Multibody System Dynamics (2025) 66:1033–1055
https://doi.org/10.1007/s11044-025-10085-x

RESEARCH

Kalman filter based on marker and force plate measurements and a full multibody model, for the real–time capture, reconstruction and analysis of human movement

Urbano Lugrís[1] · **Manuel Pérez–Soto**[2] · **Santiago Beron**[1] · **Florian Michaud**[1]

Received: 31 December 2024 / Accepted: 13 May 2025 / Published online: 28 May 2025

Abstract

Currently, optical motion capture remains the gold standard for human motion analysis. This technique estimates the movement of a subject by tracking a set of markers attached to their skin, then using the captured trajectories to reconstruct the movement of an underlying rigid–body model. However, since the markers are not rigidly fixed to the skeleton, their motion relative to the bones can induce significant estimation errors, especially when using the computed accelerations to calculate the joint torques by solving the inverse dynamics. This paper presents an extended Kalman filter aimed at reducing such errors, through the incorporation of a complete multibody model of the human body. The method builds upon an existing Kalman filter based on a purely kinematic plant model, which uses only optical markers as sensors. The proposed observer adds the motor efforts and external reactions to the system states, and incorporates force plates into the set of sensors. This allows the measured ground reactions to directly affect the dynamics, rather than remaining an input to a subsequent inverse dynamics. The dynamic model can be added to the Kalman filter with multiple options and simplifications, which are here examined to find the combination with the best balance between efficiency and accuracy. The performance of the selected implementation is then compared with the existing kinematics–based observer. The results show that the proposed method improves the estimation of joint torques and reduces the influence of marker bounces on them, in exchange for a higher computational cost and an increased estimation delay.

Keywords Biomechanics · Motion capture · Extended Kalman filter

1 Introduction

When performing human motion analysis, the most frequent tool of choice is marker–based optical motion capture. Although other techniques exist, such as inertial sensors, or marker-

---

U. Lugrís
urbano.lugris@udc.es

M. Pérez–Soto
manuel.perez@unizar.es

[1] Laboratory of Mechanical Engineering, CITENI, Campus Industrial de Ferrol, Universidade da Coruña, Mendizábal s/n, 15403 Ferrol, Spain

[2] IDERGO, University of Zaragoza, María de Luna 3, 50010 Zaragoza, Spain

less algorithms relying on computer vision, marker–based methods remain the gold standard due to their superior accuracy and real–time capabilities [1, 2]. However, this technology still presents many problems, among which the so–called *skin motion artifact*, also known as *soft tissue artifact*, remains as one of the most relevant [3, 4]. The problem arises from the fact that the optical markers are directly attached to the skin of the subject, whereas most motion reconstruction algorithms assume they are rigidly fixed to the underlying skeleton. Apart from the obvious implications on the accuracy of the reconstructed motion, the soft tissue artifact can become a major problem when said motion is used for input estimation, i.e., to compute motor efforts by solving the inverse dynamics. Since the markers are not rigidly attached to the bones, they can achieve significant accelerations with respect to them, especially in the presence of impacts and energetic movements. These accelerations, due to the very nature of the method, are inevitably transferred to the estimated motion of the underlying rigid–body model, thus altering the results of the inverse dynamics.

There exist multiple approaches to address the skin motion artifact in the literature. Many methods aim at finding the rigid–body motion that best tracks the measured markers, either by performing a model fitting after the motion has been reconstructed [5], or by stating the whole motion estimation as a global optimization [6, 7]. However, this does not completely solve the problem, since the relative displacements of the markers are not random, so their effects do not necessarily cancel out after the optimization. There exist also attempts at modeling the skin motion artifact, for instance by correlating the marker deviations to the joint angles for specific movements [8]. Although such methods can be effective to reduce the marker tracking error, they do not account for inertia–driven marker perturbations, such as those produced during an impact.

Optimal control algorithms such as direct collocation, model–predictive control, or moving–horizon estimation can also be used for solving input estimation problems. Most of these methods are either based on a forward–dynamics approach, or enforce the fulfillment of the equations of motion by treating them as optimization constraints, thus ensuring dynamic consistency. Although these techniques have been commonly used in the past for predictive simulations [9, 10], several authors have also employed them for estimating joint torques [11, 12]. However, optimal control methods are computationally expensive, since they require running numerous simulations, which renders them less suitable for real–time applications, although it is possible to achieve such performance for reduced models with limited frame rates [13].

The extended Kalman filter (EKF) is another alternative to reconstruct human motion [14–16], well suited for real–time applications [17, 18]. However, the referred implementations use a purely kinematic plant model, in such a way that the movement is completely decoupled from its driving efforts, which are obtained in a post–processing stage. A more integrated way of performing input estimation is to incorporate a dynamic model of the system into the plant, rather than a simple velocity or acceleration propagation one. There exist multiple state observers combining multibody models and Kalman filters in non–biomechanical applications [19–21], including several methods specifically devoted to input estimation [22, 23]. A thorough review of such applications can be found in [24].

This work presents an EKF with an embedded multibody model and force sensors aimed at real–time human motion capture and analysis. The algorithm, hereafter referred to as EKF-D, is more computationally intensive than using an observer with a simple acceleration propagation model followed by inverse dynamics [18] (denoted as EKF-K from now). Adding a dynamic model to a Kalman filter requires to integrate it in time, and also to compute complex state propagation and plant noise covariance matrices, which are in turn based on a model linearization. In order to reduce the impact on efficiency without severely penalizing accuracy, multiple simplifications and implementation alternatives are explored in this work. This is done through a virtual experiment with known solution, consisting on a basic 2D double pendulum model performing a predefined motion. Subsequently, the EKF-K and the best EKF-D implementation are compared using full–body 3D motion capture data.

2 Marker–based state observers in biomechanics

This Section will describe the two different marker–based methods aimed at estimating the motor efforts of human movement in real time considered here, EKF-K and EKF-D. The latter is the actual contribution of the present work, being the former included only for comparison purposes. The EKF-K is an EKF with optical markers as sensors, and relies on a basic acceleration propagation model to reconstruct the motion [25]. Since this state observer only provides the kinematics, the motor efforts and ground reactions have to be obtained separately, by solving the inverse dynamics problem at every time step [18]. However, by calculating them in this way, they have no effect on the progression of the observer states. Thus, aiming at a more integral approach, the EKF-D incorporates the complete equations of motion into the plant model, adding the force plates as sensors, and including the ground reactions and motor efforts into the vector of system states, in such a way that the movement is no longer exclusively governed by the trajectories of the optical markers.

Both observers are based on the EKF, which is a popular predictor–corrector algorithm commonly used for state estimation in nonlinear systems. The recursive equations of the EKF are well–known [25–28], so they will not be repeated here for the sake of brevity. Instead, this section will describe how to obtain all the terms involved in the algorithm, namely the state vector $\mathbf{x}$, the state propagation matrix $\boldsymbol{\Phi}$, the process noise covariance matrix $\boldsymbol{\mathcal{Q}}$, the observation function $\mathbf{h}(\mathbf{x})$, and the sensors noise covariance matrix $\boldsymbol{\mathcal{R}}$.

2.1 EKF-K: kinematics–only observer with inverse dynamics

2.1.1 Plant model

There exist multiple ways to formulate a kinematics–based plant model for a Kalman filter. The nature of the plant noise can vary depending on which magnitude is considered as such, and also on whether the propagation model is obtained by discretizing a continuous system in time or by defining it directly as discrete. In this case, a Discrete Wiener Process Acceleration model (DWPA) [25], which is very simple and uses an intuitive definition of the plant noise, is employed to describe the plant. In this model, the state propagation is defined directly as a discrete equation, where the acceleration increment at every time step is assumed to behave as discrete Gaussian noise. Considering a system with n_z degrees of freedom, the state vector $\mathbf{x}_i$, containing the position z_i, velocity $\dot{z}_i$ and acceleration $\ddot{z}_i$ of a given degree of freedom i, propagates in time between two consecutive time steps k and $k+1$ as follows:

$$\mathbf{x}_{i_{k+1}} = \boldsymbol{\Phi}_i \mathbf{x}_{i_k} + \boldsymbol{\Gamma}_i v_{i_k} \qquad i = 1, \dots, n_z \tag{1}$$

Here, v_{i_k} represents a random acceleration increment (or angular acceleration increment, depending on the nature of the corresponding degree of freedom) which remains constant along time step k. The state propagation matrix $\boldsymbol{\Phi}_i$ and the noise gain vector $\boldsymbol{\Gamma}_i$ are both constant and equal for all degrees of freedom, with the following structure:

$$\begin{bmatrix} z_i \\ \dot{z}_i \\ \ddot{z}_i \end{bmatrix}_{k+1} = \begin{bmatrix} 1 & \Delta t & \frac{1}{2}\Delta t^2 \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} z_i \\ \dot{z}_i \\ \ddot{z}_i \end{bmatrix}_k + \begin{bmatrix} \frac{1}{2}\Delta t^2 \\ \Delta t \\ 1 \end{bmatrix} v_{i_k} \qquad i = 1, \dots, n_z \tag{2}$$

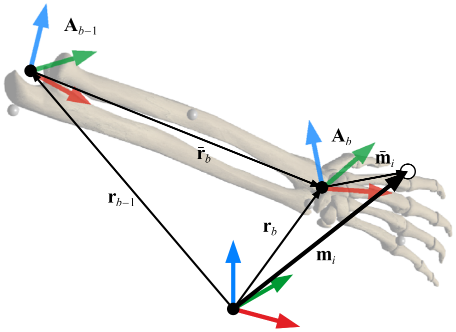

Fig. 1 Observation function schematic

The time step Δt is fixed to the frame rate of the cameras used by the motion capture system. The state vector $\mathbf{x}$ and the propagation matrix $\boldsymbol{\Phi}$ of the whole system are obtained by assembling all the individual $\mathbf{x}_i$ and $\boldsymbol{\Phi}_i$ terms, leading to a system of dimension $3 \times n_z$.

