Multiple-Model Linear Kalman Filter Framework for Unpredictable Signals

Carolina Brum Medeiros, *Student Member, IEEE,* and Marcelo M. Wanderley, *Member, IEEE*

***Abstract*— This paper presents sensor fusion techniques for systems where the process model is a function of the human input and, therefore, unpredictable. The system consists of free and user-driven motion regimes. The free regime can be modeled as a damped sinusoidal waveform, while the driven regime and the transitions between regimes do not respect any sort of probability, pattern, or sequence. The quantity of interest is the deflection of a clamped beam, measured using three sensor technologies: 1) strain gages; 2) infrared; and 3) Hall effect sensors. Experiments using infrared-based motion capture as reference measuring system show that: 1) none of the sensors present optimal performance for both motion regimes and 2) measurement errors of each sensor differ significantly according to the motion regime. These findings suggest the use of sensor fusion techniques with low processing cost, compatible with real-time embedded applications. Our solution is based on a multiple-model linear Kalman filter in combination with motion segmentation. The motion segmentation discriminates gestures according to the knowledge of their process model. This allows a more predictive estimation during periods of free motion, while relying on a less predictive approach for unknown user-driven signals. In addition, we propose a framework on evaluation and selection of process models for unpredictable signals. The implementation was compared with single-sensor and single-model filter designs. Results based on human subject data reveal that the proposed method improves the error covariance of the estimate by a factor of 2.2 for driven motions and 12.7 for free motions in comparison with single-sensor filter design.**

***Index Terms*— Sensor fusion, user interfaces, Kalman filter evaluation, classification, strain gages.**

I. Introduction

THE popularity of input devices for computer control has raised attention to gestures and sensors. Typically, consumer electronics manufacturers introduce their own vocabulary of gestures to control their devices. These gestures are tracked by a number of sensors embedded in the devices. A predefined set of gestures, forming a vocabulary, facilitates the mapping between gestures and control, on account of expected patterns on sensor data. In these cases, sensor data should be accurate enough to allow for the gesture to be corrected classified within the predefined vocabulary. However, some devices do not have a predefined vocabulary of gestures, allowing freedom for the user to define his own gesture vocabulary. An example of the latter are Digital Musical Instruments (DMIs).

DMIs are musical instruments where the user interaction is measured by sensors and mapped to sounds [1], [2]. These instruments have gained interest among musicians and performers in the last few decades and have given rise to artistic practices widespread across a variety of art forms (music performance, interactive installations, dance and multimedia). Musicians are known to have refined motor control – they perform highly-developed body gestures while playing instruments – and proficient auditory perception. These qualities make them aware of the output of each gesture, allowing them to identify any incoherence on the mapping between gesture and sound produced. Also, on many occasions, musicians perform similar gestures with different qualities, extending the technique to something unique. In order to address the musicians' needs, the instrumentation design – sensor and signal conditioning – must meet several requirements. First, the DMI must not restrict the user interaction to an enclosed vocabulary, empowering the musician to control the instrument in an expressive way. Additionally, the instrumentation design should be accurate, reproducible, monotonic, with fast response in real-time and preferably embedded in the device.

In practice, most of the DMIs are based on one sensor technology [1]. For these, the operation is vulnerable to the limitations of the sensor technology used. Additionally, nowadays many DMIs are developed in a Do-It-Yourself (DIY) manner, prioritizing ordinary sensors, that is, easily available sensors which require simple assembly and signal conditioning circuits. We believe that robuster designs are needed to allow musicians to express themselves musically as well as to encourage a greater dissemination and commercialization of DMIs.

In this work, we discuss instrumentation for a DMI based on three sensor technologies: strain gages(SG), infrared(IR) and Hall effect(HL) sensors. Results demonstrate that each sensor has advantages and drawbacks. Besides that, it is not possible to determine an optimal single sensor solution that is advantageous in all common operations [3]. It follows that the most favourable solution would be a combination of features of several sensors. This suggests the use of sensor fusion in order to design a robust instrumentation for a DMI.

Manuscript received August 27, 2013; revised October 18, 2013; accepted November 1, 2013. Date of publication November 20, 2013; date of current version February 12, 2014. This work was supported by the Natural Sciences and Engineering Research Council of Canada under an individual Discovery Grant to M. M. Wanderley. The associate editor coordinating the review of this paper and approving it for publication was Dr. Ashish Pandharipande.

C. Brum Medeiros is with the Music Technology Program, Schulich School of Music, McGill University, Montreal, QC H3A 1E3, Canada, and also with the Input Devices for Musical Interaction Laboratory, McGill University, Montreal, QC H3A 1E3, Canada (e-mail: carolina.medeiros@mail.mcgill.ca).

M. M. Wanderley is with the Schulich School of Music, McGill University, Montreal, QC H3A 1E3, Canada, and also with the Centre for Interdisciplinary Research in Music Media and Technology, McGill University, Montreal, QC H3A 1E3, Canada (e-mail: marcelo.wanderley@mcgill.ca).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSEN.2013.2291683

We then propose a linear Kalman filter in order to accomplish the fusion task. The Kalman filter is proven to be effective when the process model of the system is known. For instance, the numerous examples of Kalman filter application in navigation where the state variables are position, velocity and acceleration [4]–[6]. In the present case, the user interaction with the DMI is unpredictable, therefore the process model of the system is not entirely known. Kalman filter performance rely on an accurate description of the system and measurements [7]. Model uncertainties deteriorate the optimality of the estimation or can even cause divergence [8].

Few works discuss Kalman filter design for time-varying statistical parameters or models. Basically, some categories of solutions can be identified: adaptive Kalman filter, extended Kalman filter and signal/statistical processing.

The idea behind the adaptive Kalman filter is to modify the error covariance of the measurements, system or estimate according to some cues [9]–[12]. This solution is limiting as it opens the bandwidth for noisy data by reducing the trust in the process model.

The second solution is to use an Extended Kalman filter (EKF), a nonlinear version of Kalman filter. The EKF is a common solution for problems where accurate knowledge of the process model is not available [13]. The disadvantage of EKF in relation to its linear counterpart is the higher computation cost, which makes it nonviable for embedded real-time applications. Some works implement the EKF setting the unknown parameters as state variables to be estimated, and therefore, estimating both states and process parameters [14]–[16]. For these cases, several factors contribute for the limitation of the EKF filter, most of them closely related to the linearization process and unavoidable in our application [17]–[20].

Other solutions implement multiple-model filter design in order to better describe distinguished system behaviours. Interesting implementations arise from the combination of multiple-model filters and signal processing techniques or statistical tools. Some examples of these techniques are pattern recognition, machine learning, sequential rules, transition probabilities and statistical models or tools [21]–[27]. The combination of a multiple-model filter and these signal processing techniques is not suitable for our problem, as the transitions respect no probability, pattern or sequence.

One of the requirements for a DMI instrumentation is low processing time, due to its real-time application. This makes prohibitive all linearization-based and machine learning based solutions described above. Additionally, no statistical rule for multiple-model transitions is applicable as it is not possible to define duration, grip type or magnitude of gestures that will be performed by the musician. This sheds light on the need of a new method for sensor fusion when model and transition between models cannot be described accurately.

This paper is organized as follows. In Section II, the device and its instrumentation are introduced. Section III presents the sensor data conditioning. The problem is stated in Section IV. The proposed solution is presented in Section V. The implementation is presented in Sections VI and VII. Results and discussions are presented in Section VIII. Finally, conclusions are given in Section IX.

Fig. 1. *The Rulers*: a Digital Musical Instrument, a set of seven cantilever beams designed to evoke deflection gestures (photo and design by David Birnbaum).

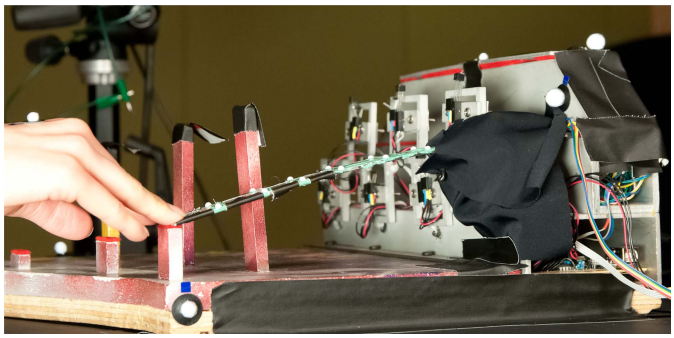

Fig. 2. Performer executing **bending:** driven motion (photo by Guillaume Pelletier).

II. Device and Instrumentation Description

The DMI we focus on in this paper is David Birnbaum's *The Rulers*, shown in Fig. 1, a set of seven cantilever beams designed to evoke deflection gestures [28]. In this work, *deflection* refers to the distance between the equilibrium vertical position and the vertical position next to the free edge of the beam at any time.

The main categories of gestures used to play *The Rulers* are "*bending*" and "*plucking*". Bending is described as a driven motion where the user grips the free edge of the beam, bending it up or down or stoping a motion (see Fig. 2). Alternatively, the user can perform an initial driven deflection - bending - followed by the beam's release. If the grip is released when the beam is not at the equilibrium position, the beam deflects around its equilibrium position with respect to a damping ratio. This free oscillation is called "plucking" (see Fig. 3). In other words, bending happens whenever the user is controlling the beam deflection and plucking happens whenever the user is not interacting with the beam.

Bending gestures can be approximated as a deflection resulting from a concentrated load next to the free edge.

