Testing the efficiency and accuracy of multibody-based state observers

Emilio Sanjurjo [*], **José L. Blanco** [#], **José L. Torres** [#], **Miguel A. Naya** [*]

[*] Laboratorio de Ingeniería Mecánica
University of La Coruña
Mendizábal s/n, 15403 Ferrol, Spain
[emilio.sanjurjo, minaya]@udc.es

[#] Escuela Politécnica Superior
University of Almería
La Cañada de San Urbano, 04120 Almería, Spain
[jlblanco, jltmoreno]@ual.es

ABSTRACT

Multibody simulations are already used in many industries to speed up the development of new products. However, improvements in multibody formulations and the continuous increase of computational power open new fields of applications for the multibody simulations, such as using them as a basis for developing state observers. This work introduces novel state observers developed by combining a multibody model with some probabilistic estimators from the family of Kalman filters. Together with other multibody-based state observers already proposed in the literature, have been benchmarked by applying them to two mechanisms: a planar four-bar linkage, and a five-bar linkage. The accuracy of the estimations as well as the computational costs are examined under several scenarios: using encoders or gyroscopes as sensors, with different sampling rates for the sensors, and assuming different levels of modeling errors. The results obtained in this work can be employed to select a suitable state observer for a new application. All methods have been implemented as a reusable MATLAB toolkit which has been released as Open Source in `https://github.com/MBDS/mbde-matlab`.

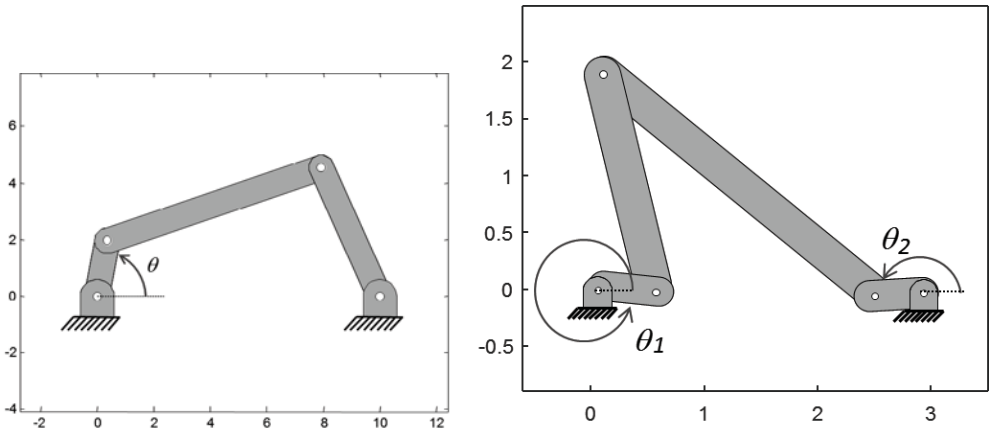
Keywords: Multibody simulation, State observers, Kalman filtering, Benchmark.

1 INTRODUCTION

Multibody simulations (MBS) are a common practice in the industry to speed up the development of new products. Since many years ago, this tool was employed offline to predict the behavior of new concepts, or extreme situations which are difficult or expensive to attain with a prototype. After that, with the development of new multibody formulations and the increase of computational power, real-time multibody simulations became a reality, allowing people and machines to interact with the multibody models in simulators and virtual test benches. Nowadays, with the advent of low cost, low power consumption computers, a multibody simulation could be run in vehicles or robots as part of their control algorithms, providing information about immeasurable magnitudes. The problem with this approach is that, in general, a multibody model can be very accurate in the short term if the forces are accurately known, but it will diverge over the time.

For this reason, multibody models should be corrected with information from the actual mechanisms in order to be used as reliable state observers. One way to reach this aim is employing information fusion techniques to combine information about multibody models with data provided by real sensors installed on the machine. A prominent example of information fusion technique is the Kalman filter (KF) which, however, presents some problems regarding its direct application to this problem. Firstly, the KF was originally formulated for first order, linear and unconstrained models, whereas multibody models are, in general, second order, highly nonlinear, and constrained systems. Moreover, the algorithm should be efficient in order to be run in real time. Because of these reasons, this is an open field of research. In a previous work [1], some existing methods [2, 3], and several new formulations were tested with different modeling errors and with different levels of sensor's noise.

In this work, two new MBS-based formulations were developed: the errorEKF, an error-state Kalman filter, and the projectionEKF, a method in dependent coordinates in which a projection technique is employed to fulfill the constraints of the multibody model, based on the methods



(a) (b)

Figure 1. Mechanisms employed in this work.

proposed in [4]. Their accuracy and performance was compared against the best methods presented in [1]: the CEKF and the UKF methods for their accuracy, and the DEKF and the DIEKFpm methods for their performance. The tests performed comprise different kind of sensors, including gyroscopes and encoders, different levels of modeling errors, and different sampling rates for the sensors, everything applied to two planar mechanism: a four-bar, and a five-bar linkages.

To provide a reference level for attainable accuracy, to which all these results can be compared, a batch estimator is applied to the whole history of sensor measurements, based on the non-linear optimization of the graphical model proposed in [5].

2 METHODOLOGY

At this moment there is a huge amount of different possibilities regarding the design of a state observer for a multibody model. For this reason, the only feasible approach to perform a large amount of tests is by means of simulations. For every set of tests, a multibody model is run playing the role of a real mechanism. This model provides the ground truth, and also is employed to build the signals from the sensors. To do this, perfect sensors are modeled, and then random noise is added to their measurements. A second multibody is built, but modifying some of the properties, to simulate modeling error. Usually, the geometry and the mass properties of any machine can be known with a great accuracy. However, the level of accuracy in force models is usually not so good. For this reason, the parameter which is intentionally modified is the acceleration of gravity. Finally, the state observer is built using the latter multibody model (the imperfect one), and corrected with the information provided by the sensors built from the simulation of the first multibody model.

The mechanisms selected for this benchmark are a four-bar and a five-bar linkages (see fig. 1). The four bar mechanism was selected because is the simplest closed-loop mechanism. The five bar mechanism is the next step in complexity to check that the algorithms can be generalized to any number of degrees of freedom. Gravity is the only actuating force.

