

[5] W.-H. Zhu and T. Lamarche, "Velocity estimation by using position and acceleration sensors," *IEEE Transactions on Industrial Electronics*, vol. 54, no. 5, pp. 2706–2715, Oct. 2007.

[6] E. Hedberg, J. Norén, M. Norrlöf, and S. Gunnarsson, "Industrial robot tool position estimation using inertial measurements in a complementary filter and an EKF," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 12 748–12 752, July 2017.

[7] W. Chen and M. Tomizuka, "Direct joint space state estimation in robots with multiple elastic joints," *IEEE/ASME Transactions on Mechatronics*, vol. 19, no. 2, pp. 697–706, Apr. 2014.

[8] J. Vihonen, J. Honkakorpi, J. Mattila, and A. Visa, "Geometry-aided angular acceleration sensing of rigid multi-body manipulator using MEMS rate gyros and linear accelerometers," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nov. 2013, pp. 2514–2520.

[9] N. Rotella, S. Mason, S. Schaal, and L. Righetti, "Inertial sensor-based humanoid joint state estimation," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, pp. 1825–1831.

[10] 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.

[11] ——, "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, Apr. 2020.

[12] Y. Zimmermann, E. B. Küçüktabak, F. Farshidian, R. Riener, and M. Hutter, "Towards dynamic transparency: Robust interaction force tracking using multi-sensory control on an arm exoskeleton," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2020, pp. 7417–7424.

[13] P. D. Groves, *Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems*, 2nd ed. Boston, MA: Artech House, 2013.

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

[15] J. Carpentier, G. Saurel, G. Buondonno, J. Mirabel, F. Lamiraux, O. Stasse, and N. Mansard, "The pinocchio C++ library : A fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives," in *2019 IEEE/SICE International Symposium on System Integration (SII)*, Jan. 2019, pp. 614–619.

[16] InvenSense, *MPU-9250 Product Specification*, June 2016, rev. 1.1. [Online]. Available: https://invensense.tdk.com/wp-content/uploads/2015/02/PS-MPU-9250A-01-v1.1.pdf