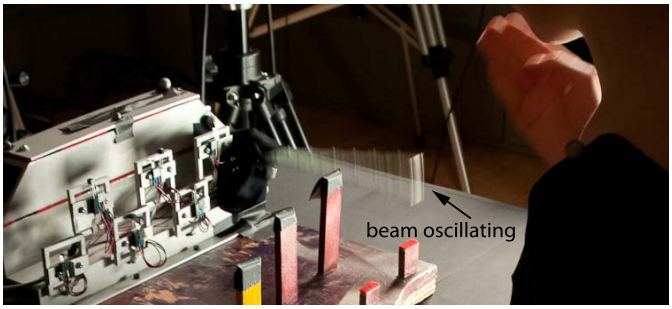

Fig. 3. Performer executing **plucking:** free beam oscillation originated by an initial driven deflection (photo by Guillaume Pelletier).

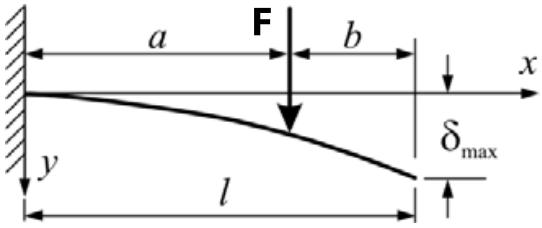

Fig. 4. A concentrated force **F** is applied at a given point. The distance between the load and the clamped edge is a and the distance between the load and the free edge is b. The deflection next to the free edge δ_{max} depends on the beam material, beam geometry, force magnitude **F**, distance between load and clamped edge a and length of the beam l.

The deflection depends on the mechanical properties of the material, the beam geometry, the force magnitude and the distance between clamp and force application point. Fig. 4 shows a diagram of a clamped beam being bent.

Given a concentrated force **F**, the deflection δ at a distance x from the clamp where $(a < x \leq l)$ is given by:

$$\delta = \frac{\mathbf{F}\, a^2}{6\, E\, I}\,(3\, x - a) \tag{1}$$

where E is the Modulus of Elasticity, intrinsic to the material, and I is the area moment of inertia, defined by the beam geometry. Note that the force **F** and the distance a are not definable once they are determined by user input.

Plucking gestures can be approximated as an underdamped oscillation, and therefore, satisfies the second-order differential damped wave equation. The oscillation depends on the angular frequency, the damping ratio and on the deflection magnitude at the beam release.

A. Limitations of Previous Instrumentation Strategies

Previous work evaluated and compared three sensor technologies to measure the deflection for *The Rulers* [3]. Infrared and Hall effect sensors are recurrent solutions for measuring proximity. Strain gages are the state-of-art solution to measure strain, deriving quantities such as stress, pressure, deflection and flow. Infrared and Hall effect sensors can be applied to the specimen in a straight-forward manner and require relatively simple conditioning circuits, while strain gages require specialized knowledge of the system's mechanical characteristics, careful application on the specimen and high-quality instrumentation circuits in order to provide an output signal with sufficient signal-to-noise ratio (SNR) [29].

The infrared sensor has a phototransistor detector with peak operating distance at 2.5 mm and a daylight blocking filter. Its collector current varies with the distance to a reflecting surface [30]. The Hall effect sensor has temperature compensation, adjustable gain and was set to bipolar operation. Magnets were attached to the beam, so that the Hall effect sensor could sense the proximity of the beam. Even if the manufacturer claims the sensor as a "Linear Output Magnetic Field Sensor", it is crucial to note that being linear to magnetic fields does not imply linearity to magnetic field variability that results from the beam deflection [31], [32]. The strain gages are connected in a full Wheatstone bridge: two submitted to compression and two submitted to extension, installed on the bottom and top of the beam. Considering that there is no airflow or temperature gradient between top and bottom of the beam, a temperature compensation is performed by the full bridge configuration.

It is important to note that none of the sensors are directly sensitive to distance, but rather to infrared light, magnetic field (Hall effect) and strain. Therefore, they indirectly measure the quantity of interest: deflection. As a consequence, these sensors might be vulnerable to environmental factors [33], [34], such as external magnetic fields, temperature, other stress sources and stage lighting with considerable infrared spectrum.

We are interested in the deflection next to free edge, where the user input is most perceptible. Fig. 5 presents the deflection next to the free edge *versus* the sensor output for bending motions. Subsequently, some practical considerations for the sensors are presented, followed by a summary of their characteristics.

Infrared and Hall sensors measure the beam-sensor distance next to the clamped edge. The ratio between the deflection measured by the sensor and the deflection next to the free edge was investigated. Tests using a Qualisys© 16-camera passive infrared motion capture system for measuring deflection reported that the mentioned ratio is constant along the full deflection range. In addition, for infrared and Hall effect sensors, the distance between sensors and beam varies with respect to an angle. This can be an issue specially dealing with infrared sensors whose operation is based on surface reflection. The solution for this potential artifact was to make sure that the surface area of beam and magnet are larger than the focus of the sensors.

Furthermore, infrared and Hall effect sensors might present measurement ranges that are not monotonic. Non-monotonic ranges require intense processing to map input to output and this is prohibitive in embedded real-time systems. In order to avoid these non-monotonic regions, a careful placement of these sensors is required. For instance, for a given sensor placement, the infrared transfer function in Fig. 5 illustrates the sensor operating in a non-monotonic range (*Region 1*). An effort to place the Hall effect sensor in a manner that guarantees monotonicity led it to saturation as shown in *Region 2* in Fig. 5. Another example of how problematic is to tweak the placement of Hall effect and infrared sensors it is shown in *Region 3* (Fig. 5). This region is monotonic and is not under saturation, however the knee-shape region requires high

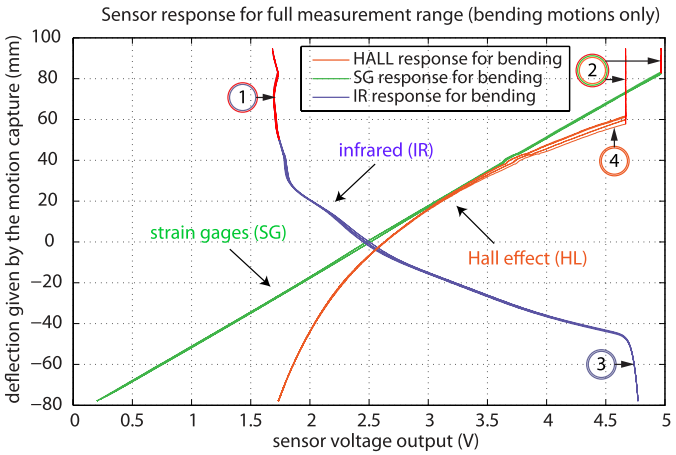

Fig. 5. Sensor response for bending motions (over multiple runs). Sensor placement is extremely important to avoid saturation, non-monotonic ranges and high order polynomial regression. **Region 1** indicates non-monotonic range; **Region 2** indicates saturation; **Region 3** requires high order polynomial regression; **Region 4** points to hysteresis in the Hall sensor response.

order polynomial regression functions to model its behaviour, therefore, increasing complexity and processing time. In short, an extremely careful placement for infrared and Hall effect sensors should take place in order to avoid issues such as non-monotonicity, saturation and high order polynomial regression.

As shown in Fig. 5, infrared and Hall effect responses are not linear in relation to the input quantity, deflection. Therefore, their sensitivity is not constant along the measurement range. In contrast, strain gages present a linear relationship between output, voltage, and input, deflection.

Strain gages measure the strain related to the stress at the measurement point. This stress derives from a force applied by the user or from the clamp reaction force. It is then expected that strain gages present distinguishable responses between driven and free motions.

A summary of the features of each of the sensors is listed as follows:

strain gages: high complexity, high cost, highly distinguishable operation between driven and free motions, linear;

infrared: non-linear, non-monotonic, sensitive to stage light, simple setup, low cost;

Hall effect: non-linear, non-monotonic, high hysteresis, simple setup, low cost.

Preliminary experiments showed that none of the sensors is optimal under all motions and they have significant performance differences between the two categorized motions. The methodology of these experiments was improved and results are reported in following session (Session III).

III. Sensor Data Conditioning

In order to design an improved instrumentation strategy, we evaluated each sensor according to the gesture performed: bending or plucking. The evaluation is based on deflection given by the sensors and on the deflection given by a reference measuring system. Sensor data and reference system are synchronized to each other. The deflection reference measuring system is a 16-camera passive infrared motion capture system

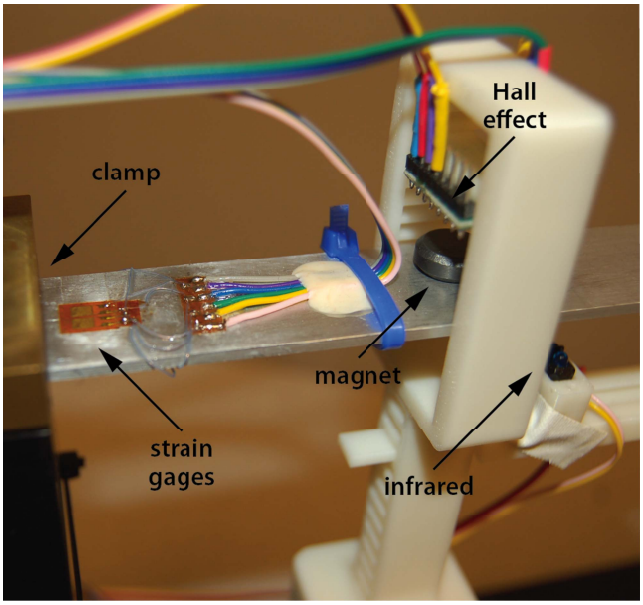

Fig. 6. Apparatus for sensor placement allows 2D positioning for infrared and Hall effect sensors: discrete positioning for vertical position and continuous positioning for horizontal position (photo by Vanessa Yaremchuk).

operating at 441 Hz. The system has a resolution of 0.01 mm with a standard deviation given by calibration of 0.31 mm. The tracking system acquired the 3D position of reflective markers placed along the beam, with an analog acquisition system registering the sensor outputs at 6615 Hz, 16 bits. The sensors are installed next to the clamp: on, below and above the beam, as shown in Fig. 6, indirectly measuring the deflection next to the free edge of the beam.