Assuming that the discrete Gaussian noise v_i has a variance $\sigma_{v_i}^2$, the process noise covariance matrix $\mathcal{Q}_i$ for the same degree of freedom can be obtained as [25]:

$$\mathcal{Q}_i = \boldsymbol{\Gamma}_i \sigma_{v_i}^2 \boldsymbol{\Gamma}_i^{\mathrm{T}} = \begin{bmatrix} \frac{1}{4}\Delta t^4 & \frac{1}{2}\Delta t^3 & \frac{1}{2}\Delta t^2 \\ \frac{1}{2}\Delta t^3 & \Delta t^2 & \Delta t \\ \frac{1}{2}\Delta t^2 & \Delta t & 1 \end{bmatrix} \sigma_{v_i}^2 \qquad i = 1, \ldots, n_z \tag{3}$$

This matrix is also constant, although not necessarily equal for all degrees of freedom. The complete $\mathcal{Q}$ matrix of the whole system is, again, obtained by assembling the individual $\mathcal{Q}_i$ matrices, according to the arrangement of variables established for the state vector $\mathbf{x}$.

2.1.2 Observation model

The observation model is the same as described in [17] and [18]. The state observer uses n_m optical markers attached to the subject as position sensors, so the observation function $\mathbf{h}(\mathbf{x})$ must provide a vector $\mathbf{m}$ with their $3 \times n_m$ absolute coordinates. Since the markers are position sensors, $\mathbf{h}(\mathbf{x})$ depends only on the position–level part of $\mathbf{x}$, i.e. the independent coordinates of the model $\mathbf{z}$. The kinematic chain uses the pelvis as base body, thus $\mathbf{z}$ contains its position vector, together with the Euler angles that define the orientations of all the n_b body segments of the model.

In the observation function, the reflective markers are considered as rigidly attached to their corresponding body segments. As shown in Fig. 1, the absolute position vector $\mathbf{m}_i$ of a given marker i is obtained from the location of the proximal joint $\mathbf{r}_b$ of the body b the marker is attached to, the orientation matrix $\mathbf{A}_b$ of said body, and the local coordinates $\bar{\mathbf{m}}_i$ of the marker within the local frame of body b:

$$\mathbf{m}_i = \mathbf{r}_b + \mathbf{A}_b \bar{\mathbf{m}}_i \qquad i = 1, \ldots, n_m \tag{4}$$

The absolute location of the origin of body b, in turn, is obtained recursively in the same manner, from the location of the proximal joint of its parent body in the kinematic chain, $\mathbf{r}_{b-1}$, the parent's orientation matrix $\mathbf{A}_{b-1}$, and the local coordinates $\bar{\mathbf{r}}_b$ of the proximal joint of body b within the frame of reference of body $b-1$:

$$\mathbf{r}_b = \mathbf{r}_{b-1} + \mathbf{A}_{b-1} \bar{\mathbf{r}}_b \qquad b = 1, \ldots, n_b \tag{5}$$

The local coordinates of markers and joints, $\bar{\mathbf{m}}_i$ and $\bar{\mathbf{r}}_b$, are constant, and depend on the size of the subject. An automatic scaling process, described in detail in previous works [17, 18], is required in order to obtain the optimal values for these parameters at every capture session.

The state propagation function from Eq. (2) is linear, but the observation function $\mathbf{h}(\mathbf{x})$ defined by Eqs. (4) and (5) presents considerable nonlinearities, which leads to the necessity of using an EKF [25–28]. There exist more accurate options for nonlinear systems, such as the unscented Kalman filter but, in general, their higher computational cost makes them unsuitable for real–time applications, at least in systems with numerous states. As the propagation function of the EKF-K is linear, the algorithm only requires to linearize the observation function. Provided that the state vector $\mathbf{x}$ is assembled by placing the independent positions at the beginning, followed by the velocities, and leaving the accelerations at the end, then the Jacobian matrix of $\mathbf{h}(\mathbf{x})$ has the following structure:

$$\mathbf{H}(\mathbf{x}) = \begin{bmatrix} \dfrac{\partial \mathbf{m}}{\partial \mathbf{z}} & \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{6}$$

In the current implementation of the EKF-K, the rotation matrices of most bodies are defined by absolute Euler angles, which is enabled by the fact that nearly all kinematic joints in the model are spherical. This prevents the accumulation of rotation matrices along the kinematic chain, thus rendering the Jacobian of the observation function $\mathbf{H}(\mathbf{x})$ very simple and sparse, which improves the computational efficiency. According to this approach, the absolute rotation matrix of each body b is defined by three independent angles α_b, β_b and γ_b such that:

$$\mathbf{A}_b = \mathbf{R}_z(\gamma_b)\,\mathbf{R}_y(\beta_b)\,\mathbf{R}_x(\alpha_b) \qquad b = 1, \ldots, n_b \tag{7}$$

where $\mathbf{R}_x$, $\mathbf{R}_y$ and $\mathbf{R}_z$ are the standard elementary rotation matrices about the x, y and z axes, respectively. Using this sequence of rotations, the singularity happens when any β angle takes a value of $\pm\pi/2$. Assuming that the x axis points forward and the z axis points upward, the β angle corresponds to pitch rotation, which never reaches the aforementioned value in most common biomechanical applications, such as gait analysis. However, for more general movements, this problem needs to be addressed. The simplest way of (mostly) overcoming it is using relative rotations: since human joints have a limited range of motion, the rotation orders at each joint can be adjusted individually to avoid the singularity. However, this solution is not entirely general since, in principle, the base body can adopt any arbitrary orientation. A more robust and general option is using quaternions, although that implies introducing algebraic constraints into the EKF, which, especially in nonlinear systems, may hinder its computational efficiency [29]. However, in many applications, it is sufficient to normalize the quaternions after the Kalman corrector [30, 31].

The $3 \times n_m$ sensors are treated as being independent by the filter, so the sensor noise covariance matrix $\boldsymbol{\mathcal{R}}$ is assumed to be diagonal. In this work, for the sake of simplicity, the three coordinates of all markers are considered as affected by zero–mean Gaussian noise with the same variance σ_s^2. Therefore, all the terms in the diagonal of $\boldsymbol{\mathcal{R}}$ are equal to this value. In order to obtain improved results, the individual variances can be tuned according to the reliability of each specific marker.

Markers are subject to two main sources of error. Firstly, the motion capture system introduces a certain amount of low–amplitude Gaussian noise to the measured trajectories, which gets amplified when using finite differences to calculate velocities and accelerations. In traditional reconstruction methods, this is solved by applying a low–pass filter to the positions prior to the numerical differentiation [32]. The second source of error is the already mentioned skin motion artifact, i.e. the displacement of the markers from their reference rigid–body locations, due to their attachment to soft tissue. This effect cannot be strictly regarded as Gaussian noise, because it is correlated to the motion, so the results obtained are not going to be optimal.

2.1.3 Inverse dynamics

Once the model has been properly scaled and the Kalman filter is running, the joint torques and external reactions can be computed by solving the inverse dynamics problem. The equations of motion of the multibody system are stated in independent coordinates, using a velocity projection method, and have the following form:

$$\mathbf{M}(\mathbf{z})\,\ddot{\mathbf{z}} = \mathbf{Q}_0(\mathbf{z},\dot{\mathbf{z}}) + \mathbf{Q}_e \tag{8}$$

In this equation, the vector $\mathbf{Q}_e$ contains the unknown motor torques at the joints, plus a residual wrench acting at the base body, all of them expressed directly as generalized forces acting on the independent coordinates $\mathbf{z}$. The other two terms, namely the mass matrix $\mathbf{M}$ and the vector of remaining known generalized forces $\mathbf{Q}_0$, can be calculated by means of a velocity projection, using the procedure described in [18]. In order to obtain these terms, three sets of data are required: the motion, the inertia parameters of the body segments, and the ground reactions at the feet.

The motion is directly provided by the Kalman filter, which estimates positions, velocities and accelerations in real time. The inertia parameters are estimated by scaling their values from a standard model, as explained in [18]. In short, the procedure consists on taking reference parameters from the literature [5, 33], and applying to them the same scale factors obtained when fitting the model to the markers. And the ground reactions are measured using force plates, synchronized with the optical motion capture system. They provide the six components of the external wrench applied to each foot, usually expressed in their own local reference system. When solving the inverse dynamics problem, these measured reactions have to be included in the vector of known generalized forces $\mathbf{Q}_0$, so they must be properly transformed according to the location and orientation of the force plates.

