IEEE CSS

Received XX September 2022; revised ; accepted XX October 2022; Date of publication XX November 2022; date of current version XX November 2022. This paper was recommended by Associate Editor F. A. Author.

Digital Object Identifier 10.1109/OJCSYS.2021.Doi Number

Velocity Estimation of Robot Manipulators: An Experimental Comparison

S. B. LIU[1] (Grad. Stud. Member, IEEE), A. GIUSTI[2] (Member, IEEE), M. ALTHOFF[1] (Member, IEEE)

[1]Department of Informatics, Technical University of Munich, Garching, 85748, Germany
[2]Fraunhofer Italia Research, Bolzano, 39100, Italy

CORRESPONDING AUTHOR: S. B. Liu (e-mail: stefan.liu@tum.de)

This work was supported by the European Union's Horizon 2020 Research and Innovation Program under Grant Agreement 101016007 (Project CONCERT).

ABSTRACT Accurate velocity information is often essential to the control of robot manipulators, especially for precise tracking of fast trajectories. However, joint velocities are rarely directly measured and instead estimated to save costs. While many approaches have been proposed for the velocity estimation of robot joints, no comprehensive experimental evaluation exists, making it difficult to choose the appropriate method. This paper compares multiple estimation methods running on a six degrees-of-freedom manipulator. We evaluate: 1) the estimation error using a ground-truth signal, 2) the closed-loop tracking error, 3) convergence behavior, 4) sensor fault tolerance, 5) implementation and tuning effort. To ensure a fair comparison, we optimally tune the estimators using a genetic algorithm. All estimation methods have a similar estimation error and similar closed-loop tracking performance, except for the nonlinear high-gain observer, which is not accurate enough. Sliding-mode observers can provide a precise velocity estimation despite sensor faults.

INDEX TERMS Velocity estimation, robots, manipulators, genetic algorithms, tuning.

I. INTRODUCTION

Accurate joint velocity signals of robot manipulators are needed for many fundamental control purposes, e.g., trajectory tracking, collision detection, and force control [1]. Sensors for measuring joint positions, e.g., encoders, have become inexpensive, reliable, and have a high resolution. The same cannot be said for velocity measurements. Direct measurements, e.g., through magnetic tachometers are affected by discontinuities of the magnetic field, ripple torques, and other high-frequency noise [2], while encoders are much more robust. Compactness and economic reasons often lead to not integrating joint velocity sensors at all.

Starting with the works of Nicosia and Tomei [3] in the 1990s, velocity estimation for robots has been discussed widely in the literature, and many different methods have been proposed since then. From a practitioner's point of view, however, it is still hard to select a proper estimation method, because 1) it is hard to infer differences between estimation methods from previous papers, 2) many techniques have only been evaluated in simulation, and 3) the evaluations have been carried out on different robots.

Our paper addresses this issue by systematically comparing popular velocity estimation concepts and evaluating them using criteria which are important to practitioners, such as tuning and robustness to faults. Together with this paper, we also publish a MATLAB tool package, that includes an implementation of all discussed methods ready-to-use.

Previous studies that involve comparing velocity estimation methods can be found in [4]–[14]. Simulative comparisons of derivative filters for discrete position measurements showed that no approach works best for all velocity profiles [4]–[6]. In the simulative comparison in [7], an extended Kalman filter, a nonlinear high-gain observer, and a linear observer have been compared considering their position estimation error and tracking error on a two degrees-of-freedom (DOF) robot. The authors in [8], [9] experimentally

compare the tracking error of different tracking controllers using linear high-gain observers. The study in [10] experimentally analyzes the tracking error of a 2-DOF planar robot using five different observers. However, each observer uses a different tracking controller and only trapezoidal trajectories were tested. An experimental comparison between a nonlinear high-gain observer, a Kalman filter, and a lead-lag based filter has been conducted on a parallel kinematics robot in [11], where the authors use a dual-mode controller and a proportional-derivative (PD) controller. The comparisons in [10] and [11] have two main drawbacks: 1) the gains of the proposed methods were chosen without apparent justification, although the performance of velocity estimation mainly depends on such gains, and 2) these comparisons did not evaluate the performance in terms of velocity estimation error. Our paper addresses these issues by including a gain tuning approach that allows a comparison of optimally tuned estimators, while the velocity estimation error is measured through an actual ground-truth signal. Further works that compare model-free and model-based observers, used in conjunction with different controller structures, can be found in [12]–[14].

In contrast to previous works, our paper presents for the first time an experimental comparison of a wide variety of estimation methods, including multiple filters and multiple observers. We use the same 6-DOF robot manipulator, the same tracking controller, and the same test trajectory for all estimators.

This paper is organized as follows: in Sec. I, we state the problem at hand and survey the literature for popular existing velocity estimators. In Sec. II, we provide an in-depth review of selected estimators, which we consider to be among the most suitable for practical applications. The automatic parameter tuning is explained in Sec. III. The evaluated estimators are compared experimentally in Sec. IV and we conclude the paper in Sec. V.

A. PROBLEM STATEMENT

Let us consider the rigid dynamics of robot manipulators with n revolute joints written in state-space form as

$$\begin{aligned} \dot{x}_1 &= x_2, \\ \dot{x}_2 &= f(x_1, x_2, u) = M^{-1}(x_1)(u - n(x_1, x_2)), \end{aligned} \tag{1}$$

where $x_1 \in \mathbb{R}^n$ is the vector of joint positions, $x_2 \in \mathbb{R}^n$ is the vector of joint velocities, $u \in \mathbb{R}^n$ is the vector of motor torques, $M(x_1) \in \mathbb{R}^{n\times n}$ is the inertia matrix, and $n(x_1, x_2) \in \mathbb{R}^n$ is the vector-valued function including Coriolis and centripetal forces, gravity, and friction. The joint positions are measured at a finite resolution using rotational encoders.

The robot tracks a desired trajectory of positions, velocities, and accelerations $x_1^d(t), x_2^d(t), x_3^d(t) \in \mathbb{R}^n$ via an inverse dynamics controller [15, Sec. 8.5.2]

$$u = M(\hat{x}_1)\nu + n(\hat{x}_1, \hat{x}_2), \tag{2}$$

$$\nu = x_3^d + K_p(\hat{x}_1 - x_1^d) + K_d(\hat{x}_2 - x_2^d), \tag{3}$$

where $\hat{x}_1, \hat{x}_2$ are the vectors of estimated joint positions and velocities. Some of the discussed methods do not estimate $\hat{x}_1$; but since x_1 is measured directly, $\hat{x}_1$ can be replaced by x_1 in (2) and (3), when applicable.

The objective of this paper is to compare different methods to obtain $\hat{x}_2$ and to tune the parameters of all estimators, such that the error $x_2 - \hat{x}_2$ between the estimated and the ground-truth velocity is minimized. To obtain a ground-truth signal, any method can be used that returns a significantly more accurate velocity than the evaluated estimations, e.g., using external encoders with a higher resolution and sampling rate. In our paper, we simulate external measurement by artificially decreasing the sensor resolution and sampling rate of the internal sensors for closed-loop control, while the ground truth is obtained using the actual sensor resolution at a higher sampling rate.

The estimation methods are subject to disturbances in our robot system. Amongst others, there can be

- *quantization errors* due to the finite resolution of the encoders;
- *high-frequency noises* due to manufacturing errors of the encoders [16];
- *modeling errors* due to an inaccurate parametrization of $f(x_1, x_2, u)$;
- *sensor faults* due to communication errors.

In the subsequent literature survey we group the approaches we identified for velocity estimation into *model-based* approaches, that require the computation of the nonlinear dynamical model in (1), and *model-free* approaches, which do not need this model. Model-free methods can be implemented decentrally at each individual joint, if the methods do not have dependencies between joints. Model-based methods, however, must be implemented in a centralized manner. The considered approaches are collected in Table 1, which also sorts them according to the fact that they are validated in the literature using simulations or experiments.