As discussed in the previous section, sensor placement is an issue for infrared and Hall sensors. In order to solve that, we designed an apparatus to optimize the placement of infrared and Hall effect sensors, limiting their measurement range to a monotonic range, free of saturation. The apparatus was built using parts manufactured by a 3D printer and LEGO© bricks as partially shown in Fig. 6. The measurement range for the strain gages was scaled to be similar to the other sensors. The curves in Fig. 7 reflect the results achieved using the apparatus for sensor placement.

A. Measurement Functions and Measurement Errors

The measurement functions are those describing the measured output quantity value as a function of the known input quantity value [33], [34]. In order to obtain them, polynomial regression is applied to the following subsets of data. The first n-length subset, $\mathbf{X}_1$, contains samples with deflection driven by the user. The samples are equally distributed in positive and negative deflections. The second n-length subset, $\mathbf{X}_2$, contains samples with free deflection. This subset excluded samples with signal-to-noise ratio (SNR) lower than 2, since lower SNR signals in the end of the oscillation tend to bring the bias to zero-mean, even if it is not accurate along most of the measurement range. The third n-length subset, $\mathbf{X}_3$, contains both motions: $n/2$ samples of $\mathbf{X}_1$ and $n/2$ samples of $\mathbf{X}_2$.

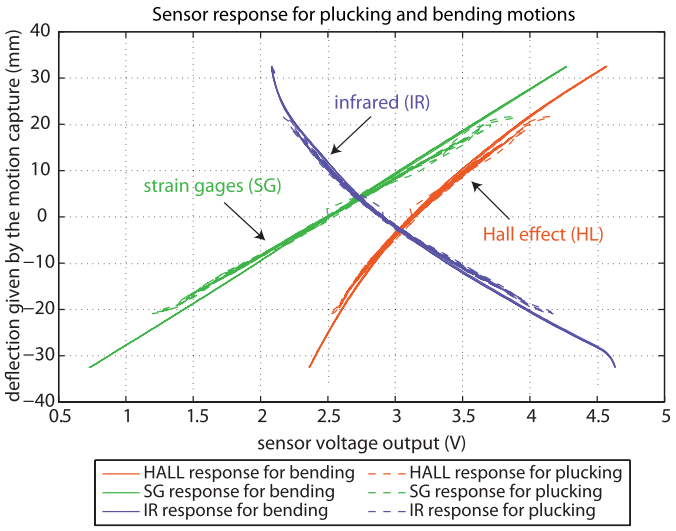

Fig. 7. Sensor response for bending and plucking motions: deflection given by the reference measuring system, in mm, *versus* the sensor output, in Volts. The use of the apparatus limits the measurement range to a monotonic range, free of saturation.

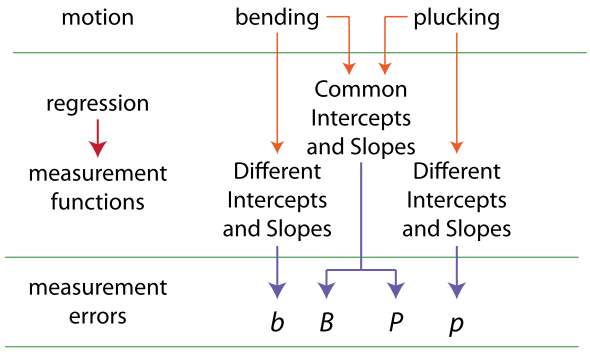

Fig. 8. Sensor evaluation method. *b*, *B*, *p*, *P* are the suffix letter to be assigned to the sensor name. *b* represents residuals for bending motions submitted to Different Slopes and Intercepts regression; *B* represents residuals for bending motions submitted to Common Slopes and Intercepts regression; *p* represents residuals for plucking motions submitted to Different Slopes and Intercepts regression; *P* represents residuals for plucking motions submitted to Common Slopes and Intercepts regression.

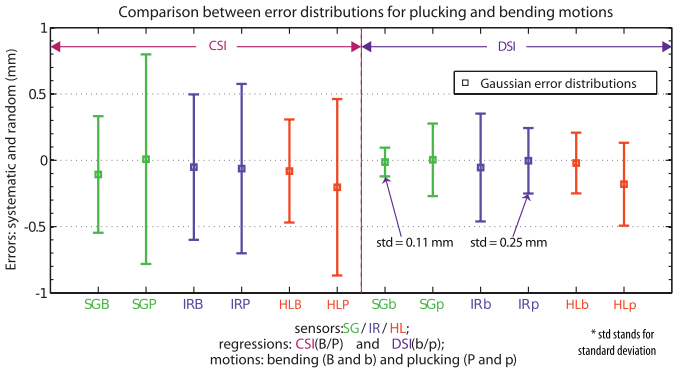

Fig. 9. Error distributions for CSI (left) and DSI (right) measurement functions, for plucking samples (*P,p*) and for bending samples (*B,b*). *b* and *p* denote errors for bending and plucking samples respectively, using DSI measurement functions. *B* and *P* denote errors for bending and plucking samples respectively, using CSI measurement functions. *std* stands for standard deviation, *CSI* stands for Common Slopes and Intercepts and *DSI* stands for Different Slopes and Intercepts. DSI measurement functions result in lower systematic and random errors.

There are two approaches for linear regression. The first approach uses *Different Slopes and Intercepts (DSI)* regression for each motion. Therefore, for each motion and its respective subset $\mathbf{X}_1$ and $\mathbf{X}_2$, there will be a *response variable* $\mathbf{Y}_1$ and $\mathbf{Y}_2$, respectively. The linear regression equations for this approach have the form:

$$\mathbf{Y}_1 = \mathbf{X}_1\,\beta_1 + \alpha_1 + \epsilon_1 \tag{2}$$

$$\mathbf{Y}_2 = \mathbf{X}_2\,\beta_2 + \alpha_2 + \epsilon_2 \tag{3}$$

where $\mathbf{Y}_1$ and $\mathbf{Y}_2$ are the response variables for bending and plucking respectively, $\mathbf{X}_1$ and $\mathbf{X}_2$ are the subsets for bending and plucking respectively, β_1 and β_2 are the slopes, α_1 and α_2 are the intercepts and ϵ_1 and ϵ_2 are the error terms.

The second approach finds a *Common Slope and Intercept (CSI)* for all motions, using the subset $\mathbf{X}_3$. The equation for this approach is presented in the form:

$$\mathbf{Y}_3 = \mathbf{X}_3\,\beta_3 + \alpha_3 + \epsilon_3 \tag{4}$$

where $\mathbf{Y}_3$ is the response variable for all motions, $\mathbf{X}_3$ is the subsets containing bending and plucking samples, β_3 is the common slope, α_3 is the common intercept and ϵ_3 is the error term.

The regression techniques result in measurement functions $\mathbf{Y}_i \approx f(\mathbf{X}_i)$. Once slopes, intercepts and errors are determined by Least Square Estimation, the measurement functions provide the deflection in millimetres ($\mathbf{Y}_i$), given the sensor output in Volts ($\mathbf{X}_i$).

The measurement errors are defined by the measured quantity value ($\mathbf{Y}_i$) minus the reference quantity value. The reference is a motion capture system providing deflection measurements. The measurement errors consist of error probability density function type, random and systematic errors. The probability density function was assumed and fit to Gaussian in order to use the measurement errors on a linear Kalman filter implementation later. The measurement errors for the following combinations of datasets and regression were analyzed (Fig. 8):

- bending motions, $\mathbf{X}_1$, submitted to DSI regression $(\beta_1, \alpha_1, \epsilon_1) \Longrightarrow$ measurement error *b*;
- plucking motions, $\mathbf{X}_2$, submitted to DSI regression $(\beta_2, \alpha_2, \epsilon_2) \Longrightarrow$ measurement error *p*;
- bending motions, $\mathbf{X}_1$, submitted to CSI regression $(\beta_3, \alpha_3, \epsilon_3) \Longrightarrow$ measurement error *B*;
- plucking motions, $\mathbf{X}_2$, submitted to CSI regression $(\beta_3, \alpha_3, \epsilon_3) \Longrightarrow$ measurement error *P*.

The Gaussian measurement errors are presented in Fig. 9. As expected, DSI regression outputs smaller measurement errors than the CSI regression. The improvement factors for the random errors range from 1.3 to 4, when using the DSI regression. This leads us to state that the sensors and the system have significantly different behaviour for plucking and bending motions. The standard error improvement is bigger for the strain gages: approximately 1.6 greater than the difference reported for Hall effect and infrared sensors.

Additional conclusions regarding the measurement errors are summarized as follows:

- for plucking motions, the infrared sensor has the lowest random error, followed by the strain gages and then by the Hall sensor;
- for bending motions, the strain gage has the lowest random and systematic errors. The second lowest random error is from the Hall effect sensor followed by the infrared sensor;
- systematic errors of Hall effect and infrared sensors vary according to the distribution of samples in the positive and negative deflection. As the dispersal of positive and negative deflection samples is impossible to estimate, these systematic errors are impossible to correct.

Furthermore, it is noticeable that all sensors differ according to motion, with respect to their error. This fact can be explored using sensor fusion, where the best features of each sensor can collaborate to provide an optimal estimate. For instance, strain gages presented the lowest random error for bending motions, while infrared sensors presented the lowest random error for plucking motions.