Every test was run 20 times, and the results were averaged. The different tests performed comprise the two mechanisms, with tho different levels of error in the magnitude of the gravity, 4 different sampling rates for the sensors, and two different sets of sensors for every mechanism.

3 DESIGN OF THE OBSERVERS

Next we introduce the different filtering algorithms that are benchmarked in this work. The reader can refer to table 1 as a summary of the notation employed in this work.

n	Number of dependent coordinates
m	Number of constraints
$g = n - m$	Number of degrees of freedom
$\mathbf{z}$	Vector of independent coordinates
$\mathbf{q} = \mathbf{q}(\mathbf{z})$	Vector of dependent coordinates
$\mathbf{\Phi}(\mathbf{q}) = 0$	Constraint equations
$\mathbf{\Phi_q}$, $\mathbf{\Phi_x}$	Jacobian of $\mathbf{\Phi}$ with respect to $\hat{\mathbf{q}}$, $\hat{\mathbf{x}}$
$\mathbf{M}$	Mass matrix
$\mathbf{Q}$	Vector of generalized forces
$\mathbf{x}, \hat{\mathbf{x}}$	Real value and estimation of the filter state vector
$\hat{\mathbf{x}}_k^-, \hat{\mathbf{x}}_k^+$	Estimation mean at time step k, before and after the update stage
$\mathbf{P}_k^-, \mathbf{P}_k^+$	Estimation covariance at time step k, before and after the update stage
$\mathbf{f}(\cdot), \mathbf{f_x}, \mathbf{f_q}$	Transition model and its Jacobians w.r.t. $\hat{\mathbf{x}}$ and $\hat{\mathbf{q}}$
$\mathbf{h}(\cdot), \mathbf{h_x}, \mathbf{h_q}$	Observation (sensor) model and its Jacobians w.r.t. $\hat{\mathbf{x}}$ and $\hat{\mathbf{q}}$
$\mathbf{o}_k$	Sensor measurements at time step k
$\mathbf{\Sigma}^P$	Covariance matrix of system transition ("plant") noise
$\mathbf{\Sigma}^S$	Covariance matrix of sensors noise
$\mathbf{K}$	Kalman gain matrix
$\mathbf{I}_N$	The $N \times N$ unit matrix

Table 1. Notation summary.

3.1 Independent coordinates methods

The Kalman filter assumes that its states are independent. For this reason, the more natural way to combine a multibody model and a Kalman filter is by using independent coordinates.

3.1.1 Continuous extended Kalman filter (CEKF)

This formulation was already described in some previous works [2] but it will be reproduced here for the convenience of readers. The main idea under this formulation is to adapt the multibody equations in order to fit the Kalman filter structure. In its most basic form, the dynamics of a multibody system is described by the constrained Lagrangian equations:

$$\left\{ \begin{array}{rcl} \mathbf{M}\ddot{\mathbf{q}} + \mathbf{\Phi_q}^\top \boldsymbol{\lambda} & = & \mathbf{Q} \\ \mathbf{\Phi} & = & \mathbf{0} \end{array} \right. \tag{1}$$

As the multibody equations are expressed in the form of continuous-time differential equations, it seems natural to adopt the continuous-time version of the Kalman filter. The multibody formalism employed is the R-matrix formulation [6]. The main idea behind this formulation is to obtain an ODE with dimension g equal to the number of degrees of freedom, starting with the identity $\dot{\mathbf{q}} = \mathbf{R}\dot{\mathbf{z}}$, which relates dependent and independent velocities. Accelerations can be then expressed as follows:

$$\ddot{\mathbf{q}} = \mathbf{R}\ddot{\mathbf{z}} + \dot{\mathbf{R}}\dot{\mathbf{z}} \tag{2}$$

Going back to Eq. (1), premultiplying by the transpose of $\mathbf{R}$, and having in mind that $\mathbf{\Phi_q R} = \mathbf{0}$,

$$\ddot{\mathbf{z}} = \left(\mathbf{R}^\top \mathbf{M} \mathbf{R}\right)^{-1} \left[\mathbf{R}^\top \left(\mathbf{Q} - \mathbf{M}\dot{\mathbf{R}}\dot{\mathbf{z}}\right)\right] = \bar{\mathbf{M}}^{-1}\bar{\mathbf{Q}} \tag{3}$$

If now the filter state is defined as the vector $\mathbf{x}^\top = \left\{\mathbf{z}^\top, \dot{\mathbf{z}}^\top\right\}$, it turns out that:

$$\left\{ \begin{array}{c} \dot{\mathbf{z}} \\ \ddot{\mathbf{z}} \end{array} \right\} = \left\{ \begin{array}{c} \dot{\mathbf{z}} \\ \bar{\mathbf{M}}^{-1}\bar{\mathbf{Q}} \end{array} \right\} \quad \Rightarrow \quad \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) \tag{4}$$

These equations perfectly fit the continuous extended Kalman filter equation, so they can be straightforwardly applied. In particular, the state-space transition matrix is obtained as the linearization:

$$\mathbf{A} = \frac{\partial \mathbf{f}}{\partial \mathbf{x}} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ \dfrac{\partial \left(\bar{\mathbf{M}}^{-1}\bar{\mathbf{Q}}\right)}{\partial \mathbf{z}} & \dfrac{\partial \left(\bar{\mathbf{M}}^{-1}\bar{\mathbf{Q}}\right)}{\partial \dot{\mathbf{z}}} \end{bmatrix} \tag{5}$$

which can be approximated by:

$$\mathbf{A} \simeq \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix} \tag{6a}$$

$$\mathbf{A}_{21} = -\bar{\mathbf{M}}^{-1}\mathbf{R}^{\top}\left(\bar{\mathbf{K}}\mathbf{R} + 2\mathbf{R}_{\dot{\mathbf{q}}}\mathbf{R}\ddot{\mathbf{z}}\right) \tag{6b}$$

$$\mathbf{A}_{22} = -\bar{\mathbf{M}}^{-1}\mathbf{R}^{\top}(\bar{\mathbf{C}}\mathbf{R} + \mathbf{M}\dot{\mathbf{R}}) \tag{6c}$$

where $\bar{\mathbf{K}}$ and $\bar{\mathbf{C}}$ are the stiffness and damping matrices, respectively. In this case the size of the problem is $2g$. Next, we introduce the CEKF *correction stage* [7], which fuses the sensor information into the filter, leading to:

$$\dot{\mathbf{z}} - \hat{\dot{\mathbf{z}}} + \mathbf{K}^{\mathbf{z}}(\mathbf{y} - \mathbf{o}) = \mathbf{0} \tag{7a}$$

$$\bar{\mathbf{M}}\ddot{\mathbf{z}} - \bar{\mathbf{Q}} + \bar{\mathbf{M}}\mathbf{K}^{\dot{\mathbf{z}}}(\mathbf{y} - \mathbf{o}) = \mathbf{0} \tag{7b}$$