B. MODEL-BASED METHODS

We first survey model-based schemes. The popular and pioneering model-based method of Nicosia and Tomei in [3] presents an asymptotically stable observer whose region of attraction can be enlarged via the observer gain. In contrast to previous work, the authors design the model-based observer in conjunction with a controller; many subsequent works followed this idea. The authors in [17] propose a model-based observer which provides semi-global exponentially stable error dynamics of the velocity tracking error, considering a dedicated controller structure. Effectiveness of this approach is shown by experiments on a 2-DOF manipulator. Another model-based extension of the approach from Nicosia and Tomei can be found in [18], in which the authors show semi-global exponential stability of their proposed combined observer and controller. The authors in [19] and [20] discuss nonlinear high-gain observers for the velocity estimation problem, including how to avoid the peaking phenomenon

in the transient behavior, i.e., the initial estimation error may exhibit an impulse that could destabilize the controller. An adaptive approach providing locally asymptotically stable estimation error dynamics has been proposed in [21] in which the authors show that their proposed approach is superior to simple numerical differentiation; the authors perform experiments on a 6-DOF PUMA-560 robot. The work in [22] introduces a combined observer/controller structure providing global exponential convergence of the estimation error. However, that paper only shows practical effectiveness by means of simulations. More recently, the author in [23] showed theoretically that a proposed Luenberger-like observer with a simple proportional-derivative control with gravity compensation achieves uniformly ultimately bounded stability, which is confirmed by experiments on a 2-DOF robot. Model-based approaches using sliding mode observers have been proposed for robots in [2], [24], [25], whose effectiveness was only demonstrated by simulations.

C. MODEL-FREE METHODS

In this subsection, we survey model-free approaches. In the works of Nicosia et. al. [8], a simple high-gain observer is introduced, which supports distributed implementations; this approach also provides uniformly ultimate boundedness of the velocity estimate and is presented and tested using both simulations and experiments on a 6-DOF robot. The work in [26] introduces a model-free observer providing uniformly ultimate boundedness of the velocity estimation error. This scheme accounts for model uncertainties in its design and its effectiveness have been verified by means of experiments with a 2-DOF robot. Both [8] and [26] consider inverse dynamics control and proportional-derivative control for tracking, provide a closed-loop stability analysis for both cases, and suggest parameters to ease gain tuning of the observer. Subsequently, the authors in [27] also proposed a model-free observer that provides uniformly ultimate boundedness of both tracking and observer errors when used in conjunction with their proposed robust controller. The authors in [28], [29] introduce a model-free observer which provides asymptotic stability of the velocity estimation error dynamics. To achieve this, [28] uses passivity arguments, while more general Lyapunov arguments are used in [29], where also external disturbance is taken into account and the performance is shown using simulations on a 2-DOF robot. Similarly, the works in [30], [31] present model-free observers that provide asymptotic stability demonstrated in simulation [30] and experiments [31]. A model-free sliding-mode observer has been proposed in [32]; its practical effectiveness has been presented by simulations. As an extension, some estimators incorporate neural networks [33], [34].

Furthermore, there exist popular estimators without explicit closed-loop stability proofs. Kalman filters [35] are such an example, which assume white noise to approximate the robot dynamics. Also, the derivative filtering methods, such as the ones in [5], are not yet proven to be stable in closed-loop. However, the author of [36] introduces a possible theoretical framework to foster the use of derivative filtering in place of state observers for a stable output-feedback control of robots.

TABLE 1. Velocity estimation methods identified in this survey (references with * are evaluated in our comparison).

	Validated in simulation	Validated in experiments
model-free	[5]*, [12]–[14], [28]–[30], [32]–[34], [37]*	[8]*, [26], [27], [31]
model-based	[2], [12]–[14], [18], [19], [20]*, [22], [24], [25]*	[17], [21], [23]

From the available literature, we select several estimation methods, of which we conduct an in-depth review, which can be divided into four model-free methods from the works in [5], [8], [37], and two model-based methods from the works in [20], [25]. The selected methods have an asterisk in Tab. 1. These have been mainly selected for their popularity, ease of implementation, ease of tuning, and their robustness with respect to the chosen controller.

II. REVIEW OF SELECTED ESTIMATORS

In this section, we discuss the estimators that we experimentally compare, namely moving average filtering [5], derivative filtering [5], Kalman filtering [37], linear high-gain observer [8], nonlinear high-gain observer [19], and sliding-mode observer [25]. We review their respective properties as studied in the literature. Furthermore, we discuss the implementation aspects.

A. FINITE DIFFERENCE AND MOVING AVERAGE FILTERING

This basic technique numerically approximates the derivative by dividing the difference between successively obtained position measurements by a time window $p\Delta t$, where Δt is the sampling time of the controller and p is an integer that determines the size of the window for which we take the average. We denote a position measurement as $x_{1,k-1} = x_1((k-1)\Delta t)$. The estimated velocity is given by

$$\hat{x}_{2,k} = \frac{x_{1,k} - x_{1,k-p}}{p\Delta t}. \tag{4}$$

With large p, the averaging effect attenuates quantization noise in the measurements, but introduces a delay in the estimated velocity, while small p values amplify the noise [37]. For our comparison, we use $p = 1$, which we also call the finite difference (*FinDiff*) method, and an optimally chosen $p > 1$, which we call the moving average (*MovAv*) method.

B. DERIVATIVE FILTERING

Here, we describe a class of methods that compute the derivative through filtering the position signal. Various predictive strategies have been proposed in the literature based on a polynomial fitting of previous measurements, such as Taylor series expansion (TSE), and backward difference

expansion (BDE), which are characterized by the number of samples n_{TSE} and n_{BDE}. To counter the problem of overfitting and the resulting noise amplification, the least-squares fit (LSF) has been proposed in [5], that uses regression to find a polynomial of the order p_{LSF} with the smallest error among n_{LSF} measurements, where $p_{\mathrm{LSF}} < n_{\mathrm{LSF}}$. In-depth comparisons of these methods and a description of their implementation can be found in [4], [5], [38]. The findings of Brown et al. [5] are that TSE and BDE are good for transient responses and LSF filters are more suited for constant velocities. For velocity profiles that vary a lot, such as for robot manipulators, no single filtering method is best [38]. In our comparison, we will first evaluate the TSE, BDE, and LSF filters amongst each other, and choose the best one for the overall comparison with other methods.

C. KALMAN FILTER

The Kalman filter is a linear observer, that has been used in many engineering fields, such as for state and parameter estimation, data merging, or signal processing [39]. Bélanger proposes such an observer for rotary encoders [37], and instead of using the dynamical model in (1), assumes a triple integrator model for each individual axis i, consisting of the state $z_i = [x_{1,i}, x_{2,i}, x_{3,i}] \in \mathbb{R}^3$ (position, velocity, and acceleration of each axis), the output y_i, and the Gaussian white noises v_i (sensor noise) and w_i (process noise) [37, Eq. 14]:

$$\begin{aligned}
\dot{z}_i &= Az_i + \Gamma w_i \\
y_i &= Cz_i + v_i, \\
A &= \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}, \Gamma = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \\
C &= \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}.
\end{aligned}$$

According to [37], v_i can be chosen to be zero-mean with variance $\frac{\Delta q_{m,i}^2}{3}$, where $\Delta q_{m,i}$ is the quantization error of each axis. The noise w_i is also assumed to be zero-mean and has a variance Q_i, which has to be tuned for each axis. In the same work, a second-order system is additionally proposed for velocity estimation which, however, does not perform as well as the third-order one. The analysis in [37] showed an improvement compared to the finite difference method, especially at low speeds up to one tenth of an encoder increment per time step.

To further improve the acceleration estimation [7] or to provide estimations for flexible robots [40], one could estimate the state of the full model of the robot considering the nonlinear dependencies between joints. For those systems, extended Kalman filters are required due to the nonlinearity of the system. Since both cases are not relevant in our application, we deliberately exclude this method in our comparison.