IV. Problem Statement

As previously discussed, the present case is suitable for a sensor fusion design. The embedded real-time employment of a DMI requires low processing cost for a sensor fusion design. Therefore, the reasonable solution is the implementation of a linear Kalman filter. However, in Kalman filter designs, the knowledge of measurement and process parameters is essential. Poor description of these parameters might lead to inaccurate estimates or even divergence problems [35]. The user-driven bending motions are not predictable, and therefore, rule out the use of physical modeling. Besides that, these motions do not respect any probability, pattern or sequence. For these cases, the range of solutions for estimating states is reduced, particularly in real-time applications where processing time is an issue.

A solution for the problem should account for the definition of a process model for unknown signals and for the evaluation of the selected process, given the danger of inaccuracy and divergence.

V. Proposed Solution

Not only the sensors but the system presents distinguishable physical behaviour under driven (bending) and free (plucking) motions. The driven motions are unpredictable while the free motion is a dampening sinusoidal waveform. Additionally, measurement functions and measurement errors vary according to the motion (Fig. 7). We propose a framework that takes advantage of the discernible physical behaviour of the sensors and the system. Therefore, for any problem where the signals are partially unpredictable, we suggest the following:

1) **motion segmentation**: identification of all known and unknown motions. Known motions are the ones eligible for physical modeling;
2) **motion classification**: clustering of known and unknown motion process models according to the segmentation;
3) **regression**: determination of different slopes and intercepts for each motion;
4) **multiple-model process model**:
   - physical modeling for known motions;
   - descriptor analyses for selecting process model for unpredictable signals;
5) **evaluation**.

To sum up, we propose the combination of motion segmentation and classification, DSI regression and multiple-model filter design. Ultimately, we predict the system behaviour using an appropriate process model during periods of free motion, while counting on a less predictive approach during the unknown user-driven regime. Additionally, we propose a method to determine and evaluate the process model for unpredictable signals based on descriptors. The evaluation of the processes using various datasets is essential to guarantee the robustness of the filter implementation, given the process model uncertainty.

VI. Filter Design

This session focuses on the design of a linear Kalman filter. A filter evaluation method is introduced based on qualitative and quantitative descriptors. Lastly, the descriptors values are used to determine the most advantageous process model for each motion.

A. Kalman Filter Design Basics

Kalman filters are based on the implementation of predictor-corrector estimators. The filter estimates the state $\mathbf{x} \in \Re^n$ given a controlled process governed by the linear stochastic difference equation [14]

$$\mathbf{x}_k = \Phi\, \mathbf{x}_{k-1} + \mathbf{B}_f\, \boldsymbol{u}_k + \boldsymbol{\omega}_{k-1}, \tag{5}$$

where $\mathbf{x}_k$ is an n-vector representing the state and its *state variables*; $\mathbf{u}_k$ is an n-vector representing the inputs to the system. The $n \times n$ matrix Φ represents the state-propagation matrix, when no driving function or process noise are considered. The $n \times n$ matrix $\mathbf{B}_f$ represents the optional control input $\mathbf{u} \in \Re^l$. The system state propagation is subjected to a noise $\boldsymbol{\omega}$.

The system is observed by a set of measurement variables forming a measurement vector $\mathbf{z}_k \in \Re^m$

$$\mathbf{z_k} = \mathbf{H}\, \mathbf{x_k} + \boldsymbol{\nu}_k. \tag{6}$$

where the $m \times n$ matrix $\mathbf{H}$ is the measurement matrix; $\mathbf{x}_k$ is the states vector and $\mathbf{z}_k$ is the measurement vector. The observation of the true quantity value of the state ($\mathbf{x}_k$) includes a noise $\boldsymbol{\nu}$.

The noises $\boldsymbol{\omega}$ and $\boldsymbol{\nu}$ represent the process and measurement noise vectors respectively. The noises are assumed to be independent from each other, white, Gaussian and bias-free, given by Equations 7 and 8,

$$p(\boldsymbol{\omega}) \sim N(0, \mathbf{Q}_k), \tag{7}$$

$$p(\boldsymbol{\nu}) \sim N(0, \mathbf{R}_k) \tag{8}$$

where $\mathbf{Q}_k$ and $\mathbf{R}_k$ are respectively the process error covariance and the measurement error covariance matrices. Equation 7 is read as the probability density function of $\boldsymbol{\omega}$ is a zero-mean Gaussian distribution which error covariance equals $\mathbf{Q}_k$. Equation 8 is read analogously. It is interesting to recall that the determination of $\mathbf{Q}_k$ and $\mathbf{R}_k$ and other measurement and process parameters should agree with the actual physical behaviour of the system and the signals. If this statement cannot be guaranteed, divergence may occur [8].

We determine the measurement error covariance matrix $\mathbf{R}_k$ according to the measurement errors described in Section III.

The lack of detailed knowledge about the statistical properties of the process model prohibits a straight forward determination of the process error covariance matrix $\mathbf{Q}_k$. First, let's consider a process model with two state variables defining a 2×2 state-propagation matrix $\boldsymbol{\Phi}$. Next, one should define which state variables are subject to noise. As an example, for the given $\boldsymbol{\Phi}$, considering the higher order state variable noisy, we define a matrix $\mathbf{Q}_e$ that specifies the presence of noise on each of the state variables. $\mathbf{Q}_e$ is given by Equation 9,

$$\mathbf{Q}_e = \boldsymbol{\Phi} \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \tag{9}$$

The process error covariance matrix $\mathbf{Q}_k$ is then given by Equation 10,

$$\mathbf{Q}_k = C_{mc} \int_0^{t_s} \boldsymbol{\Phi}(t)\, \mathbf{Q}_e\, \boldsymbol{\Phi}^T(t)\, dt \tag{10}$$

where C_{mc} is the tuning constant given by a Monte Carlo optimizer, t_s is the sampling time. The Kalman filter algorithm operates under a trade-off between process model trust and measurement trust. Given that the measurement trust is fixed and defined by the measurement errors, the process error covariance matrix expresses the trade-off between a more effective filter and a wider bandwidth filter that is able to outline disagreements of the model with the system behaviour.

B. Linear Kalman Filter Algorithm

A linear Kalman filter algorithm comprises prediction and correction. The predicted variables are called *a priori* and are denoted by a minus sign ($^-$) superscript to the variable. The corrected variables are called *a posteriori*. Predicted and corrected *estimated* variables are represented by the symbol ($\hat{}$) superscript to the variable. The subscript indices k and *k-1* define the current and previous steps respectively.

$$\hat{\mathbf{x}}_k^- = \Phi\, \hat{\mathbf{x}}_{k-1}^- + \mathbf{B}_f\, \mathbf{u}_{k-1} \tag{11}$$

$$\mathbf{P_k}^- = \Phi\, \mathbf{P}_{k-1}\, \Phi^T + \mathbf{Q}_{k-1} \tag{12}$$

$$\mathbf{K}_k = \mathbf{P}_k^-\, \mathbf{H}_k^T\, (\mathbf{H}_k\, \mathbf{P}_k^-\, \mathbf{H}_k^T + \mathbf{R}_k)^{-1} \tag{13}$$

$$\mathbf{P}_k = [\mathbf{I} - \mathbf{K}_k\, \mathbf{H}_k]\, \mathbf{P}_k^-\, [\mathbf{I} - \mathbf{K}_k\, \mathbf{H}_k]^T + \mathbf{K}_k\, \mathbf{R}_k\, \mathbf{K}_k^T \tag{14}$$

$$\mathbf{P}_k = [\mathbf{I} - \mathbf{K}_k\, \mathbf{H}_k]\, \mathbf{P}_k^- \tag{15}$$

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + \mathbf{K}_k\, (\mathbf{z}_k - \mathbf{H}_k\, \hat{\mathbf{x}}_k^-) \tag{16}$$

The implemented algorithm loop is presented from Equation 11 to Equation 16 [5]. $\hat{\mathbf{x}}_k^- \in \Re^n$ is the *a priori* state estimate at step k, $\hat{\mathbf{x}}_k \in \Re^n$ is the *a posteriori* state estimate at step k, and $\hat{\mathbf{x}}_{k-1} \in \Re^n$ is the *a posteriori* state estimate at step $k-1$. $\mathbf{P}_k^-$ is the *a priori* error covariance matrix at step k, $\mathbf{P}_k$ the *a posteriori* error covariance matrix at step k, and $\mathbf{P}_{k-1}$ the *a posteriori* error covariance matrix at step $k-1$. $\mathbf{K}_k$ is the Kalman gain and $\mathbf{I}$ denotes an identity matrix of order n. Finally, $\mathbf{z}_k$ is the measurement matrix at step k.

The Kalman gain defines the weights for each of the sensors. The error covariance matrix describes the variance of the estimate. There are several ways to calculate the error covariance matrix update: Equation 14 and Equation 15 [5]. Equation 14 requires the least computation and it is recommended when the number of measurement sources is significant less than the number of states [5]. Equation 15, referred to as *Joseph form*, has symmetric operations only and it is recommended for numerical stability. Equations with lower computational cost and higher numerical stability are essential for real-time embedded applications.