In order to numerically integrate the result of the filter, the implicit single-step trapezoidal rule has been selected as integrator:

$$\hat{\dot{\mathbf{z}}}_{n+1} = \frac{2}{\Delta t}\hat{\mathbf{z}}_{n+1} - \left(\frac{2}{\Delta t}\hat{\mathbf{z}}_n + \hat{\dot{\mathbf{z}}}_n\right) \tag{8a}$$

$$\hat{\ddot{\mathbf{z}}}_{n+1} = \frac{2}{\Delta t}\hat{\dot{\mathbf{z}}}_{n+1} - \left(\frac{2}{\Delta t}\hat{\dot{\mathbf{z}}}_n + \hat{\ddot{\mathbf{z}}}_n\right) \tag{8b}$$

Combining Eq. (7) and Eq. (8) leads to the following nonlinear system,

$$\begin{cases} \mathbf{g}_1(\hat{\mathbf{x}}_{n+1}) = \mathbf{0} \\ \mathbf{g}_2(\hat{\mathbf{x}}_{n+1}) = \mathbf{0} \end{cases} \Rightarrow \mathbf{g}(\hat{\mathbf{x}}_{n+1}) = \mathbf{0} \tag{9}$$

This system can be iteratively solved, e.g. by means of the Newton-Raphson method, employing the following approximate Jacobian matrix:

$$\frac{\partial \mathbf{g}}{\partial \mathbf{x}} = \begin{bmatrix} \dfrac{2}{\Delta t}\mathbf{I} & -\mathbf{I} \\ \mathbf{R}^{\top}\bar{\mathbf{K}}\mathbf{R} & \mathbf{R}^{\top}(\bar{\mathbf{C}}\mathbf{R} + \mathbf{M}\dot{\mathbf{R}}) + \dfrac{2}{\Delta t}\bar{\mathbf{M}} \end{bmatrix} + \begin{bmatrix} \mathbf{K}^{\mathbf{z}}\mathbf{h}_{\mathbf{z}} & \mathbf{K}^{\mathbf{z}}\mathbf{h}_{\dot{\mathbf{z}}} \\ \bar{\mathbf{M}}\mathbf{K}^{\dot{\mathbf{z}}}\mathbf{h}_{\mathbf{z}} & \bar{\mathbf{M}}\mathbf{K}^{\dot{\mathbf{z}}}\mathbf{h}_{\dot{\mathbf{z}}} \end{bmatrix} \tag{10}$$

where the pair $\mathbf{h}_{\mathbf{z}}$ and $\mathbf{h}_{\dot{\mathbf{z}}}$ and the pair $\mathbf{K}^{\mathbf{z}}$ and $\mathbf{K}^{\dot{\mathbf{z}}}$ are the position and velocity parts of the sensor Jacobian matrix and the Kalman gain matrix, respectively.

3.1.2 Discrete extended Kalman filter (DEKF)

This is the discrete-time version of CEKF described above. A key difference between CEKF and the rest of estimators described from now on, which work in discrete time steps, is that the filter formulation clearly consists of two separated stages: state transition (also called *prediction*) and *state update*. The former relies on the transition model of the system (dynamical equations) while the latter includes the information from sensors, or *observations* – this is in contrast to CEKF where both stages are seamlessly fused together.

Each stage comprises differentiated equations for updating the state vector and the covariance matrix. Starting with the prediction stage, the EKF equations in their most generic form are:

$$\hat{\mathbf{x}}_k^- = \mathbf{f}(\hat{\mathbf{x}}_{k-1}^+) \tag{11a}$$

$$\mathbf{P}_k^- = \mathbf{f}_{\mathbf{x}k-1}\mathbf{P}_{k-1}^+\mathbf{f}_{\mathbf{x}k-1}^{\top} + \mathbf{\Sigma}_{k-1}^P \tag{11b}$$

where $\mathbf{f}(\cdot)$ stands for the transition model of the system. By considering now the state vector of a MBS estimator in independent coordinates, $\hat{\mathbf{x}}^\top = \left\{\hat{\mathbf{z}}^\top, \hat{\dot{\mathbf{z}}}^\top\right\}$, and assuming the usage of the Euler method for numerical integration with time step Δt, we can put the integrator in a form that fits that required by the EKF transition function $\mathbf{f}(\cdot)$:

$$\hat{\mathbf{x}}_k^- = \mathbf{f}(\hat{\mathbf{x}}_{k-1}^+) \quad \longrightarrow \begin{bmatrix} \hat{\mathbf{z}}_k \\ \hat{\dot{\mathbf{z}}}_k \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{z}}_{k-1} + \Delta t \hat{\dot{\mathbf{z}}}_{k-1} \\ \hat{\dot{\mathbf{z}}}_{k-1} + \Delta t \ddot{\mathbf{z}}_{k-1} \end{bmatrix} \tag{12}$$