D. LINEAR HIGH-GAIN OBSERVER

High-gain observers are theoretically well understood (see, e.g., the works of Khalil [20]) and have been experimentally examined, e.g., in [7]–[11]. In this work, we discuss both the linear and the nonlinear versions. The linear observer (*linHG*) uses a scalar gain ϵ_l and two matrix gains $H_1, H_2 \in \mathbb{R}^{n\times n}$ [8, Eq. 9]:

$$\begin{aligned}
\dot{\hat{x}}_1 &= \hat{x}_2 + \frac{1}{\epsilon_l} H_1 (x_1 - \hat{x}_1), \\
\dot{\hat{x}}_2 &= \frac{1}{\epsilon_l^2} H_2 (x_1 - \hat{x}_1).
\end{aligned}$$

This observer is asymptotically stable if the eigenvalues of $\begin{bmatrix} -H_1 & I \\ -H_2 & 0 \end{bmatrix}$ have negative real parts [20]. It has been shown in [8], that there exists an ϵ_l^* so that the closed-loop dynamics is asymptotically stable for $\epsilon_l \in [0, \epsilon_l^*]$, for any uniformly asymptotically stable controller. In other words, high-gain observers can be flexibly combined with any tracking controller, while overall stability is guaranteed.

In practice, however, the observer gains are limited, i.e., ϵ_l is lower-bounded by measurement noise and the sampling time of the controller [20]. Therefore, a trade-off between the noise suppression and estimation accuracy has to be found. To partially overcome this compromise, one can filter measurements and implement time-varying gains, as discussed in [20]; this extension is excluded in our comparison since we limit ourselves to easily implementable approaches. Also, a peaking phenomenon occurs (not examined by [8]). For these cases, an input saturation is sufficient for stability [20].

E. NONLINEAR HIGH-GAIN OBSERVER

Considering the full robot model (1) as an additional source of information, there exists a potential for better results using nonlinear observers. The nonlinear high-gain observer (*nnlHG*) is such a model-based approach, that is similar to its above-discussed linear version. For robots, this observer has first been introduced by Lee and Khalil in [19]. The observer uses the scalar gain ϵ_n and two matrix gains $L_1, L_2 \in \mathbb{R}^{n\times n}$ [20, Eq. 9.4]:

$$\begin{aligned}
\dot{\hat{x}}_1 &= \hat{x}_2 + \frac{1}{\epsilon_n} L_1 (x_1 - \hat{x}_1), \\
\dot{\hat{x}}_2 &= f(x_1, \hat{x}_2, u) + \frac{1}{\epsilon_n^2} L_2 (x_1 - \hat{x_1}).
\end{aligned}$$

Similar to the linear version, asymptotic stability is given if the real part of the eigenvalues of $\begin{bmatrix} -L_1 & I \\ -L_2 & 0 \end{bmatrix}$ are negative and a nonlinear separation principle can be established for the stability of the closed-loop system [41], meaning that also this observer can be flexibly combined with any stable tracking controller. A simulation study in [20] has shown that, indeed, a better velocity estimation compared to the linear version can be achieved, if the model is precise. However, this advantage becomes less and less significant, when the gains ϵ_l and ϵ_n decrease [20].

F. SLIDING MODE OBSERVER

Robustness, finite-time convergence, and the ability to handle discontinuous systems are major reasons for the application

of sliding mode observers (*SliMod*) [42]. Real implementations of sliding mode observers, however, suffer from chattering while sliding along the switching surfaces. This effect can be alleviated by second or [42] or third-order [43] sliding-mode observers. Third-order versions experience a slower convergence than second-order observers, as shown by Fraguela Cuesta *et al.* in [43]. The version we consider in our comparison is the third-order version proposed in [25], which adds a linear term to improve convergence. It consists of the gain vectors $\alpha_1, \alpha_2, \alpha_3 \in \mathbb{R}^{n\times 1}$, the linear gain matrices $K_1, K_2 \in \mathbb{R}^{n\times n}$, the signum function $sgn(\cdot)$, the element-wise absolute norm $|\cdot|$, and element-wise powers, such that [25, Eq. 28]:

$$\begin{aligned}
\dot{\hat{x}}_1 &= \hat{x}_2 + \alpha_3|x_1 - \hat{x}_1|^{2/3}sgn(x_1 - \hat{x}_1) + K_1(x_1 - \hat{x}_1),\\
\dot{\hat{x}}_2 &= \hat{f}(x_1, \hat{x}_2, u) + \alpha_2|\dot{\hat{x}}_1 - \hat{x}_2|^{1/2}sgn(\dot{\hat{x}}_1 - \hat{x}_2),\\
&\quad + K_2(x_1 - \hat{x}_1) + \hat{z}_{eq},\\
\dot{\hat{z}}_{eq} &= \alpha_1 sgn(\dot{\hat{x}}_1 - \hat{x}_2),
\end{aligned}$$

where $\hat{z}_{eq}$ is the observed input disturbance, which could also be used to improve the tracking performance, as described in [25]. For continuous-time systems, this observer has finite-time convergence, and can thus be trivially combined with stable tracking controllers, since the observer only has to reach the exact velocity before the controlled system would leave the stability bounds [42].

G. IMPLEMENTATION

Except for derivative filtering, the above estimation methods and their properties have been developed and presented in the literature assuming continuous-time control. Real implementations, however, are usually in discrete time, which is why we briefly review their implementation here.

Discrete-time versions of Kalman filters can be obtained by transforming the system model to discrete time and solving the discrete Riccati equation. As an example, the MATLAB functions `c2d`, `dlqe`, and `destim` provide the respective functionality. Discrete-time implementation of both linear and nonlinear high-gain observers are reviewed in [20, Ch. 9], and boundedness of the estimation error has been shown. For the linear version, the bilinear transformation performs best, as shown in [20], and can also be formulated as an FIR filter [44]. For both nonlinear high-gain and sliding-mode observers, the forward difference transformation can be used, for which boundedness of the estimation error has been shown in [20] and [42].

III. GAIN TUNING USING A GENETIC ALGORITHM

Control gain tuning is one of the main concerns in industrial applications [45]. For a fair comparison, one has to find the optimal gains for each estimator—some of them feature up to 90 gains, when every matrix element is considered (see Tab. 2). Manually tuning the gains is a time-consuming task for some estimation methods. Instead, we propose to use an automatic approach to find the optimal gains, which can be applied to all estimators.

TABLE 2. Number of gains of velocity estimators for 6-DOF robots

Method	Gains	Total #	Reduced #
Moving Average	n	6	6
BDE, TSE	$n_{\mathrm{BDE,TSE}}$	6	6
LSF	$n_{\mathrm{LSF}}, p_{\mathrm{LSF}}$	12	12
Kalman filter	Q	6	6
Linear high-gain	ϵ_l, H_1, H_2	73	12
Nonlinear high-gain	ϵ_n, L_1, L_2	73	12
Sliding Mode	$\alpha_1, \alpha_2, \alpha_3, K_1, K_2$	90	8

Possible automatic tuning techniques for PID controllers are reviewed in [46]; however, these are not applicable to multi-input multi-output systems. Instead, genetic algorithms (GA) have shown promising results for the gain tuning of nonlinear controllers, as demonstrated by simulations in [47] for flight controllers and in [48]–[50] for robot controllers. Genetic algorithms are bio-inspired techniques, for which existing tools can be used, e.g., the Global Optimization Toolbox in MATLAB.

To accelerate the tuning process, we reduce the number of gains. For high-gain observers, the original authors propose to choose H_1, H_2, L_1, L_2 a priori and subsequently decrease ϵ_l, ϵ_n as far as possible. As can be seen in the equations of Sec. II-D and Sec. II-E, the ϵ gains are, however, redundant, since one can equally choose large values for the matrices. This is why we arbitrarily set $\epsilon_l = \epsilon_n = 0.03$ a priori and tune H_1, H_2, L_1, L_2 instead. Additionally, we only consider to tune their diagonals to reduce the number of gains. For the sliding-mode observer, we choose $\alpha_1 = 1.1f^+, \alpha_2 = 1.5(f^+)^{1/2}$, and $\alpha_3 = 1.9(f^+)^{1/3}$, as proposed in [42], [43], where $f^+ \in \mathbb{R}^6$ represents the upper bound of the model perturbation. Furthermore, we replace K_1 and K_2 by the scalars k_1 and k_2, respectively. We found these choices to be suitable as a compromise between optimality and decreased tuning effort.