Having all the required data available, the generalized motor torques $\mathbf{Q}_e$ can be obtained by simply isolating them from Eq. (8). However, taking into account how the independent coordinates in $\mathbf{z}$ have been defined, it is not easy to directly interpret the provided results, so it is customary to transform them into a vector of applied forces and torques $\mathbf{T}$, which contains an equivalent set of motor efforts, but expressed in a more meaningful form. This is achieved by using a position–dependent input matrix $\mathbf{B}$ [18], such that:

$$\mathbf{Q}_e = \mathbf{B}^{\mathrm{T}}\mathbf{T} \tag{9}$$

The vector $\mathbf{T}$ must have the same dimension as $\mathbf{Q}_e$, so the input matrix should be invertible. Therefore, the external efforts can be obtained by solving the following linear system:

$$\mathbf{B}^{\mathrm{T}}\mathbf{T} = \mathbf{M}\ddot{\mathbf{z}} - \mathbf{Q}_0 \tag{10}$$

2.2 EKF-D: state observer with embedded multibody dynamics

2.2.1 Plant model

The EKF-D is based on the same dynamic model used in Sect. 2.1.3 for stating the inverse dynamics problem. In order to express the model in the state–space form required for a

Kalman filter, the first step is isolating the accelerations from Eq. (10):

$$\ddot{\mathbf{z}} = \mathbf{M}^{-1}\left(\mathbf{Q}_0 + \mathbf{B}^{\mathsf{T}}\mathbf{T}\right) \tag{11}$$

Then, the state vector $\mathbf{x}$ is augmented by adding the external efforts vector $\mathbf{T}$, which now includes the motor torques and the external reactions:

$$\mathbf{x} = \begin{bmatrix} \mathbf{z} \\ \dot{\mathbf{z}} \\ \mathbf{T} \end{bmatrix} \tag{12}$$

This state vector allows to rewrite the equations of motion as a first–order ODE system, to which an additive noise that accounts for the uncertainties in the external efforts is added:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \tilde{\mathbf{w}}) = \begin{bmatrix} \dot{\mathbf{z}} \\ \mathbf{M}^{-1}\left(\mathbf{Q}_0 + \mathbf{B}^{\mathsf{T}}\mathbf{T}\right) \\ \mathbf{0} \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{I} \end{bmatrix} \tilde{\mathbf{w}} \tag{13}$$

The time derivative of the motor efforts $\dot{\mathbf{T}}$ is modeled as a vector of zero–mean Gaussian noise $\tilde{\mathbf{w}}$, in such a way that the filter behavior can be tuned by adjusting the characteristics of its components.

At this point, there are two possible approaches for including the force plate measurements into the model. The simplest way is to consider them as inputs to the system, so their raw values can be added directly to $\mathbf{Q}_0$ through an appropriate input matrix, thus leaving the motor torques as the only states constituting $\mathbf{T}$. Another possibility is to consider the measurements as sensors. However, this poses a problem, since the observation function $\mathbf{h}(\mathbf{x})$ must be able to compute the sensor measurements from the states $\mathbf{x}$, and that is not possible when there are redundant measurements. During the double–support phase of gait, the 12 components of the ground reactions acting at both feet can be mapped into the space of the independent coordinates $\mathbf{z}$, but the reverse operation is indeterminate [34]. A solution to overcome this problem is to include the n_r ground reactions as part of the state vector, in such a way that $\mathbf{T}$ becomes of dimension $n_z - 6 + n_r$ (the six degrees of freedom of the root body are not actuated, hence the -6). Since this approach allows for finer tuning of the filter, by taking into account the noise present in the force plate measurements, it will be the method used in this state observer.

In order to implement an extended Kalman filter, the system has to be linearized at every time step about its current state $\mathbf{x}_k$. Since the ODE system displayed in Eq. (13) does not have any input, and there is no explicit dependency of time, the linearized equations are reduced to [26]:

$$\delta\dot{\mathbf{x}} = \mathbf{F}_k \delta\mathbf{x} + \mathbf{G}\tilde{\mathbf{w}} \tag{14}$$

where $\delta\mathbf{x}$ is the state perturbation $\mathbf{x} - \mathbf{x}_k$, and $\mathbf{F}_k$ and $\mathbf{G}$ are the sensitivity and noise gain matrices respectively, which are defined as follows [26]:

$$\mathbf{F}_k = \left.\frac{\partial \mathbf{f}(\mathbf{x}, \mathbf{0})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}_k} = \left.\begin{bmatrix} \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \ddot{\mathbf{z}}_{\mathbf{z}} & \ddot{\mathbf{z}}_{\dot{\mathbf{z}}} & \ddot{\mathbf{z}}_{\mathbf{T}} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}\right|_{\mathbf{x}=\mathbf{x}_k} \qquad \mathbf{G} = \left.\frac{\partial \mathbf{f}(\mathbf{x}_k, \tilde{\mathbf{w}})}{\partial \tilde{\mathbf{w}}}\right|_{\tilde{\mathbf{w}}=\mathbf{0}} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{I} \end{bmatrix} \tag{15}$$

The terms $\ddot{\mathbf{z}}_{\mathbf{z}}$, $\ddot{\mathbf{z}}_{\dot{\mathbf{z}}}$ and $\ddot{\mathbf{z}}_{\mathbf{T}}$ are the Jacobian matrices of the accelerations from Eq. (11) with respect to the positions $\mathbf{z}$, velocities $\dot{\mathbf{z}}$ and applied efforts $\mathbf{T}$, respectively. The evaluation of these terms, especially the first two, can become very complex and computationally expensive, depending on the chosen dynamic formulation. This work uses the method to assemble the equations of motion described in [18], because that enables to obtain the derivatives in a relatively simple way by applying the chain differentiation rule, similarly to what is done in [21] or [22]. However, the computational performance of this implementation is far from optimal, and can be vastly improved by using, for instance, a semi–recursive formulation that better exploits the open–loop topology of the system [35]. The efficient application of the chain differentiation rule to the derivatives of such formulation, while still taking advantage of recursion, is described in [36], where the semi–recursive method is used for sensitivity analysis.

In order to improve the computational efficiency of the filter without having to thoroughly optimize the calculation of $\mathbf{F}_k$, a simplified version of this matrix is tested in this work. The simplified version is obtained by neglecting the $\ddot{\mathbf{z}}_{\mathbf{z}}$ and $\ddot{\mathbf{z}}_{\dot{\mathbf{z}}}$ derivatives, and retaining only the $\ddot{\mathbf{z}}_{\mathbf{T}}$ term, which is required to maintain the observability of the system. The results of comparing both alternatives show that the simplified model can greatly improve the computational efficiency and, in many cases, even without incurring a significant penalty in the accuracy of the torque estimations, although there are use cases where the complete model may become necessary.

The Kalman filter predictor step, i.e. the progression of the system between two consecutive time steps k and $k+1$, is more complex in this case, since the plant is no longer linear. When the system is nonlinear, the predictor requires to integrate the ODE system over a time step, by using a standard numerical integrator. In this work, the influence of the numerical integrator in the results is studied by comparing three different integrators, detailed in Sect. 2.2.2: forward Euler, Heun's method, and the implicit trapezoidal rule.

In order to compute the updated state error covariance matrix $\boldsymbol{\mathcal{P}}_{k+1}$ and the Kalman gain matrix $\boldsymbol{\mathcal{K}}_{k+1}$ at every time step, it is necessary to determine the state transition matrix $\boldsymbol{\Phi}_k$ of the linearized system. This matrix relates the current state perturbation $\delta\mathbf{x}_k$ to its value at the next time step $k+1$, when the system progresses in the absence of inputs and plant noise. Therefore, it represents the sensitivity of the homogeneous solution $\delta\mathbf{x}^H_{k+1}$ resulting from integrating Eq. (14) over an entire time step, with respect to the current value $\delta\mathbf{x}_k$:

$$\delta\mathbf{x}^H_{k+1} = \boldsymbol{\Phi}_k \delta\mathbf{x}_k \tag{16}$$

For a linear time–invariant (LTI) system such as that defined by Eq. (14), the state transition matrix for a given time step length Δt can be obtained as the following matrix exponential [26, 27]:

$$\boldsymbol{\Phi}_k = e^{\mathbf{F}_k \Delta t} \tag{17}$$

which is itself defined as an infinite power series:

$$e^{\mathbf{F}\Delta t} = \mathbf{I} + \mathbf{F}\Delta t + \frac{(\mathbf{F}\Delta t)^2}{2!} + \frac{(\mathbf{F}\Delta t)^3}{3!} + \cdots \tag{18}$$

In some cases this power series converges rapidly, so the matrix exponential can be easily obtained by neglecting the higher–order terms. In other situations, however, this is not the case, so it becomes necessary to resort to alternative methods, generally based on a Padé approximant. In this work, three different alternatives are tested, in order to assess which one shows the best balance between efficiency and accuracy. The simplest approximation is obtained by truncating the power series at the first–order term, thus approximating the state propagation matrix $\boldsymbol{\Phi}$ as $\mathbf{I}+\mathbf{F}\Delta t$. The next alternative adds the second–order term of Eq. (18). Finally, in order to have a reference result, the last option uses the accurate scaling and squaring algorithm implemented in the `expm` function of Matlab [37].