Here, the only unknown term is the acceleration vector $\ddot{\mathbf{z}}_{k-1}$ for the previous time step, which must be computed by solving the multibody equations of motions as in Eq. (3). Thus, it follows that the transition model Jacobian $\mathbf{f}_\mathbf{x}$ has a fairly simple structure:

$$\mathbf{f}_\mathbf{x} \equiv \frac{\partial \mathbf{f}}{\partial \hat{\mathbf{x}}} = \frac{\partial}{\partial \{\hat{\mathbf{z}}, \hat{\dot{\mathbf{z}}}\}} \begin{bmatrix} \hat{\mathbf{z}} + \Delta t \hat{\dot{\mathbf{z}}} \\ \hat{\dot{\mathbf{z}}} + \Delta t \ddot{\mathbf{z}} \end{bmatrix} = \begin{bmatrix} \mathbf{I}_g & \Delta t \mathbf{I}_g \\ \mathbf{0}_{g\times g} & \mathbf{I}_g \end{bmatrix} \tag{13}$$

Regarding the $\mathbf{\Sigma}_{k-1}^P$ covariance matrix appearing in Eq. (11), it stands for the additional uncertainty of the new state $\hat{\mathbf{x}}_k$, physically attributable to unmodeled forces and errors in the parameterization of the mechanism (e.g. lengths of bars, inertia values, etc.). Assuming independent and identically distributed (iid) Gaussian noise for each independent coordinate, its structure becomes:

$$\mathbf{\Sigma}_{k-1}^P = \left[\begin{array}{c|c} \sigma_{\hat{\mathbf{z}}}^2 \mathbf{I}_g & \mathbf{0}_{g\times g} \\ \hline \mathbf{0}_{g\times g} & \sigma_{\hat{\dot{\mathbf{z}}}}^2 \mathbf{I}_g \end{array}\right] \tag{14}$$

with the parameters $\sigma_{\hat{\mathbf{z}}}$ and $\sigma_{\hat{\dot{\mathbf{z}}}}$ specifying the standard deviations of the assumed noise in position and velocities, respectively.

The second stage of the DEKF method, the *update*, incorporates the sensor readings to improve the estimate:

$$\tilde{\mathbf{y}}_k = \mathbf{o}_k - \mathbf{h}(\hat{\mathbf{x}}_k^-) \tag{15a}$$
$$\mathbf{S}_k = \mathbf{h}_{\mathbf{x}k} \mathbf{P}_k^- \mathbf{h}_{\mathbf{x}k}^\top + \mathbf{\Sigma}_k^S \tag{15b}$$
$$\mathbf{K}_k = \mathbf{P}_k^- \mathbf{h}_{\mathbf{x}k}^\top \mathbf{S}_k^{-1} \tag{15c}$$
$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + \mathbf{K}_k \tilde{\mathbf{y}}_k \tag{15d}$$
$$\mathbf{P}_k^+ = (\mathbf{I}_g - \mathbf{K}_k \mathbf{h}_{\mathbf{x}k}) \mathbf{P}_k^- \tag{15e}$$

where $\mathbf{h}(\cdot)$ stands for the observation model of the system, such that $\tilde{\mathbf{y}}_k$ in Eq. (15) is clearly the error or mismatch (often called *innovation*) between the expected sensor readings and their actual values ($\mathbf{o}_k$). The covariance matrix $\mathbf{S}_k$ in Eq. (15b), or *innovation covariance*, represents the uncertainty in the system state projected via the sensor function ($\mathbf{h}_{\mathbf{x}k} \mathbf{P}_k^- \mathbf{h}_{\mathbf{x}k}^\top$) plus an additional additive Gaussian noise originated in the sensor itself ($\mathbf{\Sigma}_k^S$). Small values of $\mathbf{S}_k$ mean that the observation introduces useful information to constrain the estimation of the system state. By evaluating the temporary term known as *Kalman gain* ($\mathbf{K}_k$) we can update the estimate mean and covariance, in Eq. (15d) and Eq. (15e), respectively. These values are then used as the input to the next iteration of this iterative filter in the next time step.

3.1.3 Unscented Kalman filter (UKF)

The Unscented Kalman Filter (UKF) [8] is an evolution of the family of Kalman filters that is better suited to cope with strong nonlinearities in the transition and observation models. Comprising the same prediction and update stages than DEKF, the differentiating feature of UKF is the avoidance of the first order Taylor approximation in the propagation of Gaussian random variables through the transition and observation functions. Instead, a set of samples are deterministically-chosen from the Gaussian distributions, transformed via the corresponding function, then those samples in the transformed space converted back into a parametric distribution, i.e. they are used to compute

the mean and covariance of the corresponding Gaussian. As shown in [8], this approach captures the correct posterior mean and covariance up to the third order of a Taylor series expansion, in contrast to the first order of DEKF and most other methods. In turn, its computational cost is in general higher than simpler methods.

For the present benchmark, the state vector of UKF comprises the independent coordinates and their velocities, that is, $\hat{\mathbf{x}}^\top = \left\{\hat{\mathbf{z}}^\top, \hat{\dot{\mathbf{z}}}^\top\right\}$. As mentioned above, each filter iteration comprises the same two steps than DEKF, so only the differences will be highlighted here. Denoting the dimensionality of the state space $|\hat{\mathbf{x}}|$ as L, a total of $2L+1$ deterministic samples (or *sigma points*) χ_i with $i = 0, \ldots, 2L$ are generated from the mean $\hat{\mathbf{x}}_{k-1}^+$ and covariance $\mathbf{P}_{k-1}^+$, each with a different weight W_i. Then, the samples are transformed with a forward Euler transition function identical to that of previous filters, and the predicted mean $\hat{\mathbf{x}}_k^-$ and covariance $\mathbf{P}_k^-$ estimated from them. A similar process apply to the propagation of the uncertainty in observations, taking into account both the uncertainty in the system state and the sensor noise (refer to the two terms in the innovation covariance of DEKF above). The reader is referred to the original work [8] for the filter equations, not reproduced here for the sake of conciseness.

3.2 Dependent coordinates methods

Solving the position problem every time step has a high computational cost. For this reason, it is worth looking for alternatives to include the constraints into the Kalman filter.