The cost function we use in this paper for tuning, as well as for evaluating the performance of the estimators is the integral squared error (ISE) of the velocity estimation

$$\mathrm{ISE}(t) = \int_0^t (x_2(\tau) - \hat{x}_2(\tau))^2 d\tau. \tag{5}$$

This cost function depends on a measurement of the ground truth of x_2, which must be more accurate than the estimation from the reviewed methods. In our case, we run the estimation methods online at a lower sampling rate with a lower encoder resolution, while the ground truth position is measured by the same encoders at a higher sampling rate and resolution. The ground truth velocity is then obtained offline by computing the finite difference and downsampling it with an anti-aliasing filter [51] to match the sampling rate of the online estimation methods. If measuring at a higher sampling rate and resolution is not feasible, we propose to use offline zero-phase filtering [51] to obtain a ground truth, so that we can minimize the phase delay of the estimated velocity.

TABLE 3. Hyperparameters of GA for gain tuning.

Parameter	Value
Population size	60
Number of elites	6
Selection	Tournament
Crossover	Uniform crossover at 87.5%
Mutation	Random mutation at 0.38%
Maximum generation	20

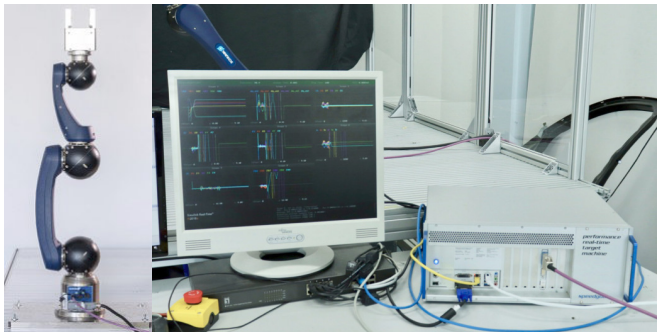


FIGURE 1. The testbed consists of a 6-DOF robot manipulator and a controller running on Simulink Real-Time.

The hyperparameters of the genetic algorithm are chosen to be almost the same as in [49] (see Tab. 3), except for the number of generations. Although our tuning is carried out on a real robot instead of simulations, we determined that 20 generations are sufficient for the gains to converge to an optimal value.

IV. EXPERIMENTAL COMPARISON

In this section, we experimentally compare the velocity estimation methods reviewed in Sec. II. Our testbed consists of a 6-DOF Schunk LWA-4P robot, whose model has been identified in [52]. The controller and estimators are implemented in Simulink Real-Time on a target machine with an i7-3770K 3.5 GHz processor (see Fig. 1). For the computed-torque controller, we choose $K_p = 100$ and $K_d = 13$. To measure the ground-truth velocity, we run the position encoder at 1 millidegree per increment at a sampling rate of 250 *Hz*. The actual velocity estimation is done at a resolution of 10 millidegrees per increment and at a sampling rate of 125 *Hz*.

We structure the experimental comparison as follows: in Sec. IV-A, we compare the tuning process using our proposed genetic algorithm. In Sec. IV-B, we show the main performance results, including the estimation error, the tracking error, and the convergence behavior of each estimator. Afterwards in Sec. IV-C, we compare the performance, when sensor faults are introduced. In Sec. IV-D, we compare the performance when using different encoder resolutions or sampling rates. The experimental comparison is concluded with a discussion of the results in Sec. IV-E. For simplicity in some of the plots, we only show the behavior of one axis because the behavior of the other axes are similar. The

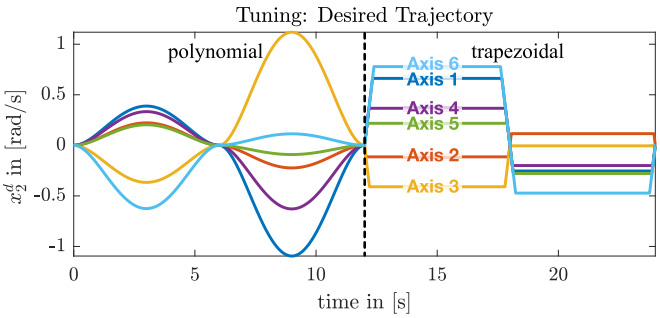


FIGURE 2. Desired trajectory for gain tuning.

TABLE 4. Optimal gains of velocity estimation methods (10 millidegrees, 125 Hz).

Method	Par.	Optimal values (diag. elements of matrices)					
MovAv	n	2					
LSF	n	4					
	p	1					
Kalman	Q	18.55	17.93	0.010	14.62	0.037	11.48
linHG	ϵ_l	0.03					
	H_1	47.01	25.98	59.80	54.96	35.83	34.25
	H_2	233.1	97.74	243.9	192.3	93.03	206.9
nonHG	ϵ_n	0.03					
	L_1	1.090	0.818	1.155	1.172	1.196	1.191
	L_2	1.812	0.999	2.205	2.428	2.006	2.552
SliMod	f^+	17.19	12.20	30.38	32.24	28.38	34.77
	k_1	84.46					
	k_2	41.03					

implemented velocity estimation methods, the experimental results, trajectories, and the robot model are provided as supplementary data[1] to this paper.

A. TUNING BEHAVIOR

We apply our automatic tuning procedure described in Sec. III to the four observers: Kalman filter, linear high-gain observer, nonlinear high-gain observer, and the sliding-mode observer. The remaining estimation methods only involve integer gains, which is why a grid search for each robot axis was sufficient. For tuning, we execute the trajectory displayed in Fig. 2 for each genome. With 20 generations, each with a population of 60 genomes, this translates to roughly 12 hours of tuning per estimation method, including the computation time.

In Fig. 3, we show how fast our genetic algorithm converges. The model-free observers (Kalman filter and linear high-gain observer) converge fast, while the model-based observers converge more slowly. According to our intuition, this may be because the gains of the model-based observers are more dependent on each other, where varying one gain affects the estimation performance of multiple joints. For the model-free observers, the gains are decoupled for each joint, which makes the search easier. The resulting optimal gains are shown in Tab. 4.

[1] https://dx.doi.org/10.21227/tse3-h285

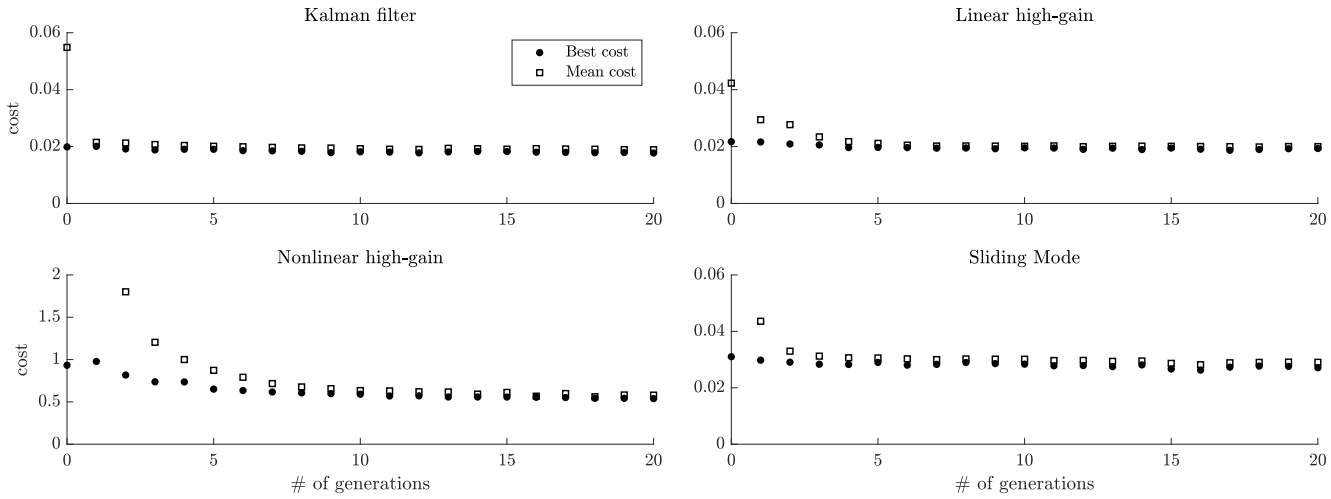


FIGURE 3. **Gain tuning using our proposed genetic algorithm. After each generation the mean cost (5) slowly approaches the best achieved cost per generation.**