Since the plant model is now being derived from a continuous nonlinear ODE system, the plant noise vector from Eq. (13) is no longer described by a set of variances. The continuous noise vector is characterized instead by a covariance matrix $\mathcal{Q}$ which, if the components of $\tilde{\mathbf{w}}$ are supposed to be independent, contains their respective power spectral densities (PSD) at its diagonal. In order to implement a discrete EKF, it becomes necessary to derive its discrete counterpart $\mathcal{Q}_k$ at every time step. The calculation of said matrix requires the evaluation of a complex convolution integral involving the state transition matrix $\boldsymbol{\Phi}$ [26]. However, it is possible to obtain an estimate by replacing $\boldsymbol{\Phi}$ by one of its approximations, and then performing the integration. This process is especially simple when the first–order approach $\boldsymbol{\Phi}\approx\mathbf{I}+\mathbf{F}\Delta t$ is considered, resulting in the following expression for $\mathcal{Q}_k$ [26]:

$$\mathcal{Q}_k=\mathcal{Q}'\Delta t+\left(\mathbf{F}_k\mathcal{Q}'+\mathcal{Q}'\mathbf{F}_k^{\mathsf{T}}\right)\frac{\Delta t^2}{2}+\mathbf{F}_k\mathcal{Q}'\mathbf{F}_k^{\mathsf{T}}\frac{\Delta t^3}{3} \tag{19}$$

where $\mathcal{Q}'$ is the covariance matrix of the continuous noise vector $\tilde{\mathbf{w}}$ (which has the same dimension as $\mathbf{T}$), projected into the space of system states:

$$\mathcal{Q}'=\mathbf{G}\mathcal{Q}\mathbf{G}^{\mathsf{T}} \tag{20}$$

When implementing Kalman filters, it is common practice to keep only the first term, which results in a constant covariance matrix (provided that $\mathbf{G}$ is also constant). However, during the tests carried out in this work, the filter did not behave well unless all three terms were taken into account.

In order to assess the validity of this expression, an accurate approximation will be obtained using the method proposed by Van Loan [38]. According to this method, for any given LTI system, one can define a matrix $\mathbf{V}_k$ such that, after multiplying it by Δt and computing the matrix exponential, provides both $\mathcal{Q}_k$ and $\boldsymbol{\Phi}_k$ in a single operation:

$$\mathbf{V}_k=\begin{bmatrix}-\mathbf{F}_k & \mathcal{Q}'\\ \mathbf{0} & \mathbf{F}_k^{\mathsf{T}}\end{bmatrix}\Longrightarrow e^{\mathbf{V}_k\Delta t}=\begin{bmatrix}\boldsymbol{\Psi} & \boldsymbol{\Phi}_k^{-1}\mathcal{Q}_k\\ \mathbf{0} & \boldsymbol{\Phi}_k^{\mathsf{T}}\end{bmatrix} \tag{21}$$

After the computation of the matrix exponential, $\mathcal{Q}_k$ can be directly extracted from the result, by premultiplying the upper right block by the lower right block, transposed. This method is very easy to implement, but, as will be shown in Sect. 3.2, it is much more computationally expensive than the previous approach.

As explained above, the elements of the diagonal of $\mathcal{Q}$ are the PSDs of the respective noise sources. For any given continuous white noise signal with constant power density S along the infinite frequency spectrum, the variance of its discretization at a finite sampling frequency f_s is the area below the spectrum across the bandwidth between $-f_s/2$ and $f_s/2$:

$$\sigma^2=Sf_s=\frac{S}{\Delta t} \tag{22}$$

Therefore, a variance σ^2 can be converted into a PSD by multiplying it by the inverse of the sampling frequency Δt. In order to make the definition of the plant noise characteristics more consistent with the EKF-K, the parameters of the plant noise in the EKF-D will be expressed as variances, so they have to be multiplied by Δt when assembling the continuous plant noise covariance matrix $\mathcal{Q}$.

It should be noted that, when using the simplified version of $\mathbf{F}_k$, this affects both the subsequent calculation of $\mathbf{\Phi}_k$ and $\mathcal{Q}_k$, as described above.

2.2.2 Time integration

As mentioned in Sect. 2.2.1, the Kalman predictor requires the time integration of the differential equations of the system. Since the equations of motion are already expressed as a standard ODE system, one can choose any of the various integrators found in the literature for this task. The simplest method considered here is the explicit forward Euler integrator:

$$\begin{aligned} \mathbf{z}_{k+1} &= \mathbf{z}_k + \dot{\mathbf{z}}_k \Delta t \\ \dot{\mathbf{z}}_{k+1} &= \dot{\mathbf{z}}_k + \ddot{\mathbf{z}}_k \Delta t \end{aligned} \tag{23}$$

where the accelerations at the current time step $\ddot{\mathbf{z}}_k$ are obtained directly from Eq. (11). This integrator is not very accurate for large time steps, and tends to accumulate error over time, but the Kalman filter includes a set of position sensors that prevent the system from drifting, so it can provide reasonably good results in this particular application.

Heun's method is another explicit integrator, which provides a higher level of accuracy without severely penalizing the computational effort. In Heun's integrator, the derivatives used to update positions and velocities are obtained through an explicit trapezoidal rule, i.e. they are computed as the average between their value at the current time step, and an estimation at $k+1$:

$$\begin{aligned} \mathbf{z}_{k+1} &= \mathbf{z}_k + \frac{\Delta t}{2}\left(\dot{\mathbf{z}}_k + \hat{\dot{\mathbf{z}}}_{k+1}\right) \\ \dot{\mathbf{z}}_{k+1} &= \dot{\mathbf{z}}_k + \frac{\Delta t}{2}\left(\ddot{\mathbf{z}}_k + \hat{\ddot{\mathbf{z}}}_{k+1}\right) \end{aligned} \tag{24}$$

The estimation of the derivatives at $k+1$ is obtained by taking a single forward–Euler step:

$$\begin{aligned} \hat{\mathbf{z}}_{k+1} &= \mathbf{z}_k + \dot{\mathbf{z}}_k \Delta t \\ \hat{\dot{\mathbf{z}}}_{k+1} &= \dot{\mathbf{z}}_k + \ddot{\mathbf{z}}_k \Delta t \end{aligned} \tag{25}$$

The estimated accelerations $\hat{\ddot{\mathbf{z}}}_{k+1}$ appearing in Eq. (24) must be again obtained from Eq. (11), so this integrator requires two evaluations of the accelerations per time step.

The implicit trapezoidal rule is similar to Heun's method, but using the actual values of velocities and accelerations at the next time step, instead of a forward–Euler estimation:

$$\begin{aligned} \mathbf{z}_{k+1} &= \mathbf{z}_k + \frac{\Delta t}{2}\left(\dot{\mathbf{z}}_k + \dot{\mathbf{z}}_{k+1}\right) \\ \dot{\mathbf{z}}_{k+1} &= \dot{\mathbf{z}}_k + \frac{\Delta t}{2}\left(\ddot{\mathbf{z}}_k + \ddot{\mathbf{z}}_{k+1}\right) \end{aligned} \tag{26}$$

These equations can not be solved explicitly, and require the use of numerical methods such as fixed–point or Newton–Raphson iterations. For this reason, this integrator can become computationally expensive, but it is interesting to test whether resorting to such an integrator in the framework of a Kalman filter that already keeps the system under control using position sensors adds any value.

2.2.3 Observation model

In the EKF-D there are two sets of sensors. Along with the optical markers $\mathbf{m}$, already present in the EKF-K, the force plate measurements $\mathbf{p}$ are now added to the measurements vector. The segment of $\mathbf{h}(\mathbf{x})$ and the block of $\mathbf{H}(\mathbf{x})$ involving the optical markers remain exactly as described in Sect. 2.1.2. The force plate measurements are just directly added to the observation vector, since they are already part of the system states, in such a way that the block of the Jacobian matrix $\mathbf{H}(\mathbf{x})$ associated to the force plates, i.e. the rows corresponding to their positions within the observation vector, and the columns corresponding to their allocation within $\mathbf{x}$, is an $n_r \times n_r$ identity matrix. Assuming that the observation vector has the markers at the top and the force plates at the bottom, and that the state vector has the structure shown in Eq. (12), with the external efforts $\mathbf{T}$ assembled such that the force plate reactions are placed at the end, the Jacobian of the observation function has the following structure:

$$\mathbf{H}(\mathbf{x}) = \begin{bmatrix} \dfrac{\partial \mathbf{m}}{\partial \mathbf{z}} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix} \tag{27}$$

The sensor noise covariance matrix $\boldsymbol{\mathcal{R}}$ is also augmented by adding n_r rows and n_r columns. The n_r components of the reactions at both plates are treated also as independent, so the block associated to the force plates is a diagonal matrix containing the variances of the corresponding sensor noises.

3 Double pendulum

In human motion capture, it is nearly impossible to verify the accuracy of the results, since there are few methods that allow to determine the exact configuration of the skeleton during movement, and the existing ones, such as fluoroscopy, are very complex and not commonly available. Therefore, it is useful to first test the algorithms using a simple benchmark problem with a known solution. In this work, a simple 2D double pendulum model will be used for that purpose. In the virtual experiment, the parameters of the model and the movement are specified in advance. Then, artificial noise is added to the computed sensor signals, and the resulting synthetic sensor data is fed to the observers, in such a way that the quality of the estimation can be properly compared to the known exact solution.