3.2.1 Discrete iterated extended Kalman filter with perfect measurements (DIEKFpm)

This method is a expansion of the standard DIEKF [7] to cope with constraints in its state space by employing so-called *perfect measurements* [4]. The key idea consists of augmenting the vector of observations $\mathbf{h}(\mathbf{x})$ to include *virtual observations* that reflect the fulfillment of the kinematics constraints in both position and velocities. The augmented observation function $\mathbf{h}'(\mathbf{x})$ is strongly nonlinear, hence the application in this case of an iterated estimator, capable of reducing the linearization errors to acceptable levels.

For the benchmark at hand, the state vector of this estimator comprises the multibody model coordinates and their derivatives, that is, $\hat{\mathbf{x}}^\top = \left\{\hat{\mathbf{q}}^\top, \hat{\dot{\mathbf{q}}}^\top\right\}$. We define the augmented observation model $\mathbf{h}'(\hat{\mathbf{x}})$ as the concatenation of the real sensors $\mathbf{h}(\mathbf{x})$ and the kinematic constraints in position and velocity, such as $\mathbf{h}'(\mathbf{x})^\top = [\mathbf{h}(\mathbf{x})^\top \ \boldsymbol{\Phi}(\mathbf{x})^\top \ \dot{\boldsymbol{\Phi}}(\mathbf{x})^\top]$. This affects the calculation of the innovation (or "residual"), which must compare the actual sensor readings and current constraint errors with their predictions. For all time steps k and iteration index i, the predicted values of the constrains are always zero, i.e.

$$\mathbf{y}_{k,i} = \begin{bmatrix} \mathbf{o}_k \\ \mathbf{0}_{2n\times 1} \end{bmatrix} - \mathbf{h}'(\hat{\mathbf{x}}_{k,i}) = \begin{bmatrix} \mathbf{o}_k - \mathbf{h}(\hat{\mathbf{x}}_{k,i}) \\ -\boldsymbol{\Phi}(\hat{\mathbf{x}}_{k,i}) \\ -\dot{\boldsymbol{\Phi}}(\hat{\mathbf{x}}_{k,i}) \end{bmatrix} \tag{16}$$

The adjective "perfect" that names this method comes from the assumption that there is no error source in the virtual observations. In practice, this implies employing an extended sensor covariance matrix $\boldsymbol{\Sigma}_k^{S'}$ with the structure:

$$\boldsymbol{\Sigma}_k^{S'} = \begin{bmatrix} \boldsymbol{\Sigma}_k^S & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{17}$$

3.2.2 Extended Kalman filter with projections (projectionEKF)

Other method in dependent coordinates was also developed. The states of this method comprise the full set of position and velocity coordinates of the multibody model. The propagation and correction stages are performed as if the states were independent. After that, the states are projected over the constraint manifold, based in the constrained Kalman filter algorithms presented in [4]. However, some modifications were made to the algorithm to adapt it to the nature of the multibody simulations. The projections are performed in two stages. First, the position coordinates

are projected. As the position constraints are non-linear, the process is iterative, using the next equation:

$$\mathbf{q} = \mathbf{q}^* - \mathbf{W}^{-1}\boldsymbol{\Phi}_\mathbf{q}\left(\boldsymbol{\Phi}_\mathbf{q}\mathbf{W}^{-1}\boldsymbol{\Phi}_\mathbf{q}{}^T\right)^{-1}\boldsymbol{\Phi}\left(\mathbf{q}^*\right) \tag{18}$$

where $\mathbf{q}^*$ are the position coordinates after the correction stage, and $\mathbf{W}$ is a weight matrix. After that, the velocity coordinates are projected. As the constraint at velocity level are linear, this projection is performed in one step:

$$\dot{\mathbf{q}} = \dot{\mathbf{q}}^* - \mathbf{W}^{-1}\boldsymbol{\Phi}_\mathbf{q}\left(\boldsymbol{\Phi}_\mathbf{q}\mathbf{W}^{-1}\boldsymbol{\Phi}_\mathbf{q}{}^T\right)^{-1}\left(\boldsymbol{\Phi}_\mathbf{q}\dot{\mathbf{q}}^*\right) \tag{19}$$

where $\dot{\mathbf{q}}$ is the vector of velocities obtained after the correction stage of the Kalman filter.

After applying the projections, the covariance matrix of the states, $\mathbf{P}$, must be updated to take into account the effect of the projections, as follows:

$$\mathbf{P}^\mathbf{q} = \mathbf{P}^{\mathbf{q}*} - \mathbf{P}^{\mathbf{q}*}\boldsymbol{\Phi}_\mathbf{q}{}^T\left(\boldsymbol{\Phi}_\mathbf{q}\mathbf{P}^{\mathbf{q}*}\boldsymbol{\Phi}_\mathbf{q}{}^T\right)^{-1}\boldsymbol{\Phi}_\mathbf{q}\mathbf{P}^{\mathbf{q}*} \tag{20a}$$

$$\mathbf{P}^{\dot{\mathbf{q}}} = \mathbf{P}^{\dot{\mathbf{q}}*} - \mathbf{P}^{\dot{\mathbf{q}}*}\boldsymbol{\Phi}_\mathbf{q}{}^T\left(\boldsymbol{\Phi}_\mathbf{q}\mathbf{P}^{\dot{\mathbf{q}}*}\boldsymbol{\Phi}_\mathbf{q}{}^T\right)^{-1}\boldsymbol{\Phi}_\mathbf{q}\mathbf{P}^{\dot{\mathbf{q}}*} \tag{20b}$$

where $\mathbf{P}^{\mathbf{q}*}$ and $\mathbf{P}^{\dot{\mathbf{q}}*}$ are the blocks of of the covariance matrix correspondent to positions and velocities before the projection process, and $\mathbf{P}^{\mathbf{q}*}$ and $\mathbf{P}^{\dot{\mathbf{q}}*}$ are the blocks of the covariance matrix after the projection process.

Nevertheless, it was found that this method does not work properly. The reason is that, in general, the information from the sensors is related not to the full state of the mechanism, but only to some coordinates. This fact produces that during the correction stage, the constraints are broken. After that, during the projection process, the mechanism is conducted to a new state in which the constrains are fulfilled, but most of the effect of the correction is lost. For this reason, this method was not considered in the benchmark.