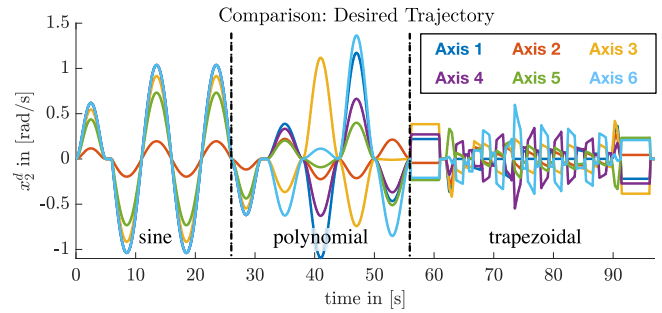


FIGURE 4. ***Estimation performance*: desired trajectory. The velocity signal $\dot{q}_d$ consists of a sine part, a polynomial part, and a trapezoidal part.**

B. ESTIMATION PERFORMANCE

We compare the performance of the velocity estimation methods using a more varied trajectory than the tuning trajectory. As shown in Fig. 4, it consists of a sine wave, a point-to-point trajectory in joint space using 5th-order polynomials, and a trapezoidal point-to-point trajectory in both joint and task space with inverse kinematics included. The experiments are performed in closed-loop control, meaning that the estimated velocities are directly applied to the computed-torque controller.

At first, we choose the best derivative filter out of the TSE, BDE, and LSF filters. Tab. 5 shows the ISE metric for the test trajectory for all considered filters. These are the same ones that have been analyzed in the previous comparisons in [4] and [5]. For this experiment (and all subsequent ones) we have determined that the LSF1/4-filter (i.e., $n_{\mathrm{BDE}} = 2$) has the smallest velocity estimation error. Mathematically, the LSF1/3-filter (i.e., $p_{\mathrm{BDE}} = 1$ and $n_{\mathrm{LSF}} = 3$) equals the moving average filter for $n = 2$, and LSF1/2, BDE1, and TSE1 equal the finite difference method, which is why we exclude them in this comparison.

TABLE 5. **Integral squared error of the derivative filters (10 millidegrees, 125 Hz).**

Filter	ISE	Filter	ISE
MovAv	0.035	LSF3/8	0.434
LSF1/4	0.046	BDE2	0.092
LSF1/8	0.114	BDE3	0.184
LSF2/8	1.197	TSE3	0.128

Next, we compare the LSF1/4-filter with all other optimally tuned estimation methods. To reflect the fact that the estimation performance can vary over time when used in closed-loop, we run the test trajectory four times for each estimator. In Fig. 5B we show the mean cumulative ISE over the course of the test trajectory, as well as their maxima and minima (shaded areas).

Except for the nonlinear high-gain observer, all other methods have a very similar estimation error. By small margins we can see that the Kalman filter and the linear high-gain observer perform slightly better than the rest. Although the finite difference method performs well in terms of ISE, we can also see in Fig. 5A that it is a noisy estimation due to the quantization error of the position encoder. On the one hand, the moving average filter improves the smoothness, but on the other hand, the error is larger due to the increased delay. Except for nnlHG, the other estimation signals are less noisy than the finDiff, while having a smaller delay than MovAv, which results in smaller estimation errors. The sliding-mode observer behaves interestingly: for the smooth sine and polynomial trajectories, it is an accurate estimation method. However, in the trapezoidal section, the error increases faster than for other methods, especially at the sections with sudden high acceleration.

In terms of the performance of the tracking control, the estimation methods do not differ significantly. As the

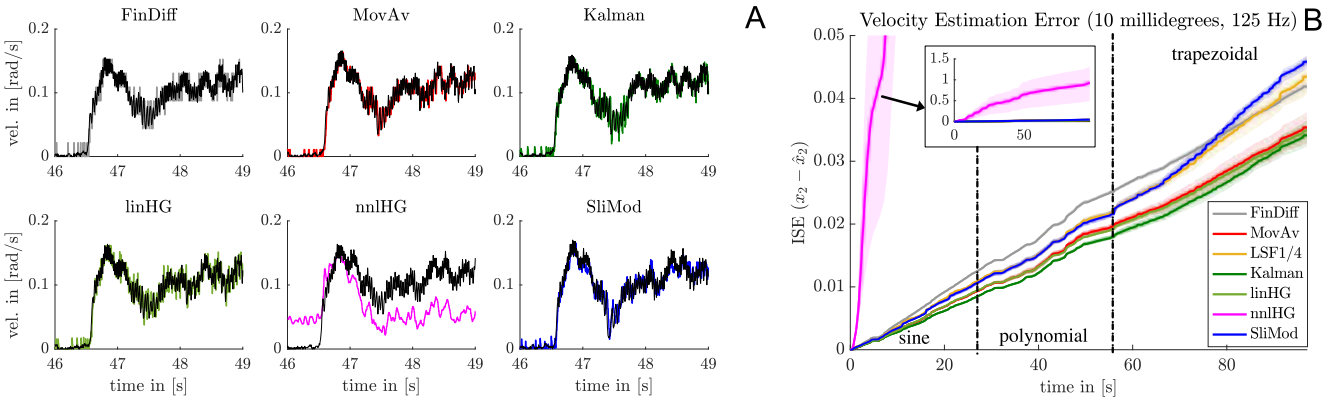


FIGURE 5. ***Estimation performance*: velocity error. The left side (A) shows an excerpt of the estimated velocity $\hat{x}_2$ (colored) versus the ground truth x_2 (black) for axis 6. The right side (B) shows the cumulative mean integral squared error (ISE) for each method (four experiments each).**

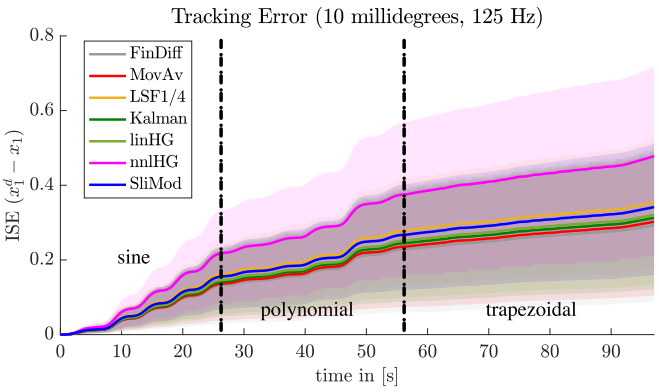


FIGURE 6. ***Estimation performance*: tracking error. Cumulative ISE of the tracking performance $x_1^d - x_1$.**

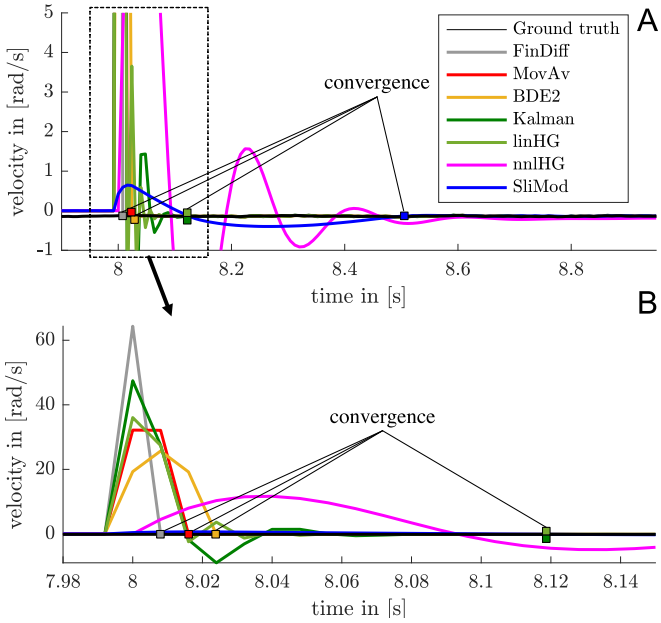


FIGURE 7. ***Estimation performance*: convergence of each method. Both plots show at different time scales the step response and the time of convergence (squares) of each estimator compared to the ground truth x_2 for axis 2 of the robot manipulator.**

overlapping shaded areas in Fig. 6 show for the ISE of the tracking error, the variation between multiple tests is far larger than the influence of the estimation method. Only nnlHG has a worse tracking error, resulting from its poor velocity estimation performance.

