Hindawi Publishing Corporation
International Journal of Navigation and Observation
Volume 2010, Article ID 869127, 10 pages
doi:10.1155/2010/869127

Research Article

Accuracy Improvement by Boundary Conditions for Inertial Navigation

Tuukka Nieminen, Jari Kangas, Saku Suuriniemi, and Lauri Kettunen

Tampere University of Technology, Department of Electronics, Unit of Electromagnetics, P.O. Box 692, 33101 Tampere, Finland

Correspondence should be addressed to Tuukka Nieminen, tuukka.nieminen@tut.fi

Received 16 September 2009; Revised 8 January 2010; Accepted 12 May 2010

Academic Editor: Paul Cross

The term *inertial navigation* is often automatically associated with the term *initial value problem*. However, there are many applications where it is possible to end up with a *boundary value problem* (BVP) as well. We show that in case of a BVP, the *finite element method* that incorporates *boundary conditions* can be efficiently used to compute position and velocity estimates not prone to accumulation of errors. For further accuracy enhancements, a method of combining inertial measurements with additional constraints is proposed. This way, we can model sensor errors, known to limit the accuracy of the system. The capabilities of the proposed methods are demonstrated with real-life examples.

1. Introduction

Typically, inertial measurements are made to have estimates of current position and velocity in real time. The set of equations used to compute the position and velocity estimates out of the actual measurements depends greatly on the application in hand. Equations for a general navigation application are presented, for example, in [1] and [2–5]. In this study, however, we shall concentrate on the growing market of *consumer applications* employing inertial sensors within a suitable price (and quality) range. Thus, we can simplify the equations needed to compute the *position and velocity estimates*. This is so, because in this case the errors are more likely to be determined by the limited accuracy of the sensors rather than the accuracy of the used equations.

In terms of the simplified equations, we basically need to solve the following problem: given $\mathbf{a}(t) : \mathbb{R} \rightarrow \mathbb{R}^3$, find $\mathbf{r}(t) : \mathbb{R} \rightarrow \mathbb{R}^3$ such that $\mathbf{r}$ satisfies

$$\ddot{\mathbf{r}}(t) = \mathbf{a}(t). \tag{1}$$

In (1), "¨" represents the second time derivative, $\mathbf{r}(t)$ is the position of the object in a suitable coordinate frame, $\mathbf{a}(t)$ represents the acceleration of the object in the same coordinate frame, and t is time. The velocity of the object ($\dot{\mathbf{r}}$), given in the same coordinate frame as $\mathbf{a}(t)$ and $\mathbf{r}(t)$, is denoted as $\mathbf{v}(t)$. Notice that the form (1) follows from the more general navigation situation by neglecting the rotation and "curvature" of the Earth.

Now, from the theory of ordinary differential equations (ODEs), we know that we need exactly two independent constraints to unambiguously solve a 2nd order ODE (1). Particularly, if these two independent constraints are both given at time $t = T_0$, we have an *initial value problem* (IVP) [6]: given $\mathbf{a}(t)$, $\mathbf{v}(T_0)$, *and* $\mathbf{r}(T_0)$, find $\mathbf{r}(t)$ such that

$$\begin{aligned} \ddot{\mathbf{r}}(t) &= \mathbf{a}(t), \\ \mathbf{v}(T_0) &= \mathbf{v}_0, \\ \mathbf{r}(T_0) &= \mathbf{r}_0 \end{aligned} \tag{2}$$

hold. Problems that are not IVPs based on the above definition are called *boundary value problems* (BVPs). A special case of a BVP, convenient to our purposes, is stated as follows: given $\mathbf{a}(t)$, one of the constraints

$$\mathbf{v}(T_0) = \mathbf{v}_0 \quad \text{or} \quad \mathbf{r}(T_0) = \mathbf{r}_0, \tag{3}$$

and one of the constraints

$$\mathbf{v}(T_1) = \mathbf{v}_1 \quad \text{or} \quad \mathbf{r}(T_1) = \mathbf{r}_1, \tag{4}$$

find $\mathbf{r}(t)$ for $T_0 \leq t \leq T_1$ such that it satisfies

$$\ddot{\mathbf{r}}(t) = \mathbf{a}(t),$$
$$\mathbf{v}(T_0) = \mathbf{v}_0, \quad \text{or} \quad \mathbf{r}(T_0) = \mathbf{r}_0, \tag{5}$$
$$\mathbf{v}(T_1) = \mathbf{v}_1 \quad \text{or} \quad \mathbf{r}(T_1) = \mathbf{r}_1.$$

We know for a fact that a problem of the form (5) has a unique solution whenever the position is fixed at least at one boundary: This knowledge comes from the fact that (7) is a one-dimensional case of a certain class of partial differential equations called Poisson problems, properties of which are well known [7].

Notice that there is a natural reason why problems (2) and (5) are distinguished. Namely, the solution methods for IVPs and BVPs differ substantially from each other [6, 8]. This is also where this paper differs from numerous research articles considering inertial navigation: previously, the problems have been given in the form (2), whereas we consider problems of the form (5).