C. Filter Evaluation

A filter design depends on an accurate knowledge about the measurement sources and the process model. The measurement sources were studied thoroughly in Section III. The process model remains to be determined as it is partially unknown. The selection of a process model implies the setting of the process model state-propagation matrix ($\boldsymbol{\Phi}_k$) and the process error covariance matrix ($\mathbf{Q_k}$) (Equation 10). Our method to select a process model consists in testing process model candidates and verifying the corresponding filter design efficiency. The efficiency is described by a set of qualitative and quantitative descriptors. Ultimately, a highly efficient filter simulation, measured according to the specifications of the descriptors, is indicative of a good selection for the process model. The descriptors for the filter evaluation are defined as follows:

1) Null Process Error Covariance Matrix ($\mathbf{Q_k}$): A reasonable qualitative test to declare a process model suitable for a dataset is to analyze the estimate error while setting the process noise to zero. A process model that do not represent the physical behaviour to any extent produces a significant error if submitted to this test. This test is proposed in [36].

2) Steady-State (a posteriori) Error Covariance for Deflection Estimation ($\boldsymbol{P}_{1,1}^{ss}$): In order to obtain this descriptor, the error covariance matrix (Equation 14) in regime is selected. Then, the first element of this matrix is isolated. It corresponds to the error covariance for estimating the deflection. The steady-state value of $\mathbf{P}_{k;1,1}$ determines the stability of the filter. A stable filter should reduce its error covariance matrix elements with time, converging to a minimum value in regime [36]. From the error covariance element $\mathbf{P}_{k;1,1}$, one can derive the standard deviation of the deflection estimate. We call it *algorithm standard deviation* (e_A). This descriptor is the well-cited by several authors as standard or unique method to test filter efficiency and stability [37]–[40]. This descriptor is sometimes called *covariance performance analysis*.

3) Confidence Bounds: This descriptor discusses how well the error covariance matrix describes the actual error between

estimate and true value given by the reference. The filter algorithm describes its accuracy through the estimate error covariance matrix $\mathbf{P}_k$. However, the filter processing can report adequate $\mathbf{P}_k$ even if the estimate is diverging from the true value [36]. To account for this, we define two standard deviations. The first one is the *algorithm standard deviation (e_A)* derived from the error covariance matrix of the estimate. The second one comes from the error between the estimate and the true value given by reference measuring system. We call it *experimental standard deviation (e_E)*. Then, for each sample, we verify if the *experimental standard deviation* is within the bounds of the *algorithm standard deviation*. Finally, we compute the percentage of samples where $e_E \leq e_A$. If this percentage is 68% or higher, the error covariance matrix is accurately describing the estimate errors [36]. This descriptor is analogous to several other tests available in the literature. For instance, some authors compare the algorithm standard deviation for correct and incorrect implementation of process and models [37], [39], [40]. Other authors perform a visual comparison between the algorithm and the experimental standard deviations. This test is often referred to as a *consistency check* [35], [41]. Finally, some authors use the threshold of 68%, as is the case in this paper [36].

4) Bandwidth: Our innovative descriptor, *bandwidth*, sets a maximum trust in the sensor data. If the process model is selected properly, the filter can rely on it, reducing the bandwidth for noisy sensor data. High levels of sensor trust, indicated by high Kalman gains, reflect weak confidence in the process model. In this case, the process model is not a good guess as to the physical behaviour of the system. In order to consider a process model a good candidate for describing the system, we define that the sum of the trust of all sensors – sum of the Kalman gains for each sensors – should not be greater than a certain threshold.

The bandwidth threshold is defined as follows. The Monte Carlo optimizer stores the bandwidth descriptor value for each of its multiple runs. For each motion, the bandwidth values are fit into a Gaussian probability density function. The threshold is defined to be the mean plus one standard deviation of this distribution. For our application, the threshold for bending and plucking motions were approximate to 0.70. It can occur that the mean and standard deviation of the bandwidth value for the different motions are not similar. In this case, the filter designer should define a dedicated threshold for each motion, defined by the value of the mean plus one standard deviation.

D. Process Model Determination

The variable we are interested in estimating is deflection next to the free edge of the beam. So, the essential state variable is position. Considering that position is not constant along time, an estimate of velocity is needed. Also, an acceleration estimate might be used. Lower order filters have the advantage of converging to the steady-state error covariance faster than higher order filters. On the other hand, higher order filters tend to track higher order derivates of the estimate better than lower order filters [36].

Firstly, good candidates for process model should be hypothesized. For plucking motions, the best guess according to the physical behaviour of the beam and to the sensor data is a damped sinusoidal model. This model is based on two parameters: the undamped angular frequency (ω), the damping ratio (d). A simpler guess would be a sinusoidal model that depends on only one parameter: the undamped angular frequency (ω). These parameters are determined by physical modeling of sensor data. It seems reasonable to inquire if simpler polynomial models disregarding physical modeling would perform as well as the non-polynomial model described above.

The specification of a process model for bending motions is more complicated as there are no predictable physical behaviour, that is, duration, frequency and magnitude of the deflection are totally unknown. First, we selected between polynomial and non-polynomial models. For polynomial models, we tested first- and second-order polynomial functions. For non-polynomial models, we tested damped and undamped sinusoidal functions.

Therefore, four process models were studied for both motions: first- and second-order polynomial models and damped and undamped sinusoidal models. All candidate models were declared observable after attaining the observability rank condition [6].

All models are analyzed according to the descriptors for each of the motions. In order to fine-tune each of the 8 possible process model and motion pairs, we adapted the algorithm to run in a Monte Carlo optimizer. The optimizer consists of a uniformly distributed number of values for the tuning constant C_{mc} (Equation 10) and for the initial error covariance matrix $\mathbf{P}_0$. The initial error covariance matrix represents the confidence about the error covariance matrix in steady-state regime. Setting $\mathbf{P}_0$ as a null matrix is equivalent to expecting no errors on the estimate. Setting $\mathbf{P}_0$ with non-zero entries implies that some errors are expected in the estimate. The main diagonal of $\mathbf{P}_0$ determines the settling time for the error covariance matrix $\mathbf{P_k}$ [36].The optimization process registers the constants along with the resultant descriptors presented in Section VI-C.

VII. System Integration

In order to clarify the description of the proposed solution, this session discusses the data flow and the integration between the steps of the framework. *Offline tasks* – those to be done before the actual fusion – and *online tasks* – those performed in real-time – are described as follows.

A. Offline Tasks

Some preliminary tasks are needed before the system can be put into operation. These initial tasks are: segmenting the motions; defining the measurement functions for each sensor and motion (regression); evaluating filters in order to select the best process model $\mathbf{\Phi}_k$, process error covariance matrix $\mathbf{Q}_k$, measurement matrix $\mathbf{H}_k$ and tuning parameter C_{mc}. Fig. 10 shows the tasks and their inputs and outputs when applicable. The tasks were introduced in Section V.

Motion segmentation aims to distinguish signals that have: different sensor responses, different durations or frequencies,

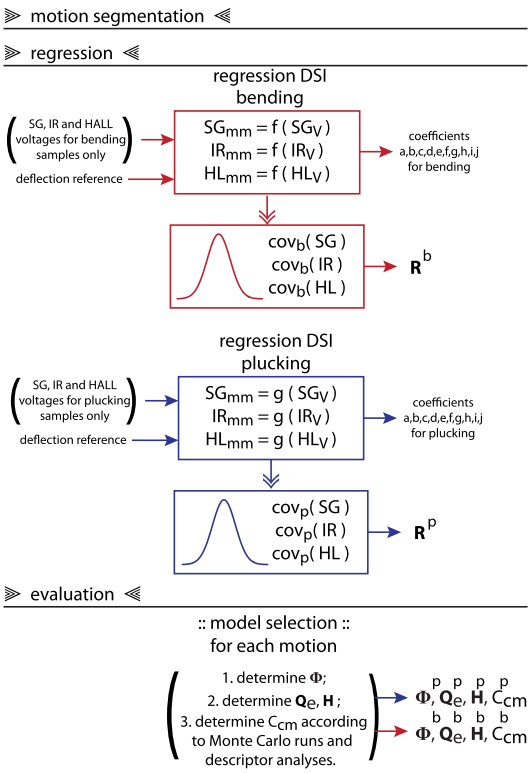

Fig. 10. Offline tasks: motion segmentation; DSI regression for each motion and process model selection for each motion. The iteration denotation k was excluded for simplicity.

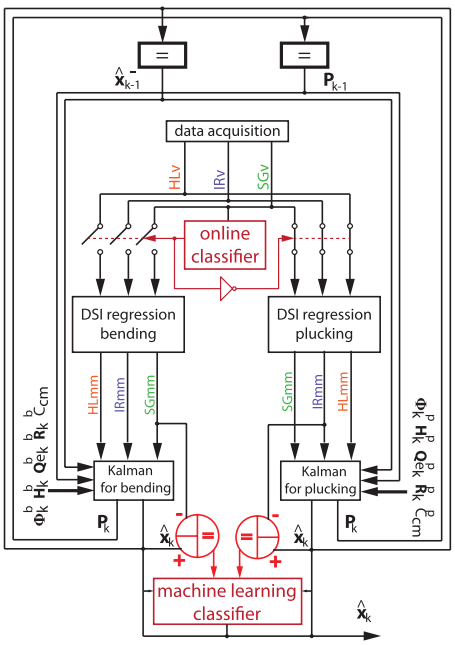

Fig. 11. Possible classifier applications: *online classifier* or *machine learning classifier*. $\hat{\mathbf{x}}_k$ is the *a posteriori* state estimate at step k, $\hat{\mathbf{x}}_{k-1}$ is the *a posteriori* state estimate at step $k-1$, $\mathbf{P}_k$ is the error covariance matrix at step k, $\mathbf{P}_{k-1}$ is the error covariance matrix at step $k-1$, Φ is the state-propagation matrix, $\mathbf{Q}_{e\,k}$ is the process error covariance matrix, $\mathbf{H}_k$ is the measurement matrix, $\mathbf{R}_k$ is the measurement error covariance matrix and C_{mc} is the tuning constant. *SGv, IRv, HLv* indicate the digitized sensor output in Volts, while *SGmm, IRmm, HALLmm* indicate the sensor data in millimetres. Indices b refer to parameters or data for bending motions, while indices p refer to parameters or data for plucking motions.

different complexity levels and finally different knowledge about the state variables with which it may be described. Prior knowledge of the nature of the underlying signals, direct observation of the signal, signal processing techniques (including frequency analysis, normalization, derivative calculation) and machine learning tool for clustering are useful techniques to carry out motion segmentation.