To explain the different behaviors of the methods, we analyse how fast they converge by analyzing their step responses. To do that, with reference to the result in Fig. 7, we execute a trapezoidal trajectory and activate the estimators simultaneously when the reference velocity is constant (t=8 seconds) to observe their response. We can see that the nonlinear high-gain observer never really converges, since it is not fast enough. The sliding-mode observer has the smallest overshoot, but requires much longer than the rest of the estimators to converge to the actual velocity. This is why in cases such as high accelerations in trapezoidal trajectories, the sliding-mode observer deviates, and the slow re-convergence accumulates to a large estimation error, although otherwise it is an accurate observer. The other methods converge significantly faster, which explains their good closed-loop performance.

C. FAULT TOLERANCE

We analyze how the estimation methods react to errors in the position measurement. To do that, we randomly simulate a loss of communication for 10% of the measurements of axis 6 of our robot. The resulting velocity estimation can be seen in Fig. 8A. The estimated velocities experience severe chattering, except for the sliding mode observer, which responds more robustly by remaining smoother.

However, the robustness of the estimators can be improved. To demonstrate that, we repeat the tuning process, in which the sensor stays faulty. As Fig. 8B shows, the estimation improves. As Tab. 6 shows, the retuned estimators significantly sacrifice accuracy during normal operation, except for the Kalman filter and the sliding-mode observer,

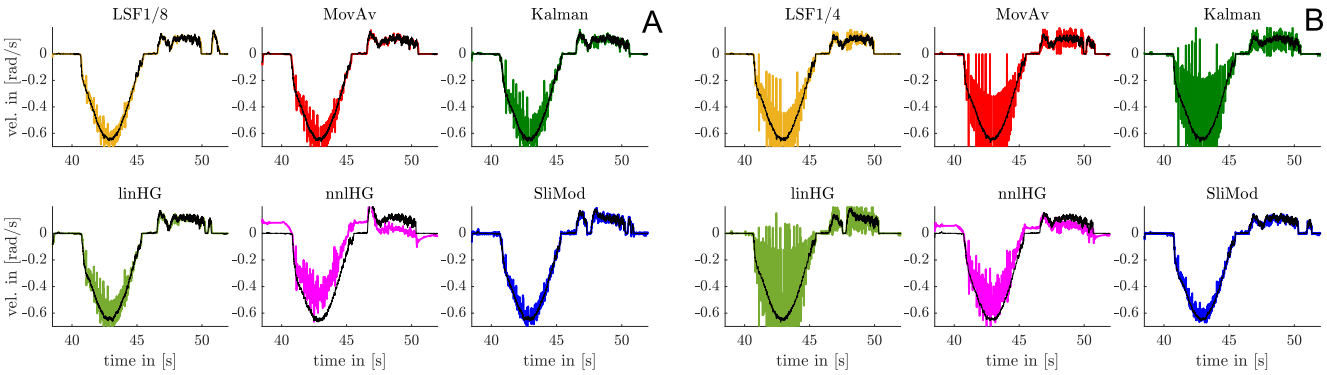


FIGURE 8. ***Sensor fault tolerance*: the left side (A) shows an excerpt of the estimated velocity of axis 6 given sensor errors, using the parameters from Tab. 4. The right side (B) shows the estimated velocity with new gains, that mitigate the noise.**

TABLE 6. Estimation error (ISE) with faulty sensors.

	normal	retuned normal	faulty	retuned faulty
FinDiff	0.041	0.041	2.737	2.737
MovAv	0.033	0.097	0.763	0.160
LSF	0.041	0.088	0.526	0.207
Kalman	0.032	0.036	1.162	0.177
linHG	0.033	0.060	1.831	0.145
nonHG	0.508	2.011	1.312	2.017
SliMod	0.047	0.049	0.067	0.074

TABLE 7. Optimal gains of velocity estimation methods (2 millidegrees, 125 Hz).

Method	Par.	Optimal values (diag. elements of matrices)					
MovAv	n	2					
LSF	n	4					
	p	1					
Kalman	Q	9.650	5.479	15.34	9.382	7.265	1.453
linHG	ϵ_l	0.03					
	H_1	36.81	30.77	45.18	27.40	55.17	34.04
	H_2	284.6	501.7	475.1	330.8	219.4	289.2
nonHG	ϵ_n	0.03					
	L_1	1.011	0.768	1.153	1.180	1.193	1.057
	L_2	1.689	0.997	2.175	2.216	2.399	1.835
SliMod	f^+	19.04	25.38	19.93	29.86	29.27	48.02
	k_1	119.5					
	k_2	41.13					

which is why we conclude that these two are the most fault tolerant methods regarding our error model.

D. HIGHER SENSOR RESOLUTION AND SAMPLING RATE

At last, we compare the estimation methods when operating the robot at a higher sensor resolution and higher sampling rate. We repeat the tuning process for two configurations: 1) 2 millidegrees per increment and sampling at 125 *Hz*, and 2) 1 millidegrees per increment and sampling at 250 *Hz*. For the latter configuration we compute the ground truth using the `filtfilt` zero-phase filter from MATLAB with a cut-off frequency at 28 *Hz*.

TABLE 8. Optimal gains of velocity estimation methods (1 millidegree, 250 Hz).

Method	Par.	Optimal values (diag. elements of matrices)					
MovAv	n	2					
LSF	n	4					
	p	1					
Kalman	Q	1.07	16.71	26.07	7.87	0.88	1.28
linHG	ϵ_l	0.01					
	H_1	15.63	19.54	17.09	9.43	15.75	19.02
	H_2	32.11	50.34	34.72	25.92	19.05	38.35
nonHG	ϵ_n	0.01					
	L_1	1.637	2.433	2.079	1.671	2.144	1.753
	L_2	3.061	3.359	3.463	3.343	3.046	3.404
SliMod	f^+	26.84	21.19	37.39	26.75	32.29	44.83
	k_1	37.47					
	k_2	88.84					

The resulting optimal gains are shown in Tab. 7 and Tab. 8, and the cumulative ISEs are shown in Fig. 9. The estimation errors decrease due to the improved measurement and the higher gains for the observer methods. What is immediately noticeable from the ISE graphs is that the relative difference between the filtering methods (FinDiff, MovAv, LSF1/4) depends much on the sampling rate and the encoder resolution. In the first case, FinDiff and LSF1/4 perform better than MovAv; in the second case, MovAv is the best of the filters. In contrast, the relative difference between the observer methods (Kalman, linHG, nnlHG, SliMod) stays similar in all considered cases. The linear high-gain observer is consistently amongst the best in terms of accuracy in all experiments.