3.1 Multibody model and experiment definition

The model, as displayed in Fig. 2, is designed to be simple, while keeping as many features of human movement as possible. The whole system has four degrees of freedom, being their corresponding independent coordinates the location of the upper point, $\mathbf{r}_0$, and the two absolute angles θ_1 and θ_2 between the pendulum segments and the vertical direction:

$$\mathbf{z} = \begin{bmatrix} x_0 & y_0 & \theta_1 & \theta_2 \end{bmatrix}^{\mathrm{T}} \tag{28}$$

The synthetic movement is designed to mimic a squat exercise, repeated in a loop: the lower point (2) stays at the global origin, in contact with the ground, while the upper one (0) moves vertically up and down, in such a way that the joint between both segments (1) behaves as

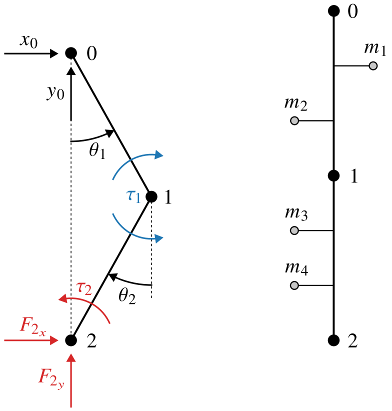

Fig. 2 Two–dimensional double pendulum model, displaying the model coordinates and acting forces and torques (left), as well as the placement of the optical markers (right)

a flexing and extending knee. Such a motion is generated by driving the model coordinates with the following functions:

$$\begin{aligned} x_0(t) &= 0 \\ y_0(t) &= 2L\cos(\Theta\sin\omega t) \\ \theta_1(t) &= \theta_2(t) = \Theta\sin\omega t \end{aligned} \tag{29}$$

where L is the length of each pendulum segment, and Θ and ω are two parameters used to set the amplitude and frequency of the exercise repetitions. Once the movement has been generated, it is possible to compute the virtual sensor data. First, the trajectories of the markers are determined by considering them as rigidly attached to the pendulum segments, as displayed on the right side of Fig. 2. Then, the inverse dynamics solution provides the corresponding external efforts vector $\mathbf{T}$ which, as shown on the left side of Fig. 2, comprises a motor torque at joint 1 and a reaction force and torque acting at point 2:

$$\mathbf{T} = \begin{bmatrix} \tau_1 & F_{2_x} & F_{2_y} & \tau_2 \end{bmatrix}^{\mathrm{T}} \tag{30}$$

The first component, τ_1, is interpreted as a knee joint torque, which is the unknown input that is going to be estimated by the state observers. The remaining three components are regarded as ground reactions, so their values will be treated as force plate measurements. Therefore, the measurement vector includes the trajectories of the four markers, the ground reaction force, and the ground reaction moment as follows:

$$\mathbf{h}(\mathbf{x}) = \begin{bmatrix} m_{1_x} & m_{1_y} & m_{2_x} & m_{2_y} & m_{3_x} & m_{3_y} & m_{4_x} & m_{4_y} & F_{2_x} & F_{2_y} & \tau_2 \end{bmatrix}^{\mathrm{T}} \tag{31}$$

To simulate real sensors, these virtual measurements are perturbed by incorporating artificial noise. The ground reactions are added zero–mean white noise with a variance σ_r^2, to account for the measurement noise introduced by the force plates. The markers, on the other hand, are affected by two different noise sources. Firstly, another low–amplitude white noise signal with a small variance σ_c^2 is used to simulate the measurement noise of the camera system. On top of that, a reduced bandwidth random signal with larger amplitude accounts for the skin motion artifact. This disturbance is initially generated as a zero–mean random sequence with Gaussian distribution, but is then low–pass filtered to reduce its frequency content, and finally scaled to match the target variance σ_m^2.

Table 1 Pendulum simulation parameters

Physical constants	
Upper segment mass	50 kg
Lower segment mass	25 kg
Segment length (L)	0.85 m
Movement	
Simulation time	4 s
Amplitude (Θ)	0.9 rad
Frequency (ω)	π rad/s
Noise characteristics	
Camera noise (σ_c)	0.02 mm
Skin motion artifact noise (σ_m)	10 mm
Skin motion artifact cutoff frequency	1 Hz
Reaction force noise (σ_r)	0.3 N (or Nm)

Table 2 Parameters of the Kalman filters

EKF-K	
Plant noise (σ_a)	300 m/s^2 (or rad/s^2)
Marker noise (σ_s)	10 mm
EKF-D	
Plant noise (σ_t)	2 kN (or kNm)
Marker noise (σ_s)	10 mm
Force plate noise (σ_p)	0.3 N (or Nm)

The virtual measurements are then fed to the different Kalman filters to reconstruct the motion, and the estimation error of the unknown input τ_1, whose exact value has been previously determined by inverse dynamics, is used as a measure of the observer accuracy.

3.2 Tests and results

The parameters used for generating the motion are summarized in Table 1. The values of σ_c and σ_r were obtained by sampling actual idle sensor noise at a 100 Hz rate and computing the standard deviation of the signal. The variance and cutoff frequency of the skin motion artifact were established by analyzing the relative movement of the measured markers with respect to their nominal positions, using data from previous motion capture sessions.

The variances of the sensors in the Kalman filters were assigned the same values previously used for generating the artificial noise. With these variances fixed, the plant noise characteristics were manually tuned for each filter, in order to get the best estimation possible. The resulting parameters are displayed in Table 2.

Section 2.2 presents many alternatives and simplifications for implementing the EKF-D. In order to compare all of them, multiple simulations using all possible combinations were carried out. Firstly, the three numerical integrators described in Sect. 2.2.2 were evaluated. For each integrator, all the different combinations of methods for calculating the transition matrix $\mathbf{\Phi}_k$ (first–order and second–order expansions of the power series, and the complete matrix exponential) and plant noise covariance matrix $\boldsymbol{Q}_k$ (first–order approximation and Van Loan's method) were tested. And all the simulations were run, in turn, using both the

Table 3 Double pendulum results for all combinations of numerical integrator, state propagation matrix, plant noise covariance matrix, and linearized model

Integrator	$\mathbf{\Phi}_k$ approx.	$\mathcal{Q}_k$ approx.	$\mathbf{F}_k$ version	RMS (% wt)	RT ratio
Forward Euler	1storder	1storder	Simplified	2.62 ± 0.38	199.9
			Full	2.57 ± 0.36	157.9
	2ndorder	1storder	Simplified	2.39 ± 0.45	194.8
			Full	2.29 ± 0.36	153.9
	Exponential	1storder	Simplified	2.39 ± 0.45	135.4
			Full	2.30 ± 0.38	106.8
		Van Loan	Simplified	2.29 ± 0.50	63.1
			Full	2.17 ± 0.38	53.4
Heun's method	1storder	1storder	Simplified	2.36 ± 0.37	170.6
			Full	2.38 ± 0.36	145.4
	2ndorder	1storder	Simplified	**2.09 ± 0.39**	**177.6**
			Full	2.12 ± 0.37	143.4
	Exponential	1storder	Simplified	2.09 ± 0.39	129.4
			Full	2.10 ± 0.37	106.2
		Van Loan	Simplified	1.96 ± 0.40	63.1
			Full	1.97 ± 0.37	52.9
Trapezoidal rule	1storder	1storder	Simplified	2.36 ± 0.37	120.4
			Full	2.38 ± 0.36	102.2
	2ndorder	1storder	Simplified	2.09 ± 0.38	118.6
			Full	2.12 ± 0.37	101.7
	Exponential	1storder	Simplified	2.09 ± 0.38	93.1
			Full	2.09 ± 0.37	80.9
		Van Loan	Simplified	1.96 ± 0.40	52.8
			Full	1.97 ± 0.37	45.4

complete $\mathbf{F}_k$ matrix and its simplified version with $\ddot{\mathbf{z}}_{\mathbf{z}}$ and $\ddot{\mathbf{z}}_{\dot{\mathbf{z}}}$ set to zero. Some combinations that do not make sense are omitted, since Van Loan's method already obtains the propagation matrix as the complete exponential of $\mathbf{F}_k \Delta t$.

The whole testing process involves 24 different implementations, whose results are displayed in Table 3. The "RMS" column refers to the numeric value of the RMS error of the estimated joint torque, normalized as a percentage of the total weight of the model, and the "RT ratio" column shows the computational efficiency, expressed as the ratio between real time and CPU time. The accuracy of the estimated torques varies depending on the specific random noise profile applied to the virtual sensors so, to make the results more consistent, the set of 24 simulations was run 100 times, each one using a newly generated sensor dataset. The error column shows the average and standard deviation of the results, whereas the efficiency column displays only the average, since the CPU time does not significantly vary among datasets. All the tests were implemented in Matlab, and were run on an AMD Ryzen 9 5950X CPU.