The regression task uses one motion at a time to define a measurement function for each sensor through linear regression. One outcome of the regression task is the coefficients for the polynomial measurement functions per sensor and motion. The other outcome is the definition of the regression errors in comparison to the deflection reference. These errors are fitted to a Gaussian distribution and will form the measurement error covariance matrix $\mathbf{R}_k$.

The evaluation tasks begins by selecting process model candidates. The candidates are defined by physical modelling of the signals when possible. Process model candidates $\mathbf{\Phi}_k$, process error covariance matrix $\mathbf{Q}_K$, measurement matrix $\mathbf{H}_k$ are fed in Monte Carlo runs. The output of the Monte Carlo optimization are the descriptors values (Section VI-C) and consequently the tuning parameter C_{mc}.

B. Classifier

There are two possible topologies to integrate the gesture classification and the Kalman filter implementation (Fig. 11). The first topology is based on setting a classifier immediately after gesture acquisition. The classification output activates either the Kalman filter for bending or the Kalman filter for plucking. The classifier output also selects the measurement functions (coefficients *a,b,...,j*), the measurement error covariances matrix $\mathbf{R_k}$, the state-propagation matrix $\mathbf{\Phi}_k$, the process error covariance matrix $\mathbf{Q_k}$, the measurement matrix $\mathbf{H_k}$ and the tuning constant C_{mc}. The sensor data, the *a posteriori* state estimate at step $k-1$ ($\hat{\mathbf{x}}_{k-1}$) and the *a posteriori* error covariance matrix at step $k-1$ ($\mathbf{P}_{k-1}$) are sent to the selected filter. We call this topology the *online classifier*. This classifier is designed using cross-zero detection, logic and first-derivative analysis.

The second topology runs the Kalman filters for bending and plucking in parallel, without any previous gesture classification. The output of both filters are compared to the best sensor output per motion: strain gages for bending motion and infrared for plucking motions. In summary, the residuals between the two filters and the two sensors are calculated. These residuals train a layer recurrent neural network with five hidden units. For each new dataset, the neural network produces alternating affirmative and negative answers. The affirmative output is the right match between motion and filter. We call this topology the *machine learning classifier*.

The *online classifier* is simpler and requires less processing time. The *machine learning classifier* is more accurate. However, it requires learning and more processing time. In this work, the *online classifier* is used due to simplicity and to meet

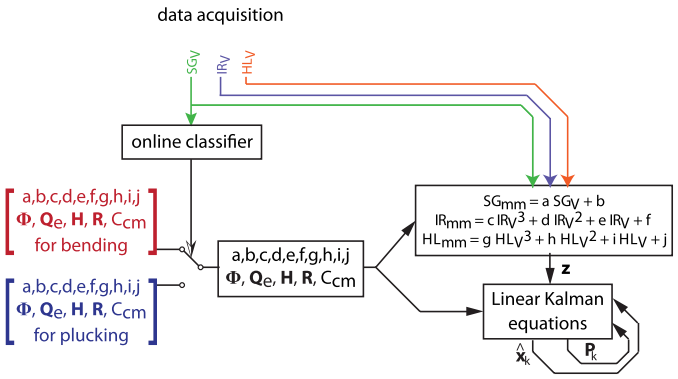

Fig. 12. Online tasks: classification; selection of parameters according to the classifier output; linear Kalman filter loop. The parameters to be selected according to the classifier output are: coefficients for the measurement functions, the state-propagation matrix Φ, the process error covariance matrix $\mathbf{Q}_e$, the measurement matrix $\mathbf{H}$, the measurement error covariance matrix $\mathbf{R}$ and the tuning constant C_{mc}. The iteration denotation k was excluded for simplicity.

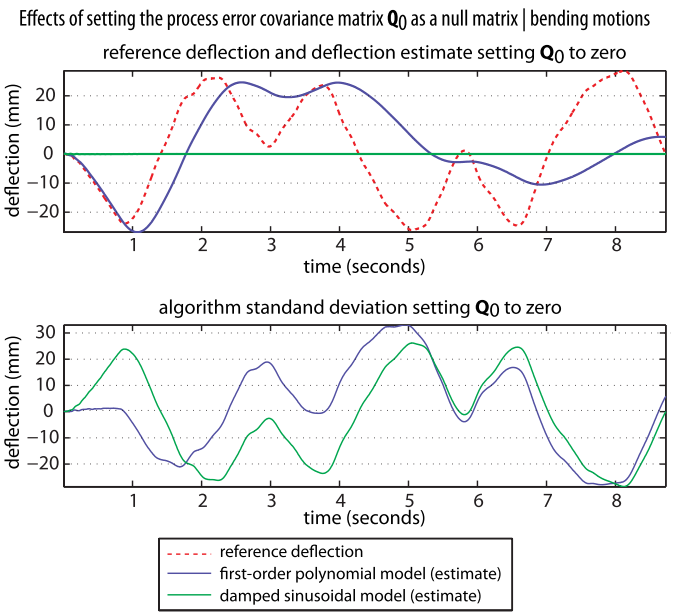

Fig. 13. Effects of setting the *process error covariance matrix* as a null matrix, for **bending** motions. Top graph displays the deflection given by the reference measuring system, the deflection estimate given by a first-order polynomial filter and by a damped sinusoidal filter (which does not estimate any deflection). The bottom graph compares the algorithm standard deviation for both model settings.

the time requirements for embedded real-time applications. The integration filter-classifier for both approaches is shown in Fig. 11.

1) Online Classifier Implementation: The *online classifier* simply consists of two tests which examine the frequency content of both types of motion. While plucking presents higher constant frequency, bending has variable lower frequency. The first test is to count the distance between the samples that cross zero. Exhaustive tests show that users do not bend the beam in opposite directions passing by the rest position faster than the plucking oscillations. The second test involves calculating the derivative of the signal by subtracting the current and the previous sample values. This simplified derivative operates as a high pass filter, attenuating the slow rate of change in bending motions. For plucking motions, the derivative output is an attenuated version of the deflection oscillations, maintaining the same frequency content. The plucking derivative has a considerably higher magnitude than the bending derivative.

C. Online Tasks

The online tasks are performed in real-time and consist of data acquisition, classification and Kalman filter loop (Fig. 12). Right after the data acquisition, the strain gage voltage output is used as input for the *online classifier* described in the previous section.

The strain gages' value in Volts is used for the following reasons:

- as the transfer function between the strain value in Volts and in millimetres is linear, the value in Volts can be used for classification without requiring a CSI regression;
- the strain gage is the sensor type that presents greater difference between bending and plucking motions (Section III-A), concerning the measurement errors.

The classifier output will select the parameters calculated during the *offline tasks* according to the motion. The coefficients *a,b,...,j* are used to define the measurement vector $\mathbf{z}$. The remaining parameters are used in the filter algorithm. The state estimate vector and the error covariance matrix are fed into the system at every new innovation.

VIII. Results

This session presents the descriptor values for each process model and motion pair, culminating in the selection of the best process model for both motions: bending and plucking. Some descriptors are used to disqualify non-efficient or incoherent simulation runs. Other descriptors are used to find the best tuning parameter for each process model and motion pair.

Next, we display the improvement on the instrumentation design using segmented gestures multiple-model sensor fusion as opposed to a single-sensor and single-model instrumentation approaches.

A. Process Model Selection

Firstly, we examine the qualitative descriptor where the process error covariance matrix is set to a null matrix. Fig. 13 and Fig. 14 show examples of process model comparison for bending and plucking motions, setting the process error covariance to zero.

Clearly, polynomial models are the better for bending motions, while non-polynomial models are better for plucking motions. For instance, Fig. 13 shows that the damped sinusoidal model does not estimate any deflection for bending. For plucking motions, as shown in Fig. 14, it is clear that the damped sinusoidal model is the most suitable process model for the motion, presenting small deflection estimate

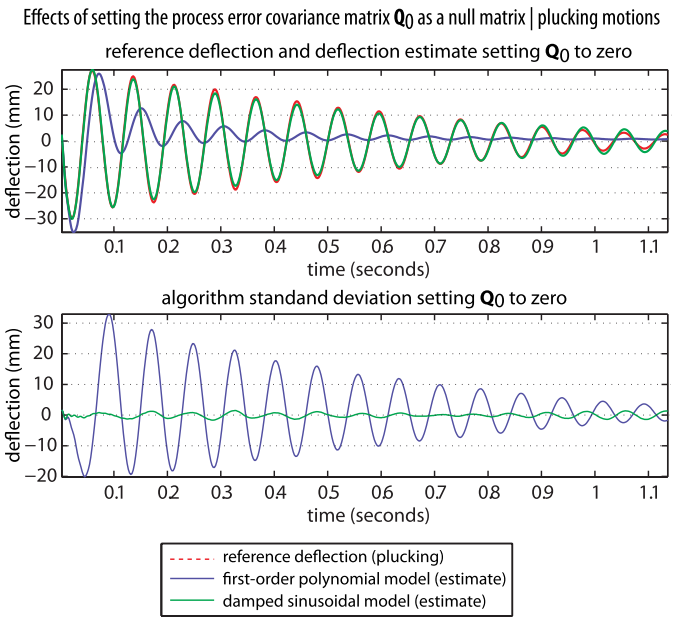

Fig. 14. Effects of setting the *process error covariance matrix* as a null matrix, for **plucking** motions. Top graph displays the deflection given by the reference measuring system, the deflection estimate given by a first-order polynomial filter and by a damped sinusoidal filter. The bottom graph compares the algorithm standard deviation for both model settings.