3.3 Mixed method: errorEKF

Most of the methods presented previously have some common drawbacks. One of them is that the integrator must be explicit. Also, the nonlinearities of the multibody models can be not properly handled by the EKF, which is only a first order approximation. Both these problems are overcome by the UKF method, but at high computational cost. So a new formulation was developed. In it, the multibody simulation is run in parallel with a Kalman filter which estimates not the position nor the velocity vectors of the multibody model, but the errors of the independent coordinates and velocities, so it is an error-state extended Kalman filter (also known as indirect extended Kalman filter). With this approach, any existing multibody simulation can be used, regardless of the kind of integrator or multibody formulation employed. After one step of the multibody simulation is performed, the estimation of the error is launched. The propagation phase is performed following the next equations:

$$\hat{\mathbf{x}}_k^- = \mathbf{0} \tag{21}$$

$$\mathbf{P}_k^- = \mathbf{f}_{\mathbf{x}k-1}\mathbf{P}_{k-1}^+\mathbf{f}_{\mathbf{x}k-1}^\top + \boldsymbol{\Sigma}_{k-1}^P \tag{22}$$

These equations are the conventional equations for the propagation of the Kalman state. However, as the estimated errors are fed back to the multibody simulation, the estimation of the error in the propagation phase is always null, as shown in Eq. (21).

The equations for the correction phase of the filter are also equivalent to the ones found in a conventional Kalman filter. After the correction stage, the estimation for the errors of positions and velocities of the independent coordinates are obtained. However, to correct the state of the multibody system, the errors for all the coordinates must be obtained, so them must be projected over the constraints manifold.

An error in position means that an increment to the coordinates should be applied to get the position corrected. Such an increment must fulfill the velocity constraints. So the increments applied

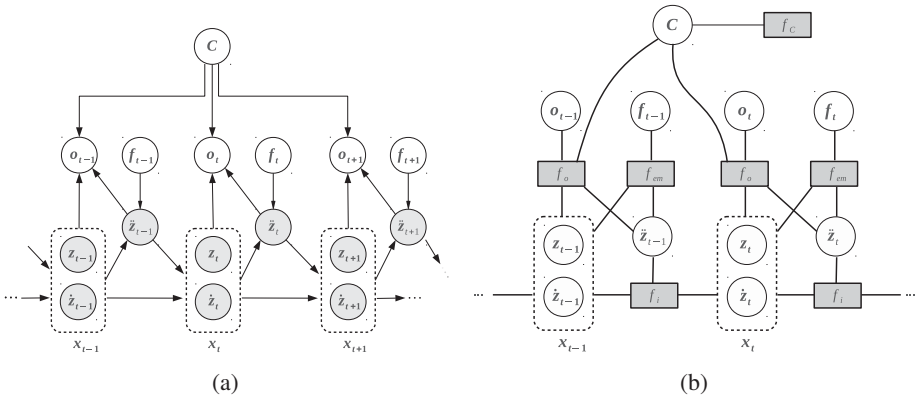


Figure 2. The final factor graph. Circle nodes are problem variables, rectangular nodes are factors.

to the coordinates of the mechanism are calculated by solving the velocity problem. This fact represent an advantage compared against the other methods based in independent coordinates in which solving the position problem is necessary, since the velocity problem is linear, and can be solved without employing iterative methods.

As said previously, the velocity problem is linear, so for correcting the velocities of the multibody system, the velocity problem must be solved using the estimation of the velocity error of the degrees of freedom. The result is added to the vector of velocities of the multibody model.

Although this method does not get to a perfect fulfilling of the constraints, it is enough to provide accurate estimations of the position and velocity of a mechanism.

3.4 The reference: a batch smoother

All state observers introduced up to this point share the key aspect of being *filters*. A filter processes the new information available at each timestep sequentially and immediately outputs an estimate of the current system state. Therefore, a clear advantage of filters is the fast availability of state estimates suitable for real-time closed-loop control, among other applications. On the other hand, the computational efficiency of filters is gained at the cost of forgetting the probability distribution of the entire history of the dynamic system except the very last timestep. If one uses Gaussian distributions to implement this process of *forgetting* information, named *marginalization* in the statistical inference literature [9], significant errors are introduced in the case of nonlinear dynamic systems, especially regarding the cross-covariances between the different variables.

The implication is that observation-based corrections of all the filters introduced above are suboptimal in some degree due to inaccuracies in cross-correlation terms of the estimate covariance matrix. Additionally, some of those filters have another source of approximation in the first (EKF) or second-order (UKF) truncation of Taylor series for system equations.

As an alternative, we introduce in this work a novel *smoother* estimator which serves as a reference of the achievable accuracy for each combination of model and sensor errors in each benchmarked mechanism. Unlike filters, a smoother handles a *joint* probability distribution for the dynamic state during a sequence of timesteps. In particular, we have designed a batch estimator which considers the full history of dynamic states, which provides a better estimation than filters since the entire trajectory of the system is simultaneously subject to all existing constraints, both forwards and backwards in time, effectively *smoothing* the estimation as much as possible –hence the name. Our novel method is based on the Dynamic Bayesian Network (DBN) model of a Multibody Dynamic system devised in [5], and which is shown in Figure 2(a). A DBN encodes the causal relationships (directed edges) between all the variables (nodes) involved in a dynamic problem. Please refer to

[5] for an in-depth discussion. In this work, we convert the DBN into its equivalent factor-graph form (see e.g. [10]), a kind of undirected bipartite graph where variables are represented as circular nodes, *factor functions* encode the relationships between variables (represented as square nodes) and undirected edges join factors with all the involved variables. Using the rules for converting DBNs into factor graphs we arrive at Figure 2(b). It is known that factor graphs allow writing down the full joint probability distribution of the system as the product of one *cost function* per factor node [9], such that in our case we arrive at:

$$\varphi(\hat{\mathbf{x}}_{0:t}, \ddot{\mathbf{z}}_{0:t}, \mathbf{o}_{0:t}, \mathbf{f}_{0:t}, \mathbf{B}) = \varphi_{ini}(\hat{\mathbf{x}}_0)\varphi_B(\mathbf{B})\prod_{i=1}^{t}\varphi_o(\mathbf{z}_i, \ddot{\mathbf{z}}_i, \hat{\mathbf{x}}_i, \mathbf{B})\varphi_{eom}(\ddot{\mathbf{z}}_i, \hat{\mathbf{x}}_i, \mathbf{f}_i)\varphi_i(\hat{\mathbf{x}}_i, \ddot{\mathbf{z}}_{i-1}, \hat{\mathbf{x}}_i) \tag{23}$$