E. DISCUSSION

We have thoroughly investigated the performance of velocity estimation methods, considering multiple aspects that are important for applying them to real robots. The moving average and the derivative filters are easy to implement and to tune, but their accuracy varies between different sensor resolutions and sampling rates. In our experiments, the derivative filters (LSF, BDE, TSE) did not perform very well and can lead to large errors, as can be seen by comparing

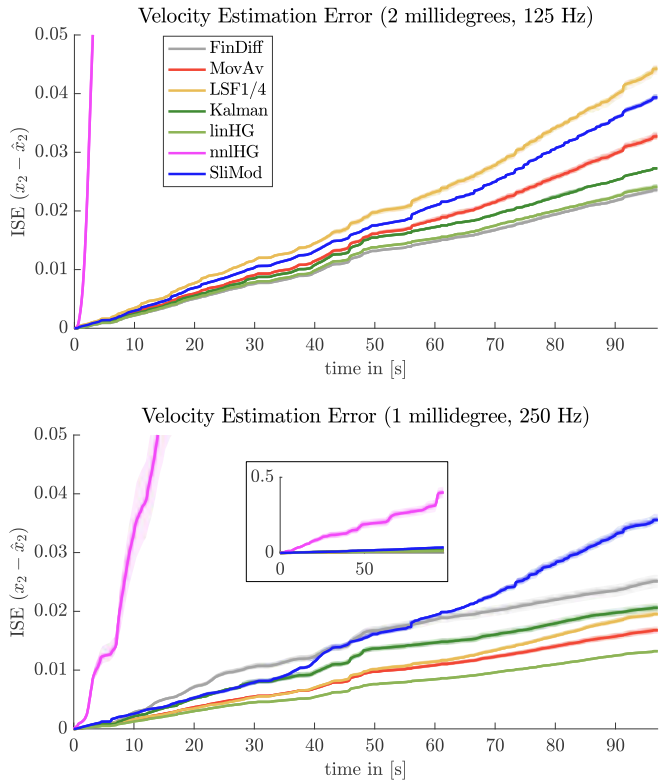


FIGURE 9. ***Higher sensor resolution and sampling rate*: comparison of the velocity estimation methods.**

Tab. 5 to the ISE. The linear high-gain observer consistently has the best accuracy when the gains are properly tuned. The nonlinear high-gain observer has not proven to be a suitable option in our experiments, although we tried our best to identify the robot model as accurately as possible. The sliding-mode observer, which is also model-based, performs well and has the added benefit of being robust against erroneous sensor measurements, but often experienced a loss of accuracy at high accelerations. From our comparison we cannot conclude that model-based observers, which are harder to implement due to the dynamical model of the robot, perform better than model-free estimation methods. Finally, our experiments have shown that most of the chosen estimators do not noticeably influence the tracking error, when they are optimally tuned. However, an inaccurate velocity estimation, such as the nonlinear high-gain observer in Fig. 6, will significantly degrade the tracking performance of the controller. This emphasizes the practical relevance of having accurate estimates and the importance of tuning to reach the best performance. In Tab. 9, we qualitatively summarize our discussed observations.

V. CONCLUSION

This work experimentally compares multiple velocity estimation methods for robot manipulators, namely the finite difference algorithm, moving average filtering, derivative filtering, Kalman filtering, linear high-gain observer, nonlinear high-gain observer and the sliding mode observer. Additionally, we propose an automatic tuning procedure based on a genetic algorithm. The linear high-gain observer is consistently amongst the best in terms of accuracy, independent of the sampling rate and sensor resolution, while the sliding-mode observer is robust against sensor faults. Simple-to-implement schemes, such as the moving average filter, can perform well enough, when optimally tuned, without affecting the tracking error of the closed-loop robot system. The nonlinear high-gain observer was not suitable for our robot. Overall, when the other estimators are tuned using our genetic algorithm, their optimal performance is similar.

TABLE 9. Qualitative comparison of each estimation method.

	MovAv	Deriv. filters	Kalman	linHG	nnlHG	SliMod
Implementation	●	●	◐	◐	○	○
Tuning	●	●	◐	◐	◐	◐
Estimation accuracy	◐	◐	◐	●	○	◐
Fault tolerance	◐	◐	●	◐	◐	●
Tracking error	●	●	●	●	◐	●

● = best, ○ = worst, and ◐ = in between

REFERENCES

[1] B. Siciliano and O. Khatib, Eds., *Springer Handbook of Robotics*. Springer International Publishing, 2016.

[2] C. Canudas De Wit, N. Fixot, and K. J. Aström, "Trajectory tracking in robot manipulators via nonlinear estimated state feedback," *IEEE Trans. on Robotics and Automation*, vol. 8, no. 1, pp. 138–144, 1992.

[3] S. Nicosia and P. Tomei, "Robot control by using only joint position measurements," *IEEE Trans. on Automatic Control*, vol. 35, no. 9, pp. 1058–1061, 1990.

[4] L. Bascetta, G. Magnani, and P. Rocco, "Velocity Estimation: assessing the performance of non-model-based techniques," *IEEE Trans. on Control Systems Technology*, vol. 17, no. 2, pp. 424–433, 2009.

[5] R. H. Brown, S. C. Schneider, and M. G. Mulligan, "Analysis of algorithms for velocity estimation from discrete position versus time data," *IEEE Trans. on Industrial Electronics*, vol. 39, no. 1, pp. 11–19, 1992.

[6] P. Carpenter, R. Brown, J. Heinen, and S. Schneider, "On algorithms for velocity estimation using discrete position encoders," in *Proc. of IEEE Conference on Industrial Electronics*, vol. 2, 1995, pp. 844–849.

[7] J. M. Daly and H. M. Schwartz, "Experimental results for output feedback adaptive robot control," *Robotica*, vol. 24, no. 6, pp. 727–738, 2006.

[8] S. Nicosia, A. Tornambè, and P. Valigi, "State estimation in robotic manipulators: some experimental results," *Journal of Intelligent & Robotic Systems*, vol. 7, no. 3, pp. 321–351, 1993.

[9] S. Islam, P. X. Liu, and A. Saddik, "Experimental comparison of model-based and model-free output feedback control system for robot manipulators," in *Proc. of the International Conference on Autonomous and Intelligent Systems*, 2011, pp. 177–188.

[10] B. Bona and M. Indri, "Analysis and implementation of observers for robotic manipulators," in *Proc. of the IEEE International Conference on Robotics and Automation*, vol. 4, 1998, pp. 3006–3011.

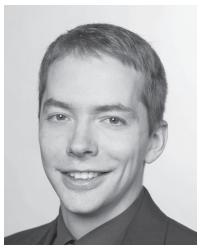
[11] G. S. Natal, A. Chemori, and F. Pierrot, "Nonlinear control of parallel manipulators for very high accelerations without velocity measurement: stability analysis and experiments on par2 parallel manipulator," *Robotica*, vol. 34, no. 1, pp. 43—-70, 2016.

[12] S. Islam and P. X. Liu, "Output feedback sliding mode control for robot manipulators," *Robotica*, vol. 28, no. 7, pp. 975–987, 2010.