TABLE I
COMPARISON BETWEEN CANDIDATE PROCESS MODELS FOR BENDING MOTIONS. BOLD FONT REPRESENTS THE BEST RESULTS

$\Phi_{BENDING}$		$\mathbf{P}^{ss}_{1,1}$ $[10^{-3}\ mm^2]$	bandwidth
first-order	noisy vel.	**5.4**	**0.69**
	noisy vel. pos.	6.4	0.71
second-order		6.6	0.74
sinusoidal		8.7	*0.96*
damped sinusoidal		8.5	*0.95*

error (bottom graph), while the first-order polynomial model is not capable of tracking the deflection variation.

Additionally, we impose a minimum value of 68% for the *confidence bounds* descriptor, as mentioned in Session VI-C. It means that only optimizer trials that accomplish this target are going to be evaluated. Lastly, we analyze the *steady-state error covariance for deflection estimate* ($\boldsymbol{P}^{ss}_{1,1}$) and the bandwidth.

1) Bending Motions: A comparison between process models for bending motions reports no significant difference between first- and second-order polynomial filters. As lower order filters are faster and as we are interested in position estimate and not in its derivatives, we selected the first-order filter. Two types of first-order filter were evaluated: one where only velocity is considered noisy (denoted by *noisy vel.*) and other where velocity and deflection are considered noisy (denoted by *noisy vel. pos.*). Refer to Table I for descriptors results in bending motions.

One might think that the non-polynomial filters are comparable to the polynomial ones since their error covariance

TABLE II
COMPARISON BETWEEN CANDIDATE PROCESS MODELS FOR PLUCKING MOTIONS. BOLD FONT REPRESENTS THE BEST RESULTS

$\Phi_{PLUCKING}$		$\mathbf{P}^{ss}_{1,1}$ $[10^{-3}\ mm^2]$	bandwidth
first-order		17.6	0.70
second-order		15.1	0.60
sine	process model error	6.3	0.25
	process model as in 1^{st} order filter	6.2	0.25
damped sinusoidal		**4.8**	**0.19**

for deflection estimation $\mathbf{P}^{ss}_{1,1}$ is similar. However, the non-polynomial filters make significantly use of the sensor data, that is, have a higher bandwidth (*italicized* in Table I), which reflects a low trust in the process model selected.

2) Plucking Motions: A comparison between process models for plucking motions (Table II), it reveal a significant difference on performance between polynomial and non-polynomial filters. The non-polynomial filters present the best performance not only concerning the steady-state covariance for deflection estimation but also the bandwidth.

Two choices of process error covariance matrix $\mathbf{Q}_k$ (Equation 10) for sinusoidal filters were tested. The first one is based on the actual state-propagation matrix $\boldsymbol{\Phi}$ for sinusoidal model. The alternative simplified $\mathbf{Q}_k$ is based on a first-order polynomial state-propagation equation. There were no significant advantages on using the simplified $\mathbf{Q}_k$ concerning any of the descriptors, except by its faster processing. Finally, the best option is the damped sinusoidal filter, matching the physical modeling of the signals.

It is remarkable the bandwidth reduction obtained due to a better description of the process model. It can be seen that the bandwidth value of *0.69* was obtained for the bending filter, whereas a bandwidth of *0.19* was obtained for the plucking filter. This strengthens the idea that the correct knowledge of the process model is essential. This work proposes that, in the case of partial knowledge of the process model, motion segmentation and multiple-models filter should be implemented in order to reduce the bandwidth for noise whenever the process model is known accurately.

B. Selected Process Model and Motion Pairs

Given the results presented in Tables I and II, we introduce the state-propagation matrices that best describe the system behaviour while in the bending or plucking regimes.

For bending motions, the best model is the first-order polynomial model. Its space-state representation ($\boldsymbol{\Phi}^b$) is presented in Equation 17:

$$\boldsymbol{\Phi}^b = \begin{bmatrix} 1 & t_s \\ 0 & 1 \end{bmatrix} \tag{17}$$

where t_s is the sampling time.

For plucking motions, the best model is the damped sinusoidal model described by its space-state representation (from

TABLE III
FILTER PERFORMANCE IN COMPARISON TO THE BEST SENSOR FOR EACH MOTION. BOLD FONT REPRESENTS THE BEST RESULTS

filter *versus* single-sensor approach		error covariance for deflection $[10^{-3}\ mm^2]$
bending	filter	**5.4**
	SG	12
plucking	filter	**4.8**
	IR	61

TABLE IV
FILTER PERFORMANCE IN COMPARISON TO THE SINGLE-MODEL APPROACH. BOLD FONT REPRESENTS THE BEST RESULTS. *Italicized* VALUES REPRESENT THE BANDWIDTH FOR EACH CONDITION

filter *versus* single-model approach	error covariance for deflection $[10^{-3}\ mm^2]$
implemented filter	**4.8** (*0.19*) to **5.4** (*0.69*)
single-model approach	4.8 (*0.19*) to 8.5 (*0.95*)

Equation 18 to Equation 22):

$$\mathbf{\Phi}^p_{11} = e^{d\,w\,t_s}\,(-d\,w\,sin(b\,t_s) + b\,cos(b\,t_s)) \tag{18}$$

$$\mathbf{\Phi}^p_{12} = e^{d\,w\,t_s}\,sin(b\,t_s) \tag{19}$$

$$\mathbf{\Phi}^p_{21} = -w^2\,e^{d\,w\,t_s}\,sin(b\,t_s) \tag{20}$$

$$\mathbf{\Phi}^p_{22} = e^{d\,w\,t_s}\,(d\,w\,sin(b\,t_s) + b\,cos(b\,t_s)) \tag{21}$$

$$\mathbf{\Phi}^p = \frac{1}{b}\begin{bmatrix}\mathbf{\Phi}^p_{11} & \mathbf{\Phi}^p_{12}\\ \mathbf{\Phi}^p_{21} & \mathbf{\Phi}^p_{22}\end{bmatrix} \tag{22}$$

where w is the undamped angular frequency, d is the damping ratio and b is the underdamped angular frequency given by $b = w\sqrt{1-d^2}$.

C. Sensor Fusion Contribution

In order to make clear the advantages of using fusion techniques in the present problem, we compared the filter output for each motion with the use of individual sensors, in terms of the error covariance for deflection. Table III compares the best filter and the best sensor for each motion. The contribution of fusing data from different sensors varies according to the motion regime. For both motions, the error covariances are better than any of the sensors individually. The improvement factor for bending motions is 2.2 and for plucking motions is 12.7.

Note that the lowest measurement error covariance for plucking motions ($61.\,10^{-3}$ mm^2) is almost five times worse than the lowest measurement error covariance for bending. Lastly, the most predictive physical behaviour is the one in which the sensors performance is the worst. Consequently, *The Rulers* is a good testbed to demonstrate that the proposed framework is satisfactory.

Additionally, it is worth to compare the proposed filter to a single-model filter approach. In case of forcing a single-model, the most advantageous model would be the one that reflects, at least partially, the physical behaviour of the system: damped sinusoidal function. The comparison is shown in Table IV. Even if the error covariance ranges are not significantly different, the bandwidth for the noisy sensor data is high, reflecting a disagreement between the selected process model and the motion being performed. Therefore, it is possible to conclude that the use of multiple-model filter improves not only the error covariance for deflection but the bandwidth.

IX. CONCLUSION

We have presented a sensor fusion technique based on a multiple-model linear Kalman filter for deflection estimation using strain gages, infrared and Hall effect sensors. The system of clamped beams has two distinguishable physical behaviours: a free damping motion (called plucking) and a driven motion controlled by the user (called bending). The problem lies in the unpredictability of bending motions, which make impossible the physical modeling of this motion.

Our solution derives from segmenting the motions according to the knowledge of their physical model, that is, according to their eligibility for physical modeling. Then, a classifier defines the gesture being performed and activates the correct measurement and process model parameters accordingly. This approach makes possible the design of a more predictive filter whenever the process model is known, i.e. free motions; while reducing the filter efficiency when the process model is impossible to be determined, i.e. user-driven motions.

The problem of selecting a process model for unknown signals is solved by defining a set of descriptors that evaluate candidate process models. Forcing threshold values for some of the descriptors exclude trials with low efficiency and/or high risk of divergence. The remaining descriptors are used to select the model parameters. Therefore, this framework facilitates the process model selection for unknown signals and evaluates the robustness of the filter design, inherently at high-risk due to the limited knowledge of the process model.

Through experiments, it was shown that the suggested framework results in improved error estimate covariance for deflection compared to the measurement error covariance of any of the sensors individually. The application of fusion filters in the present case is appropriate, since the worst measurement errors happen for plucking gestures, whose process model is known.

We recommend the application of the framework in cases where one or more of the conditions apply:

- there are signals which process model determination is unclear;
- the system has more than one operation mode and there is no clear transition between the modes;
- measurement functions and errors differ considerably between the operation modes;
- a better error covariance for the known signals estimate is desired, without compromising the estimate of unknown signals;
- a better error covariance for the unknown signals estimate is desired.

ACKNOWLEDGMENT

The first author would like to thank Capes/Brazil for the doctoral scholarship. The authors would like to thank BRAMS for the use of their laboratory facilities and S. Sinclair, M. Aldrich and C. Bronson for suggestions.