To facilitate the interpretation of these results, they are summarized in Fig. 3, where each bar represents the average result of all the simulations run with the corresponding alternative, so it is easier to infer the effect of each individual implementation option on both the accuracy and computational efficiency. The two upper graphs compare the results obtained

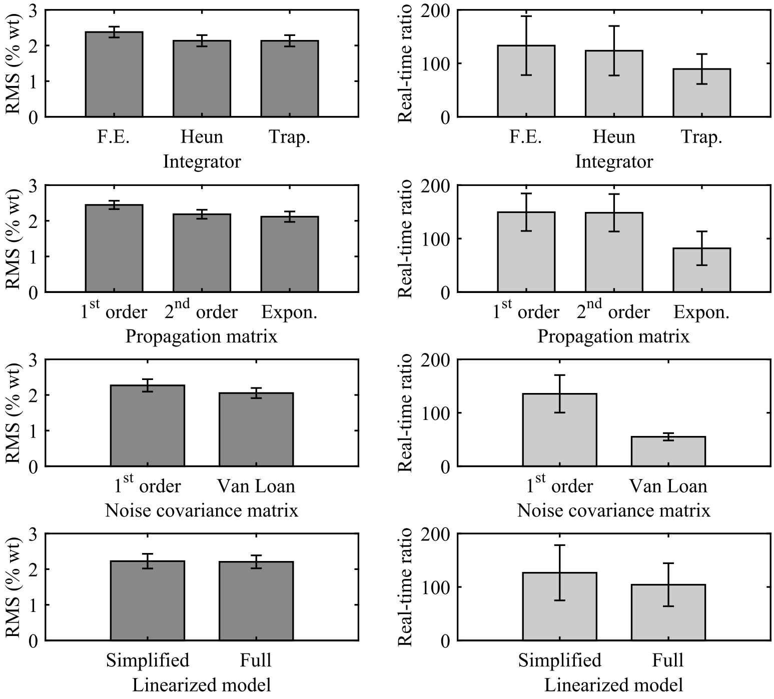

Fig. 3 Averaged effects of different numerical integrators and simplifications on the accuracy (left) and efficiency (right) of the EKF-D

for the three different numerical integrators. As expected, the forward Euler integrator obtains the lowest accuracy, but it is the fastest alternative. However, Heun's method shows an improvement in the results, without having a large impact on the real–time performance. The trapezoidal rule has a much higher computational cost, while having almost no effect on the accuracy with respect to Heun's method. The next comparison shows that computing the state propagation matrix $\mathbf{\Phi}_k$ using a second–order approximation is probably the best option, since it provides essentially the same results as using the exact matrix exponential, while keeping its CPU performance close to that obtained with the basic first–order estimate. Regarding the plant noise covariance matrix $\mathcal{Q}_k$, it is clear that the higher computational cost of using Van Loan's approximation is not worth it for real–time applications, unless the model is sufficiently small. Finally, using the simplified linearized model does not have a significant impact on the results, while providing an increased efficiency.

According to these results, it is concluded that the best compromise is to use Heun's method for the predictor, while computing the state propagation matrix by truncating the power series at the second–order term, then using the first–order approximation of the plant noise covariance matrix, and computing both of these terms with the simplified version of the linearized system. The results corresponding to this combination are highlighted in Table 3, where it can be observed that it has a good balance between efficiency and accuracy.

Once the optimal implementation for the EKF-D has been determined, the next step is to compare it to the existing EKF-K. As an example, the estimated torques obtained from one of the simulations are displayed in Fig. 4. The rapid variations shown by both Kalman filters at the beginning occur because the initial conditions at acceleration/force level are unknown, so they are set to zero, and, therefore, the filters require a few iterations to converge.

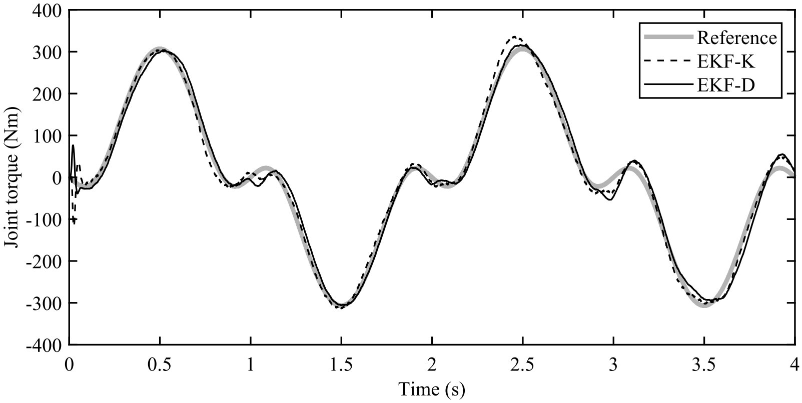

Fig. 4 Estimated motor torque at joint 1, comparing EKF-K and the selected implementation of EKF-D

To make the accuracy results more consistent, these initial frames were excluded from the computation of the estimation errors.

In terms of efficiency, the EKF-K achieves a real–time ratio of about 340, whereas the EKF-D, with the selected configuration, is about half as fast, with a ratio close to 180. However, in larger systems this difference becomes more relevant, as will be shown in Sect. 4.

Regarding accuracy, both filters provide essentially the same results for the estimated positions $\mathbf{z}$, which is something to be expected, since the spatial configuration is completely driven by absolute position sensors. However, there are substantial discrepancies in the obtained results for the joint torque τ_1, due to the distinct nature of the observers. These differences become apparent when testing the algorithms at different movement speeds. By looking exclusively at the RMS error, it may appear that the EKF-D only outperforms the EKF-K at low speeds since, in fast movements, the latter shows smaller errors. However, if one looks at the torque curves closely, it can be observed that the shape of the estimation provided by the EKF-D is closer to the reference, but it is delayed in time. This delay is the fundamental cause of the larger RMS errors presented by the EKF-D in fast movements since, as the curves become steeper, small horizontal offsets translate into larger vertical gaps.

The estimation delay affects the accelerations in the EKF-K as well as the unmeasured torque in the EKF-D, since they are both magnitudes that are obtained in an indirect manner, either from position sensors only or by adding force and torque sensors elsewhere in the system. The difference between the results of the two state observers lies in how this time shift influences their respective torque estimates. Whereas in the EKF-D the torque itself is a state directly affected by the delay, in the EKF-K the lag appears in the accelerations, so that, when calculating the torque by inverse dynamics, current force measurements are being mixed with past acceleration estimates [18]. For this reason, the torque provided by the EKF-K does not have an appreciable delay although, in return, the curve shows more distortions, especially in fast movements.

To quantify this effect, both filters were executed at multiple movement speeds, by changing the value of ω in Eq. (29). Again, in order to account for the variability introduced by the artificial sensor noise, all simulations were run 100 times with different datasets. The errors obtained by the EKF-D were calculated twice, first for the raw results, and then by

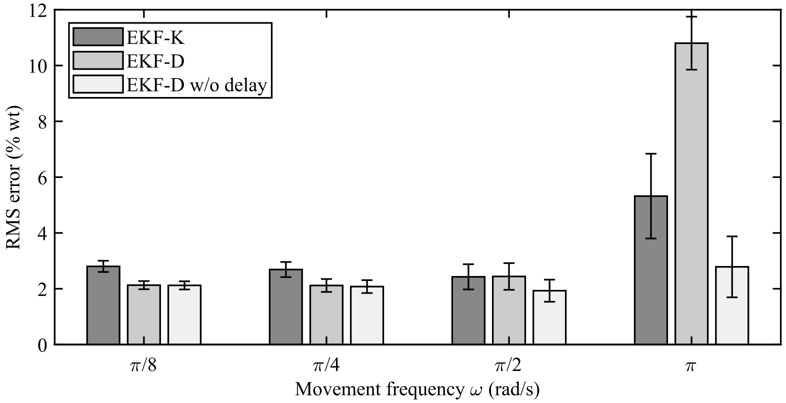

Fig. 5 Comparison of EKF-K and EKF-D accuracy at different movement speeds

shifting the curve in time by 14 ms[1] to compensate for the delay. The results are displayed in Fig. 5. It can be observed that the EKF-D performs consistently better at low speeds and, although the EKF-K provides a lower RMS error at faster movements, the EKF-D matches the reference more accurately when the delay is compensated.

The robustness of both algorithms to marker occlusions was also tested, but their performance was about the same in this respect. When markers are lost, in either method the model evolves similarly using the propagation model. If that situation is prolonged for a few time steps, the state starts to drift, so that when markers become available again, the error is suddenly eliminated. This can lead to abrupt variations in the estimated torques, which are generally smaller in the EKF-D, but not too significantly.

In order to achieve more insightful conclusions, both filters are tested in a full–body motion capture in Sect. 4. In the virtual experiment described in this Section, the skin motion artifact has been modeled as zero–mean random noise, which is not a very realistic approximation, since, in actual motion capture, the markers are subject to the elasticity and inertia of the soft tissues. Testing both methods in a real capture does not allow to compare the joint torques to a reference value, but still can provide insights about other possible benefits of adding a dynamic model and force sensors to the observer.

4 3D full–body human model

4.1 Multibody model

The three–dimensional multibody model of the human body used here is taken from the previous work [18]. The model comprises 18 anatomical segments, as shown in Fig. 6: pelvis, trunk, neck, head, arms, forearms, hands, thighs, shanks, feet, and toes, all connected through spherical joints, except for the toes, which use revolute joints, and the base of the neck, which is connected to the trunk by means of a universal joint. Therefore, the model has a total of 52 degrees of freedom. The vector of external efforts **T** has a dimension of

---

[1] This value was obtained consistently by running the filter with exact (noise–free) virtual sensors, for varying values of ω and Δt, as long as the PSD of the plant noise remained the same. Increasing the PSD reduces the delay, but the estimation gets noisier.