[13] ——, "PD output feedback control design for industrial robotic manipulators," *IEEE/ASME Trans. on Mechatronics*, vol. 16, no. 1, pp. 187–197, 2011.
[14] ——, "Robust adaptive fuzzy output feedback control system for robot manipulators," *IEEE/ASME Trans. on Mechatronics*, vol. 16, no. 2, pp. 288–296, 2011.
[15] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics. Modelling, Planning and Control*, ser. Advanced Textbooks in Control and Signal Processing. London: Springer London, 2009.
[16] R. C. Kavanagh and J. M. Murphy, "The effects of quantization noise and sensor nonideality on digital differentiator-based rate measurement," *IEEE Trans. on Instrumentation and Measurement*, vol. 47, no. 6, pp. 1448–1456, 1998.
[17] S. Y. Lim, D. M. Dawson, and K. Anderson, "Re-examining the Nicosia-Tomei robot observer-controller from a backstepping perspective," *IEEE Trans. on Control Systems Technology*, vol. 4, no. 3, pp. 304–310, 1996.
[18] C.-C. Yih, "Extended nicosia–tomei velocity observer-based robot-tracking control," *IET Control Theory & Applications*, vol. 6, pp. 51–61(10), 2012.
[19] K. W. Lee and H. K. Khalil, "Adaptive output feedback control of robot manipulators using high-gain observer," *International Journal of Control*, vol. 67, no. 6, pp. 869–886, 1997.
[20] H. K. Khalil, *High-Gain Observers in Nonlinear Feedback Control*. Society for Industrial and Applied Mathematics, 2017.
[21] M. Erlic and W.-S. Lu, "A reduced-order adaptive velocity observer for manipulator control," *IEEE Trans. on Robotics and Automation*, vol. 11, no. 2, pp. 293–303, 1995.
[22] S. Malagari and B. Driessen, "Globally exponential controller/observer for tracking in robots without velocity measurement," *Asian Journal of Control*, vol. 14, no. 2, pp. 309–319, 2012.
[23] P. Ordaz, "Nonlinear robust output stabilization for mechanical systems based on Luenberger-like controller/observer," *ASME Journal of Dynamic Systems, Measurement and Control*, vol. 139, no. 8, pp. 1–6, 2017.
[24] C. Canudas de Wit and J. J. E. Slotine, "Sliding observers for robot manipulators," *Automatica*, vol. 27, no. 5, pp. 859–864, 1991.
[25] M. Van, H.-J. Kang, Y. S. Suh, and K. S. Shin, "Output feedback tracking control of uncertain robot manipulators via higher-order sliding-mode observer and fuzzy compensator," *Journal of Mechanical Science and Technology*, vol. 27, no. 8, pp. 2487–2496, 2013.
[26] Z. Qu, D. M. Dawson, J. F. Dorsey, and J. D. Duffie, "Robust estimation and control of robotic manipulators," *Robotica*, vol. 13, no. 3, pp. 223—-231, 1995.
[27] M. Arteaga and R. Kelly, "Robot control without velocity measurements: new theory and experimental results," *IEEE Trans. on Robotics and Automation*, vol. 20, no. 2, pp. 297–308, 2004.
[28] F. Bouakrif, D. Boukhetala, and F. Boudjema, "Passivity-based controller observer for robot manipulators," *International Journal of Robotics and Automation*, vol. 25, no. 1, pp. 1–8, 2010.
[29] F. Bouakrif, "Trajectory tracking control using velocity observer and disturbances observer for uncertain robot manipulators without tachometers," *Meccanica*, vol. 52, no. 4-5, pp. 861–875, 2017.
[30] J. A. Heredia and W. Yu, "High-gain observer-based PD control for robot manipulator," in *Proc. of the American Control Conference*, 2000, pp. 2518–2522.
[31] P. Pagilla and M. Tomizuka, "An adaptive output feedback controller for robot arms: stability and experiments," *Automatica*, vol. 37, no. 7, pp. 983–995, 2001.
[32] X. Liang, X. Huang, M. Wang, and X. Zeng, "Adaptive task-space tracking control of robots without task-space- and joint-space-velocity measurements," *IEEE Trans. on Robotics*, vol. 26, no. 4, pp. 733–742, 2010.
[33] Y. Kim and F. Lewis, "Neural network output feedback control of robot manipulators," *IEEE Trans. on Robotics and Automation*, vol. 15, no. 2, pp. 301–309, 1999.
[34] T. Sun, H. Pei, Y. Pan, H. Zhou, and C. Zhang, "Neural network-based sliding mode adaptive control for robot manipulators," *Neurocomputing*, vol. 74, no. 14-15, pp. 2377–2384, 2011.
[35] P. R. Bélanger, P. Dobrovolny, A. Helmy, and X. Zhang, "Estimation of angular velocity and acceleration from shaft-encoder measurements," *International Journal of Robotics Research*, vol. 17, no. 11, pp. 1225–1233, 1998.
[36] A. Loria, "Observers are unnecessary for output-feedback control of Lagrangian systems," *IEEE Trans. on Automatic Control*, vol. 61, no. 4, pp. 905–920, 2016.
[37] P. R. Bélanger, "Estimation of angular velocity and acceleration from shaft encoder measurements," in *Proc. of the IEEE International Conference on Robotics and Automation*, 1991, pp. 585–592.
[38] S. M. Phillips and M. S. Branicky, "Velocity estimation using quantized measurements," in *Proc. of the IEEE Conference on Decision and Control*, 2003, pp. 4847–4852.
[39] F. Auger, M. Hilairet, J. M. Guerrero, E. Monmasson, T. Orlowska-Kowalska, and S. Katsura, "Industrial Applications of the Kalman Filter: A Review," *IEEE Trans. on Industrial Electronics*, vol. 60, no. 12, pp. 5458–5471, 2013.
[40] C. A. Lightcap and S. A. Banks, "An extended Kalman filter for real-time estimation and control of a rigid-link flexible-joint manipulator," *IEEE Trans. on Control Systems Technology*, vol. 18, no. 1, pp. 91–103, 2010.
[41] A. N. Atassi and H. K. Khalil, "A separation principle for the stabilization of a class of nonlinear systems," *IEEE Trans. on Automatic Control*, vol. 44, no. 9, pp. 1672–1687, 1999.
[42] J. Davila, L. Fridman, and A. Levant, "Second-order sliding-mode observer for mechanical systems," *IEEE Trans. on Automatic Control*, vol. 50, no. 11, pp. 1785–1789, 2005.
[43] L. Fraguela Cuesta, L. Fridman, and V. V. Alexandrov, "Position stabilization of a Stewart platform: high-order sliding mode observers based approach," in *Proc. of the IEEE Conference on Decision and Control and European Control Conference*, 2011, pp. 5971–5976.
[44] A. M. Dabroom and H. K. Khalil, "Discrete-time implementation of high-gain observers for numerical differentiation," *International Journal of Control*, vol. 72, no. 17, pp. 1523–1537, 1999.
[45] Z. Gao, "Scaling and bandwidth-parameterization based controller tuning," in *Proc. of the American Control Conference*, 2003, pp. 4989–4996.
[46] K. J. Åström, T. Hägglund, C. C. Hang, and W. K. Ho, "Automatic tuning and adaptation for PID controllers - a survey," *Control Engineering Practice*, vol. 1, no. 4, pp. 699–714, 1993.
[47] K. Krishnakumar and D. E. Goldberg, "Control system optimization using genetic algorithms," *Journal of Guidance, Control, and Dynamics*, vol. 15, no. 3, pp. 735–740, 1992.
[48] S. Ge, T. Lee, and G. Zhu, "Genetic algorithm tuning of Lyapunov-based controllers: an application to a single-link flexible robot system," *IEEE Trans. on Industrial Electronics*, vol. 43, no. 5, pp. 567–574, 1996.
[49] F. Nagata, K. Kuribayashi, K. Kiguchi, and K. Watanabe, "Simulation of fine gain tuning using genetic algorithms for model-based robotic servo controllers," in *Proc. of the IEEE International Symposium on Computational Intelligence in Robotics and Automation*, 2007, pp. 196–201.
[50] W. Ainhauser, J. Gerstmayr, and A. Giusti, "Multi-objective trajectory tracking optimization for robots with elastic joints," in *Advances in Service and Industrial Robotics*. Cham: Springer, 2021, pp. 250–258.
[51] A. V. Oppenheim, R. W. Schafer, and J. R. Buck, *Discrete-Time signal processing*, 2nd ed. Prentice Hall, 1999.
[52] S. B. Liu and M. Althoff, "Reachset conformance of forward dynamic models for the formal analysis of robots," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2018, pp. 370–376.

STEFAN B. LIU (Graduate Student Member, IEEE) received the B.S. degree in mechatronics, and the M.S. degree in robotics from the Technical University of Munich (TUM), Germany, in 2015, and 2017, respectively. He is currently pursuing a Ph.D. degree at the Cyber-Physical Systems Group of the TUM Department of Informatics. His research interest includes formal methods in robotics, physical human-robot interaction, modeling and identification, and modular robots.

ANDREA GIUSTI (Member, IEEE) is researcher, head of Robotics and Intelligent Systems Engineering, at Fraunhofer Italia Research, Italy. He received his Ph.D. degree in Robotics in 2018, from the Technical University of Munich (TUM), Germany. He received his Masters Degree in Mechatronic Engineering, summa cum laude, in 2013, and his Bachelors Degree in Telecommunications Engineering in 2010, both from the University of Trento, Italy. His research interests include modeling and control of robotic and mechatronic systems, modular and reconfigurable robots, and human-robot collaboration.



MATTHIAS ALTHOFF (Member, IEEE) is Associate Professor in computer science at the Technical University of Munich, Germany. He received his Diploma Engineering Degree in Mechanical Engineering in 2005, and his Ph.D. in Electrical Engineering in 2010, both from the Technical University of Munich, Germany. From 2010 to 2012 he was a postdoctoral researcher at Carnegie Mellon University, Pittsburgh, USA, and from 2012 to 2013 an assistant professor at the Ilmenau University of Technology, Germany. His research interests include formal verification of continuous and hybrid systems, reachability analysis, planning algorithms, nonlinear control, robotics, automated vehicles, and power systems.