REFERENCES

[1] E. R. Miranda and M. M. Wanderley, *New Digital Musical Instruments: Control and Interaction Beyond the Keyboard*. Middleton, WI, USA: A-R Editions, 2006.

[2] J. Malloch, S. Sinclair, and M. M. Wanderley, "A network-based framework for collaborative development and performance of digital musical instruments," in *Proc. Comput. Music Modeli. Retr.*, 2008, pp. 401–425.

[3] C. B. Medeiros and M. M. Wanderley, "Evaluation of sensor technologies for the rulers, A Kalimba like digital musical instrument," in *Proc. 8th Sound Music Comput. Conf.*, Jan. 2011, pp. 1–8.

[4] X. Yun and E. Bachmann, "Design, implementation, and experimental results of a quaternion-based Kalman filter for human body motion tracking," *IEEE Trans. Robot.*, vol. 22, no. 6, pp. 1216–1227, Dec. 2006.

[5] J. A. Farrell, *Aided Navigation: GPS with High Rate Sensors*. New York, NY, USA: McGraw-Hill, 2008.

[6] J. Raol, *Multi-Sensor Data Fusion With MATLAB*. Cleveland, OH, USA: CRC Press, Dec. 2010.

[7] G. C. Goodwin and K. S. Sin, *Adaptive Filtering Prediction and Control*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1984.

[8] D.-J. Jwo, F.-C. Chung, and T.-P. Weng, *Sensor Fusion and its Applications*. England, U.K.: InTech, 2010.

[9] C. Hu, W. Chen, Y. Chen, and D. Liu, "Adaptive Kalman filtering for vehicle navigation," *J. Global Positioning Syst.*, vol. 2, no. 1, pp. 42–47, Nov. 2003.

[10] Q. Xia, M. Rao, Y. Ying, and X. Shen, "Adaptive fading kalman filter with an application," *Automatica*, vol. 30, no. 8, pp. 1333–1338, 1994.

[11] C. Hide, T. Moore, and M. Smith, "Adaptive kalman filtering for low-cost INS/GPS," *J. Navigat.*, vol. 56, pp. 143–152, Jan. 2003.

[12] K. P. Mohamed and A. H. Schwarz, "Adaptive Kalman filtering for INS/GPS," *J. Geodesy*, vol. 73, no. 4, pp. 193–203, Dec. 1999.

[13] D.-J. Jwo and S.-H. Wang, "Adaptive fuzzy strong tracking extended Kalman filtering for GPS navigation," *IEEE Sensors J.*, vol. 7, no. 5, pp. 778–789, May 2007.

[14] S. Haykin, *Kalman Filtering and Neural Networks*. New York, NY, USA: Wiley, Oct. 2001, pp. 1–22.

[15] F. Janabi-Sharifi and M. Marey, "A Kalman-filter-based method for pose estimation in visual servoing," *IEEE Trans. Robot.*, vol. 26, no. 5, pp. 939–947, Oct. 2010.

[16] Y. Bar-Shalom, T. Kirubarajan, and X.-R. Li, *Estimation with Applications to Tracking and Navigation*. New York, NY, USA: Wiley, Jun. 2001.

[17] P. de Vallière and D. Bonvin, "Application of estimation techniques to batch reactor—III. Modelling refinements which improve the quality of state and parameter estimation," *Comput. Chem. Eng.*, vol. 14, no. 7, pp. 799–808, 1990.

[18] M. Agarwal and D. Bonvin, "Improved state estimator in the face of unreliable parameters," *J. Process Control*, vol. 1, no. 5, pp. 251–257, 1991.

[19] D. Dochain, "State and parameter estimation in chemical and biochemical processes: A tutorial," *J. Process Control*, vol. 13, no. 8, pp. 801–818, 2003.

[20] L. Ljung, "Asymptotic behaviour of the extended kalman filter as a parameter estimator for linear system," *IEEE Trans. Autom. Control*, vol. 24, no. 1, pp. 36–50, May 1979.

[21] J. Kramer and A. Kandel, "Robust small robot localization from highly uncertain sensors," *IEEE Trans. Syst., Man, Cybern. Cybern., Part C: Appl. Rev.*, vol. 41, no. 4, pp. 509–519, Jul. 2011.

[22] E. Mazor, A. Averbuch, Y. Bar-Shalom, and J. Dayan, "Interacting multiple model methods in target tracking: A survey," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 34, no. 1, pp. 103–123, Jan. 1998.

[23] X. Rong Li and Y. Zhang, "Multiple-model estimation with variable structure. V. Likely-model set algorithm," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 36, no. 2, pp. 448–466, Apr. 2000.

[24] M. Gabrea, "Robust adaptive Kalman filtering-based speech enhancement algorithm," in *Proc. IEEE ICASSP*, vol. 1. May 2004, pp. 301–304.

[25] E. A. Wan and R. van der Merwe, *Kalman Filter and Neural Networks*. New York, NY, USA: Wiley, Oct. 2001, pp. 175–220.

[26] S. Roweis and Z. Ghahramani, *Kalman Filter and Neural Networks*. New York, NY, USA: Wiley, Oct. 2001, pp. 175–220.

[27] A. Zia, T. Kirubarajan, J. Reilly, D. Yee, K. Punithakumar, and S. Shirani, "An EM algorithm for nonlinear state estimation with model uncertainties," *IEEE Trans. Signal Process.*, vol. 56, no. 3, pp. 921–936, Mar. 2008.

[28] J. Malloch, D. Birnbaum, E. Sinyor, and M. M. Wanderley, "A new conceptual framework for digital musical instruments," in *Proc. Magn. Conf.*, Sep. 2006, pp. 49–52.

[29] R. Pallás-Areny and J. G. Webster, *Sensors and Signal Conditioning*, 2nd ed. Cranbury, NJ, USA: Barnes, Nov. 2000.

[30] Vishay Semiconductors, Malvern, PA, USA. (2012, Jul.). *TCRT5000 Datasheet* [Online]. Available: http://www.vishay.com/docs/83760/tcrt5000.pdf

[31] Analog Devices, Norwood, MA, USA. (2012, Jul.). *AD22151 Datasheet* [Online]. Available: http://www.analog.com/static/imported.../data_sheets/AD22151.pdf

[32] Honeywell, Morristown, NJ, USA. (2012, May). *Hall Effect Sensing and Application* [Online]. Available: http://sensing.honeywell.com/index.php?ci_id=47847

[33] *Evaluation of Measurement Data Ñ Guide to the Expression of Uncertainty in Measurement*, Working Group 1 of the Joint Committee for Guides in Metrology, Marquez, TX, USA, 2008.

[34] *Vocabulary of Basic and General Terms in Metrology (VIM)*, Joint Committee for Guides in Metrology, Marquez, TX, USA,2012.

[35] D.-J. Jwo and T.-S. Cho, "A practical note on evaluating Kalman filter performance optimality and degradation," *Appl. Math. Comput.*, vol. 193, no. 2, pp. 482–505, 2007.

[36] P. Zarchan and H. Musoff, *Fundamentals of Kalman Filtering: A Practical Approach*, 3rd ed., P. Zarchan, Ed. San Diego, CA, USA: Amer. Inst. Aeronautics Astronautics, Jan. 2009.

[37] A. Gelb, *Applied Optimal Estimation*. Cambridge, MA, USA: MIT Press, 1974.

[38] B. P. Gibbs, *Advanced Kalman Filtering, Least-Squares and Modeling: A Practical Handbook*. New York, NY, USA: Wiley, 2011.

[39] P. S. Maybeck, *Stochastic Models, Estimation, and Control*, vol. 1. San Diego, CA, USA: Academic Press, 1979.

[40] P. Maybeck, *Stochastic Model, Estimation, and Control*. vol. 2. San Diego, CA, USA: Academic Press, 1982.

[41] D.-J. Jwo, "Remarks on the Kalman filtering simulation and verification," *Appl. Math. Comput.*, vol. 186, no. 1, pp. 159–174, 2007.

Carolina Brum Medeiros was born in Pelotas, Brazil. She received the B.S. degree in electrical engineering in 2006, and the M.S. degree in mechanical engineering in 2009 from the Universidade Federal de Santa Catarina, Florianópolis, Brazil. She is currently working towards the Ph.D. degree with the Music Technology Program at Schulich School of Music. She is also a Research Assistant at Input Devices and Music Interaction Laboratory (IDMIL), McGill University, Montreal, Canada.

In 2013, she was a Visiting Student with the Responsive Environments Group, MIT Media Laboratory. Her research interest includes sensor fusion techniques, sensor and signal conditioning, user interfaces, and human motion analyses.

Ms. Brum Medeiros is a recipient of a full doctoral scholarship from Capes/Brazil. She is a member of the Centre for Interdisciplinary Research in Music Media and Technology (CIRMMT) and CREATE-NSERC Integrated Sensor Systems Program.

Marcelo M. Wanderley received the Ph.D. degree from the Université Pierre et Marie Curie (Paris VI), France, in acoustics, signal processing, and computer science applied to music. He is currently a William Dawson Scholar and an Associate Professor of Music Technology at the Schulich School of Music, McGill University, Montreal, where he directs the Input Devices and Music Interaction Laboratory and the Centre for Interdisciplinary Research in Music Media and Technology.

He was Chair of the 2003 International Conference on New Interfaces for Musical Expression and co-authored the textbook *New Digital Musical Instruments: Control and Interaction Beyond the Keyboard* (A-R Editions). His main research interests include gestural control of sound synthesis, input device design and evaluation, and the use of sensors and actuators in digital musical instruments.

Dr. Wanderley is senior member of the Association for Computing Machinery, and a member of the Acoustical Society of America and the International Computer Music Association.