Where each term has a well-defined physical meaning: φ_{ini} represents a (typically loose) constraint regarding the a priori knowledge about the initial dynamical state of the system, φ_B stands for the knowledge about the *branch* of the mechanism for those having multiple assembly configurations (please, refer to Figure 3 in [5]), φ_o stands for the sensor models, φ_{eom} introduces the predicted accelerations according to the equations of motion and the theoretical multibody model, and φ_i represents the constraints imposed by a numerical integrator to consecutive dynamic states.

Assuming Gaussian distributions for errors in all factors, it can be shown that the negative logarithm of Eq. 23 becomes a quadratic cost function which can be optimized numerically to obtain the most likely estimation of the mechanism dynamical trajectory. In this work we employed the Levenberg-Marquardt with excellent convergence results towards the optimal solution.

4 RESULTS

Four different configurations were tested in this work. For every one, two levels of modelization error were considered: 1 and 4 m/s^2 of error in the gravity, and four different sampling rates of the sensors: one measurement every time step, and one measurement every 5, 10, and 15 time steps. The noise of the position sensors (encoders) employed in this work has a standard deviation of 1 deg, while the noise of the gyroscopes has a standard deviation of 1 deg/s

4.1 Test 1: Five-bar linkage with gyroscopes

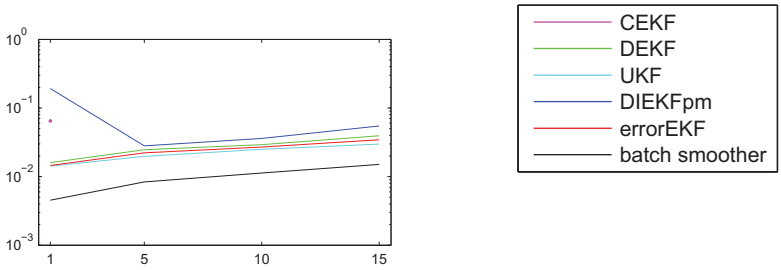
In this test, the mechanism to be considered is the the five-bar linkage. The sensors installed are two gyroscopes in both coupler links. In fig. 3 can be seen that the CEKF can only run in this configuration if it has a measurement every time step, and even in that case, the results are not very accurate. Next, filter in accuracy in the DIEKFpm. The three best filters are very close to each other, being the best the UKF, then the errorEKF, and in third place, the DEKF. The batch smoother provides the best results, as expected.

None of these methods could provide proper results with an error of 4 m/s in this configuration. Possibly, the reason for this fact is that, due to the particular geometry of the mechanism, in some ranges of the motion, big displacements of the crank correspond to small angular rates in the coupler rods, so the gyroscopes in those bars do not provide enough information to correct the position of the system.

4.2 Test 2: Five-bar linkage with encoders

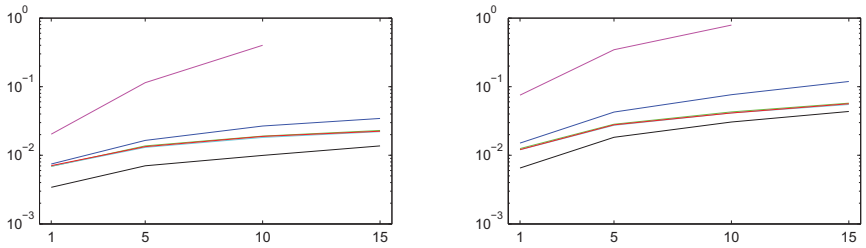
This test is based in the same mechanism than that of the previous one, but with a different set of sensors. Now, one encoder is considered in each crank. The results can be seen in fig. 4

With this configuration, bigger errors in the model can be handled. Also, the CEKF can be run with a sensor with a lower sampling rate than in the previous test. However, its performance is still the worst. The UKF, the errorEKF and the DEKF are even closer that in the previous test, with almost any difference in between them in terms of accuracy.



(a) (b)

Figure 3. (a) Results for the five-bar linkage with two gyroscopes and 1 m/s^2 of error in the gravity model. In the y axis, the RMS error in rad/s is represented, while in the x axis is represented the sampling time of the sensors, expressed as number of time steps. (b) Legend



(a) (b)

Figure 4. Results for the five-bar linkage with two encoders. (a) 1 m/s^2 of error in the gravity model. (b) 4 m/s^2 of error in the gravity model.

4.3 Test 3: Four-bar linkage with a gyroscope

The third test is based in the four-bar linkage, with a gyroscope installed on the coupler rod. The results can be seen in fig. 5

In this case, the geometry of the four-bar linkage allows the gyroscope to provide more information about the position of the mechanism than the five-bar linkage, so the observers remained stable even with the highest level of modeling error.

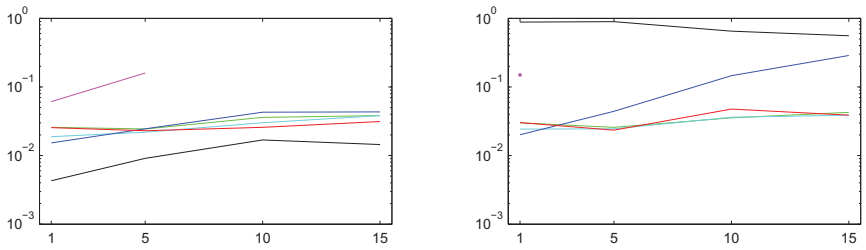
The CEKF presents problems again if the sampling rate of the sensors is not high enough. Regarding the other observers, it is remarkable that the DIEKFpm has the best accuracy if the measurement of the sensor is available at every time step, but when the sensor has a lower sampling rate, the results became worse, especially when the modeling error is high. Also, the batch smoother could not provide proper estimations in the test with the highest modeling error.