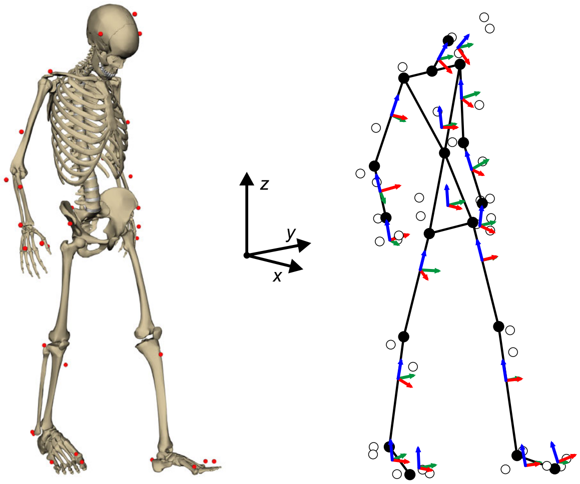

Fig. 6 Multibody model showing joints and optical markers

Table 4 Parameters of the Kalman filters

EKF-K	
Plant noise (σ_a)	100 m/s^2 (or rad/s^2)
Marker noise (σ_s)	10 mm
EKF-D	
Plant noise: forces (σ_f)	1 kN
Plant noise: moments (σ_τ)	50 Nm
Marker noise (σ_s)	20 mm
Force plate noise (σ_p)	40 N (or Nm)

52 in the EKF-K algorithm, since it includes a residual wrench at the pelvis, along with the 46 joint torques. In the EKF-D method this vector has 58 components, because it does not include the residual wrench at the pelvis, but adds the 12 components of the foot–ground reactions as system states. The procedure for scaling the model and the details about the equations of motion are thoroughly described in the previous paper [18], so they will be omitted here.

4.2 Experimental results

The experiments were carried out by tracking a set of 36 optical markers on a subject. The markers are tracked using a set of 18 Flex 3 infrared cameras (OptiTrack, Corvallis OR, USA). These cameras have a resolution of 640x480 pixels, with a refresh rate of 100 Hz. In order to measure the ground reactions, two AccuGait force plates (AMTI, Watertown MA, USA), also recording at 100 Hz, are embedded into a ground platform.

The variances of the Kalman filters used for the full–body motion capture test are summarized in Table 4. In this case, a finer tuning has been required for the EKF-D algorithm: instead of using a single variance for all the external efforts, the filter needed different values for forces and moments in order to work successfully.

The experiment consists of a subject performing a vertical jump, as shown in Fig. 7. Figure 8 displays the estimated flexion–extension moments at the ankles, knees and hips (normalized to the body weight). The results of both state observers are very similar at the

Fig. 7 Vertical jump sequence: squat for impulse, maximum height, controlled landing

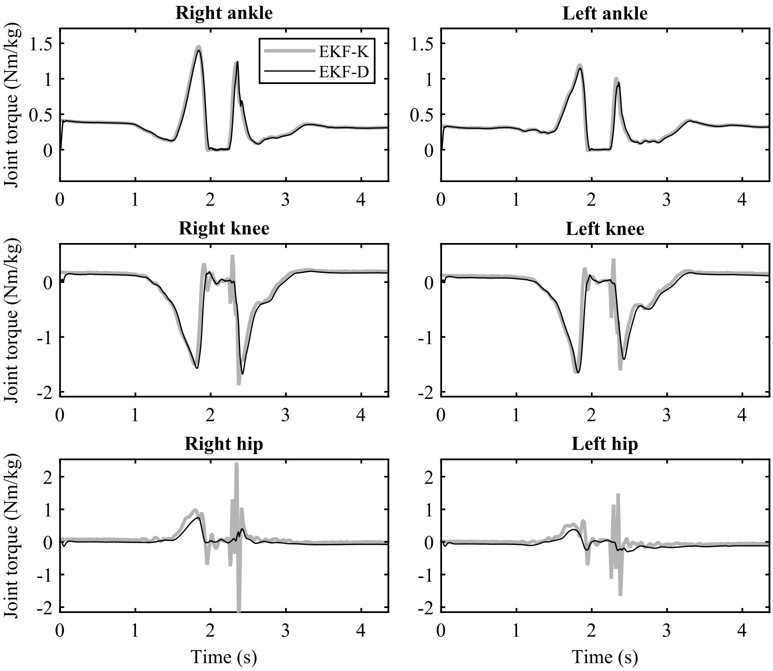

Fig. 8 Normalized flexion–extension torques at the lower limbs during a vertical jump

ankle joints, since the feet have a low mass and, therefore, the ankle torques are mostly dominated by the ground reactions. However, the agreement deteriorates at the knees, and gets even worse at the hips. In this case, unlike in the simulated experiment described in Sect. 3, there is no reference solution to assess which estimation is better. However, it is possible that the large torque oscillations shown by the EKF-K method at the hip joints, especially at the

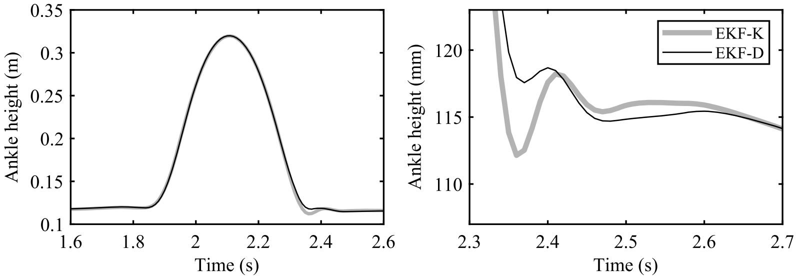

Fig. 9 Vertical motion of the right ankle joint during a vertical jump

moment of landing (about 2.35 s), derive from spurious accelerations due to the bouncing of the markers at the moment of impact. The markers are elastically attached to the body segments through soft tissues, so they can have an overshooting oscillation after the feet hit the ground. The EKF-K algorithm assumes that the rigid bodies follow the motion of the markers, so this oscillation is transmitted entirely to the underlying skeleton. Conversely, the EKF-D method does not rely on the optical markers as its only sensors, so the ground reactions are probably counteracting this overshooting effect. To illustrate this issue, Fig. 9 shows the time history of the vertical coordinate of the right ankle during the same experiment. On the left side, it can be appreciated that the trajectories are essentially equal, except at the moment of impact, when the EKF-K estimation shows a significantly higher oscillation. Therefore, it is possible that embedding a multibody model in the EKF may effectively improve the estimation of motor torques, albeit at the cost of a higher computational effort.

In its current Matlab implementation, the full–body EKF-D is much slower than the EKF-K. Whereas the Matlab version of the EKF-K achieves a real–time ratio of about 5.6, the filter with an embedded multibody model is not even capable or reaching real–time performance. Using the complete linearized model, the real–time ratio is only 0.22, which can be improved up to 0.39 by resorting to the simplified version. However, in the full–body motion capture case, using the approximated $\mathbf{F}_k$ matrix seems to be less robust, especially in the presence of strong impacts: both variants of the linearized system work well with regular ground contacts, but the vertical jump can not be successfully reconstructed by the EKF-D algorithm unless the complete $\mathbf{F}_k$ matrix is considered. Therefore, its performance needs to be vastly improved in order to make it feasible for real–time scenarios. Preliminary tests with the simplified model were carried out with a basic C++ translation of the Matlab code: using the Eigen linear algebra libraries [39], it can run the algorithm 6.5 times faster than real time. Assuming that the ratio obtained in Matlab between the efficiencies of the simplified and the complete linearized model remains about the same in a hypothetical C++ implementation, the EKF-D with the full model should still be able to run about 3.7 times faster than real time, without any optimization. Therefore, it is reasonable to assume that, by using a well–optimized multibody formulation, such as the semi–recursive one mentioned in Sect. 2.2.1, it will be possible to achieve real–time performance by an ample margin with the complete model.

5 Conclusions

This work introduces an extended Kalman filter for real–time marker–based human motion capture, aimed at improving the estimation of motor efforts by incorporating force sensors and a full multibody model. Embedding a multibody model into the state observer requires multiple complicated operations, such as linearizing the equations of motion, expressing them in state–space form, and computing the state transition and plant noise covariance matrices. In order to derive an optimal implementation of the filter, several options for efficiently calculating said terms were evaluated.

After finding the implementation with the best trade–off between estimation accuracy and computational performance, the algorithm was compared to a simple EKF with a kinematic propagation model and inverse dynamics. In a virtual experiment with known joint torques, the observer with a dynamic model consistently produced a better approximation of the unknown input, although the result was delayed in time, which may hinder its usability in some real–life scenarios. The exact lag cannot be determined precisely in the full–body experiment, but the delays with respect to the kinematic model were below 40 ms, which should be enough for most biofeedback applications [13], and can otherwise be further reduced, in exchange for some noise in the signal, by tuning the filter parameters.

The filter is intended to mitigate the soft–tissue problems associated with marker–based motion capture and, in view of the results, it effectively helps to reduce the problem to some extent. While for slow and smooth movements the problem is not very apparent, it is shown that, when there are impacts involved, the elasticity and inertia of soft tissues can produce erroneous estimations of the joint torques. The introduction of force sensors into the state observer, in such a way that the forces directly influence the progression of the dynamic system, appears to be useful to compensate the spurious accelerations introduced by the skin motion artifact.

On the other hand, it is evidenced that introducing the equations of motion within the observer incurs a significant computational cost. The efficiency results shown here are not representative, as the used implementation is mostly a proof of concept with a large margin for optimization, so it is assumed that, with the proper implementation, it is entirely feasible to use the algorithm in real time applications. However, no amount of optimization can make this method reach the efficiency of a kinematic filter with inverse dynamics. Therefore, the latter is possibly the best option when real–time performance is the main priority. This might happen, for instance, when more advanced features need to be added to the model, such as estimation of muscular efforts [18]. Another possible limitation of the method is its comparatively lower robustness: since the algorithm involves the time integration of the equations of motion, it is more prone to numerical problems, and also less permissive regarding the selection of the filter parameters.