4.4 Test 4: Four-bar linkage with an encoder

This test is based in the four-bar linkage with an encoder in the left crank. The results are shown in fig. 6. In this test, the DIEKFpm obtained the best results for the lowest level of modeling error, but the worst after the CEKF with the highest modeling error. The CEKF was again the worst in both tests. Very similar accuracy was obtained from the UKF, the CEKF, and the errorEKF.

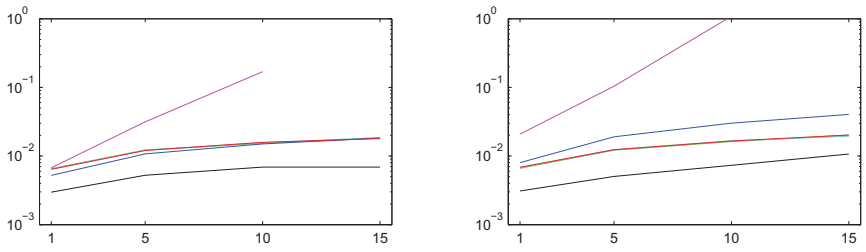
4.5 Performance

For an estate observer to be useful, it must be run in real time. Although the programs employed for this work were not designed nor implemented with performance in mind, it is expected that



(a) (b)

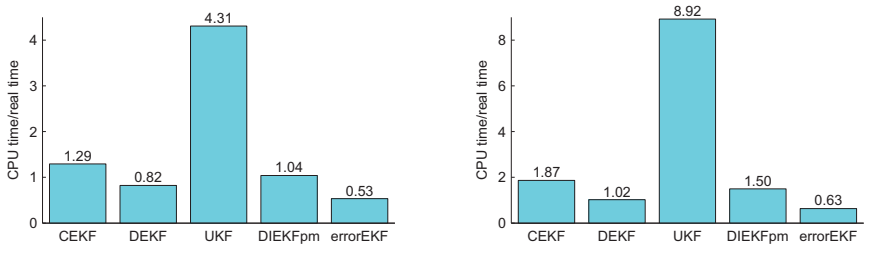
Figure 5. Results for the four-bar linkage with a gyroscope. (a) 1 m/s^2 of error in the gravity model. (b) 4 m/s^2 of error in the gravity model.



(a) (b)

Figure 6. Results for the four-bar linkage with an encoder. (a) 1 m/s^2 of error in the gravity model. (b) 4 m/s^2 of error in the gravity model.

the faster algorithm will perform faster both in the programs used here, or in a program designed and implemented specifically for real time performance. For this reason, a comparative of the time employed to run two of the previous tests is provided in fig. 7. The time varies depending on the type of test run, so this figure can be used just as a guide to see which method is faster.



(a) (b)

Figure 7. Performance of the different methods. (a) Four-bar linkage. (b) Five-bar linkage

It is worth focusing on the time of the DEKF, the UKF, and the errorEKF, because they got very similar results with regard the accuracy. However the computational cost of them is very different, being the errorEKF the fastest, and the UKF the slowest.

5 CONCLUSIONS

In this work, several state observers based on multibody models have been tested, checking the accuracy of the algorithms and their speed. Most of the methods tested were already published, while other, such as the errorEKF, the projectionEKF, and the batch smoother were developed in

this work. The batch smoother is not an observer. However, it uses the whole history of the states of the mechanism to obtain the most likely trajectory. It is used here as a reference for the best possible solution. The projectionEKF was not considered in the benchmark because it did not work as well as expected. However, the errorEKF showed an accuracy at the level of the best filters with a much lower computational cost.

The tests performed in this work were based in two planar mechanism: a four-bar and a five-bar linkages, with two different set of sensors each, and with four different sampling rates for the sensors.

Our reference MATLAB implementation has been released as Open Source[1].

6 ACKNOWLEDGMENTS

This work has been partially funded by fellowship BES-2013-063598 of Spanish Government, the project "DAVARBOT" (DPI 2011-22513) and the grant program JDC-MICINN 2011, financed by the Spanish Ministry of Science and Innovation.

REFERENCES

[1] J. Torres, J. Blanco, E. Sanjurjo, M. Naya, A. Giménez. Towards benchmarking of state estimators for multibody dynamics. In The 3rd Joint International Conference on Multibody System Dynamics. The 7th Asian Conference on Multibody Dynamics, pp. 261–262, Busan, Korea, 2014.

[2] J. Cuadrado, D. Dopico, A. Barreiro, E. Delgado. Real-time state observers based on multibody models and the extended Kalman filter. Journal of Mechanical Science and Technology, Vol. 23, No. 4, pp. 894–900, 2009.

[3] R. Pastorino, D. Richiedei, J. Cuadrado, A. Trevisani. State estimation using multibody models and non-linear Kalman filter. International Journal of Non-Linear Mechanics, Vol. 53, pp. 83–90, 2013.

[4] D. Simon, T. L. Chia. Kalman filtering with state equality constraints. IEEE Transactions on Aerospace and Electronic Systems, Vol. 38, No. 1, pp. 128–136, 2002.

[5] J.-L. Blanco, J.-L. Torres-Moreno, A. Giménez-Fernández. Multibody dynamic systems as Bayesian Networks: applications to robust state estimation of mechanisms. Multibody System Dynamics, in press.

[6] J. García de Jalón, E. Bayo. Kinematic and Dynamic Simulation of Multibody Systems: The Real Time Challenge, Springer-Verlag, 1994.

[7] D. Simon. Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches, Wiley, Hoboken2006.

[8] E. Wan, R. Van Der Merwe. The Unscented Kalman Filter for Nonlinear Estimation. In Proceedings of the IEEE Adaptive Systems for Signal Processing, Communications, and Control Symposium, pp. 153–158, 2000.

[9] C. Bishop. Pattern Recognition and Machine Learning, Springer, New York2006.

[10] H.-A. Loeliger. An introduction to factor graphs. IEEE Signal Processing Magazine, Vol. 21, No. 1, pp. 28–41, 2004.

[1] See `https://github.com/MBDS/mbde-matlab`