To make the method usable in practical real–time applications, the dynamic model has to be properly re–implemented using an efficient dynamic formulation [35]. Apart from improving the efficiency, such a formulation, by using relative coordinates, would mostly eliminate the potential gimbal lock problems associated to absolute Euler angles. Regarding the estimation delay, the simplest way to address it is by adding velocity and acceleration sensors to the filter [22]. That would also improve the robustness against marker occlusions, and can be easily achieved by exploiting the Kalman filter structure to incorporate inertial measurement units, but not as orientation devices, but by directly using their raw sensor measurements.

Author contributions U.L. supervised the work, wrote the manuscript text and prepared the figures. M.P. and U.L. developed the algorithm and its software implementations. S.B. and F.M. performed the experiments and gathered the data. F.M. reviewed the manuscript. All authors have read and agreed to the submitted version of the manuscript.

Funding information Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature. Grant PID2022-140062OB-I00 funded by MICIU/AEI/10.13039/501100011033 and by ERDF/EU. Moreover, F.M. would like to acknowledge the support of the Galician Government and the Ferrol Industrial Campus by means of the postdoctoral research contract 2022/CP/048.

Data availability No datasets were generated or analysed during the current study.

Declarations

Ethical approval Ethical review and approval were waived for this study, due to the non–invasive and non–dangerous character of the experiments.

Consent to participate Written informed consent was obtained from all subjects involved in the study.

Competing interests The authors declare no competing interests.

References

1. Adesida, Y., Papi, E., McGregor, A.H.: Exploring the role of wearable technology in sport kinematics and kinetics: a systematic review. Sensors **19**(7), 1597 (2019)
2. Moro, M., Marchesi, G., Hesse, F., Odone, F., Casadio, M.: Markerless vs. marker–based gait analysis: a proof of concept study. Sensors **22**(5), 2011 (2022)
3. Benoit, D.L., Ramsey, D.K., Lamontagne, M., Xu, L., Wretenberg, P., Renström, P.: Effect of skin movement artifact on knee kinematics during gait and cutting motions measured in vivo. Gait Posture **24**(2), 152–164 (2006)
4. Peters, A., Galna, B., Sangeux, M., Morris, M., Baker, R.: Quantification of soft tissue artifact in lower limb human motion analysis: a systematic review. Gait Posture **31**(1), 1–8 (2010)
5. Silva, M.P.T., Ambrósio, J.A.C.: Kinematic data consistency in the inverse dynamic analysis of biomechanical systems. Multibody Syst. Dyn. **8**(2), 219–239 (2002)
6. Lu, T.W., O'Connor, J.J.: Bone position estimation from skin marker co–ordinates using global optimisation with joint constraints. J. Biomech. **32**(2), 129–134 (1999)
7. Reinbolt, J.A., Schutte, J.F., Fregly, B.J., Koh, B.I., Haftka, R.T., George, A.D., Mitchell, K.H.: Determination of patient–specific multi–joint kinematic models through two–level optimization. J. Biomech. **38**(3), 621–626 (2005)
8. Bonnet, V., Richard, V., Camomilla, V., Venture, G., Cappozzo, A., Dumas, R.: Joint kinematics estimation using a multi–body kinematics optimisation and an extended Kalman filter, and embedding a soft tissue artefact model. J. Biomech. **62**, 148–155 (2017)
9. Ackermann, M., Schiehlen, W.: Dynamic analysis of human gait disorder and metabolical cost estimation. Arch. Appl. Mech. **75**(10–12), 569–594 (2006)
10. Xiang, Y., Chung, H.J., Kim, J.H., Bhatt, R., Rahmatalla, S., Yang, J., Marler, T., Arora, J.S., Abdel-Malek, K.: Predictive dynamics: an optimization–based novel approach for human motion simulation. Struct. Multidiscip. Optim. **41**(3), 465–479 (2010)
11. Venne, A., Bailly, F., Charbonneau, E., Dowling-Medley, J., Begon, M.: Optimal estimation of complex aerial movements using dynamic optimisation. Sports Biomech. **22**(2), 300–315 (2022)
12. Nitschke, M., Marzilger, R., Leyendecker, S., Eskofier, B.M., Koelewijn, A.D.: Change the direction: 3D optimal control simulation by directly tracking marker and ground reaction force data. PeerJ **11**, e14852 (2023)
13. Ceglia, A., Bailly, F., Begon, M.: Moving horizon estimation of human kinematics and muscle forces. IEEE Robot. Autom. Lett. **8**(8), 5212–5219 (2023)

14. Cerveri, P., Pedotti, A., Ferrigno, G.: Robust recovery of human motion from video using Kalman filters and virtual humans. Hum. Mov. Sci. **22**(3), 377–404 (2003)
15. Halvorsen, K., Johnston, C., Back, W., Stokes, V., Lanshammar, H.: Tracking the motion of hidden segments using kinematic constraints and Kalman filtering. J. Biomech. Eng. **130**(1), 011012 (2008)
16. Senesh, M., Wolf, A.: Motion estimation using point cluster method and Kalman filter. J. Biomech. Eng. **131**(5), 051008 (2009)
17. Cuadrado, J., Michaud, F., Lugrís, U., Pérez Soto, M.: Using accelerometer data to tune the parameters of an extended Kalman filter for optical motion capture: preliminary application to gait analysis. Sensors **21**, 427 (2021)
18. Lugrís, U., Pérez-Soto, M., Michaud, F., Cuadrado, J.: Human motion capture, reconstruction, and musculoskeletal analysis in real time. Multibody Syst. Dyn. **60**(1), 3–25 (2023)
19. Cuadrado, J., Dopico, D., Pérez, J.A., Pastorino, R.: Automotive observers based on multibody models and the extended Kalman filter. Multibody Syst. Dyn. **27**, 3–19 (2012)
20. Sanjurjo, E., Naya, M.A., Blanco-Claraco, J.L., Torres-Moreno, J.L., Giménez-Fernández, A.: Accuracy and efficiency comparison of various nonlinear Kalman filters applied to multibody models. Nonlinear Dyn. **88**(3), 1935–1951 (2017)
21. Pyrhönen, L., Jaiswal, S., García-Agundez, A., Vallejo, D.G., Mikkola, A.: Linearization–based state–transition model for the discrete extended Kalman filter applied to multibody simulations. Multibody Syst. Dyn. **57**(1), 55–72 (2022)
22. Sanjurjo, E., Dopico, D., Luaces, A., Naya, M.Á.: State and force observers based on multibody models and the indirect Kalman filter. Mech. Syst. Signal Process. **106**, 210–228 (2018)
23. Adduci, R., Vermaut, M., Naets, F., Croes, J., Desmet, W.: A discrete–time extended Kalman filter approach tailored for multibody models: state–input estimation. Sensors **21**(13), 4495 (2021)
24. Naya, M.A., Sanjurjo, E., Rodríguez, A.J., Cuadrado, J.: Kalman filters based on multibody models: linking simulation and real world. A comprehensive review. Multibody Syst. Dyn. **58**(3–4), 479–521 (2023)
25. Bar-Shalom, Y., Li, X.R., Kirubarajan, T.: Estimation with Applications to Tracking and Navigation: Theory, Algorithms and Software. Wiley, New York (2004)
26. Gibbs, B.P.: Advanced Kalman Filtering, Least–Squares and Modeling: A Practical Handbook. Wiley, New York (2011)
27. Grewal, M.S., Andrews, A.P.: Kalman Filtering: Theory and Practice with MATLAB. Wiley, Hoboken (2015)
28. Simon, D.: Optimal State Estimation: Kalman, H_∞, and Nonlinear Approaches. Wiley, New York (2006)
29. Simon, D.: Kalman filtering with state constraints: a survey of linear and nonlinear algorithms. IET Control Theory Appl. **4**(8), 1303–1318 (2010)
30. Sabatini, A.M.: Quaternion–based extended Kalman filter for determining orientation by inertial and magnetic sensing. IEEE Trans. Biomed. Eng. **53**(7), 1346–1356 (2006)
31. Zanetti, R., Majji, M., Bishop, R.H., Mortari, D.: Norm–constrained Kalman filtering. J. Guid. Control Dyn. **32**(5), 1458–1465 (2009)
32. Winter, D.A.: Biomechanics and Motor Control of Human Movement. Wiley, Hoboken (2009)
33. Dumas, R., Chèze, L., Verriest, J.P.: Adjustments to McConville et al. and Young et al. body segment inertial parameters. J. Biomech. **40**(3), 543–553 (2007)
34. Lugrís, U., Carlín, J., Pàmies-Vilà, R., Font-Llagunes, J.M., Cuadrado, J.: Solution methods for the double–support indeterminacy in human gait. Multibody Syst. Dyn. **30**(3), 247–263 (2013)
35. Dopico Dopico, D., López Varela, Á., Luaces Fernández, A.: Augmented Lagrangian index-3 semi-recursive formulations with projections. Multibody Syst. Dyn. **52**(4), 377–405 (2021)
36. López Varela, Á., Dopico Dopico, D., Luaces Fernández, A.: Augmented Lagrangian index–3 semi–recursive formulations with projections: direct sensitivity analysis. Multibody Syst. Dyn. **61**(2), 195–231 (2023)
37. Al-Mohy, A.H., Higham, N.J.: A new scaling and squaring algorithm for the matrix exponential. SIAM J. Matrix Anal. Appl. **31**(3), 970–989 (2010)
38. Van Loan, C.: Computing integrals involving the matrix exponential. IEEE Trans. Autom. Control **23**(3), 395–404 (1978)
39. Guennebaud, G., Jacob, B., et al.: Eigen v3 (2010). http://eigen.tuxfamily.org