The key assumption of our approach is the following: *inertial measurements related to a certain time period, including a set of boundary values and possibly some additional constraints, are all available when processed.* The length of the time period (i.e., *domain*) $[T_0, T_1]$ in (5) can be anything from fractions of a second to some minutes. There are different ways we may end up with a BVP of the form (5): firstly, the underlying problem may naturally be a BVP. Secondly, we may have an IVP, which we can pose as a BVP. This, of course, requires that we also know something about the result at the end of the interesting event and that we can afford to wait for the results until the end of the event. In the first case, we generally do not have any additional constraints we could use, but in the second case we end up with a BVP and at least one additional constraint. We will discuss examples of both cases in the next section.

The case where we only know the velocity at both ends is, however, special and deserves some attention. According to the discussion above, we do not have a unique solution for this kind of a BVP although it fits the definition of our model BVP (5). It is neither a valid IVP according to (2). As it turns out, it is possible to solve also this without loss of generality. This is based on the fact that we can always fix the position at one boundary without changing the "shape" of the solution. Then, we will come up with a well-defined BVP with an additional constraint.

The overall scope of this paper is to show how to treat inertial navigation problems that are naturally (or knowingly) posed as a BVP of the form (5). Attitude computations are not considered, and where necessary, attitude is assumed to be available. The form of problem (5) allows us to consider it as a set of three one-dimensional (1D) problems, rather than one three-dimensional (3D) problem. Notice that two boundary values per a dimension are required to obtain a unique solution. On the other hand, there is no need for the boundary values for different dimensions to be of the same type.

This paper is organized as follows: in Section 2, preliminaries are discussed. In Sections 3 and 4, we will show how to exploit 1D finite element method (FEM) to solve the underlying BVP with various possible choices of boundary conditions. At this stage, we assume that no additional constraints are given and that the measurements are exact. In Section 5, we face the reality with faulty measurements and exploit linear additional constraints to enhance the accuracy of the results. The underlying BVP is treated as exact, but the measurements are corrected using a linear sensor error model. As a result, we get two systems of linear equations: an exact one for the BVP and possibly an overdetermined one for the parameters of the sensor error model. We will solve these together to yield the corrected results. Finally, two real-life examples are discussed in Sections 6 and 7 before the conclusions in Section 8.

2. Preliminaries

Let us first motivate the chosen approach by means of examples: an application suitable to our approach is ski jumping, which has been the prime motivation of this study [9]. As an inertial navigation problem, it includes the use of consumer grade sensors with knowledge of the boundary values (in this case, position at both ends of the event). See Section 7 for further details.

It turns out that especially *sports applications* tend to have properties that are well suited to the considered approach: the included actions are often periodic, in such a way that the certain short-term action (e.g., a single step) repeats several times or some longer-term action a few times (e.g., a single lap or a single jump as discussed earlier) during a certain event. In these kinds of applications, it is natural to encounter problems of the form (5) rather than (2): for example, in long jump (considering only the jump part), at time T_0 ("take off"), the velocity of the shoe is known but position is not and at time T_1 ("landing"), the position of the shoe when it hits the surface of the sand can be accurately measured but the velocity is unknown.

For a more general view, even a GPS-assisted inertial navigation system can be considered as a series of separate navigation periods with given boundary values rather than a single event with additional constraints given at certain time instances. This is an example of an IVP, which can be posed as a BVP.

2.1. Some Remarks. There are two main classes of numerical methods for BVP's. One class includes so-called shooting methods and the other class methods of weighted residuals such as FEM [8, 10]. We concentrate on the latter because of its property to minimize an error norm over the whole integration interval rather than minimizing only the local error [7]. Another tempting property is that it concerns the position directly, giving us a possibility to more easily handle various types of additional constraints we will encounter later on.

In many applications, inertial measurements are not the only source of information. In practice, however, the number or the type of these additional constraints—combined with the different kinds of a solution method—does not suggest

the use of the traditional filtering (see Section 5 for details). For these situations, we will introduce a computationally cheap and easily exploitable method to enhance the accuracy of the position and velocity estimates. The proposed method is based on sensor error modeling and is characterized by the following assumptions.

(i) Sensor errors are modeled as constant errors. While the behavior of a certain sensor error is in real life a stochastic process, one is usually able to fairly model it at least momentarily as a constant error. A suitable mathematical tool to characterize this is the *Allan variance* [11].

(ii) Considering consumer grade sensors, causes of the most significant errors are usually known (e.g., bias and scale factor error, both changing from turn-on to turn-on [2]).

(iii) In particular, the sensor noise is *not* modeled. This is a conscious modeling decision to prevent unnecessary "smoothing" of data.

(iv) Additional constraints are treated as "exact". That is, the overall error in the additional constraints is assumed to be smaller than the error caused by the simplification of the navigation equations.

When additional constraints are available, problems resembling the ones considered here have previously been resolved using the means of *fixed interval smoothing* (or "Kalman smoother", if the type of the filter is fixed) [12–15]. Considering inertial navigation, the most used methods of solving the problems are the *two-filter smoother* [16] and *Rauch-Tung-Striebel smoother* [17], used for example in [15, 18–20]. In both cases, the problem itself is posed as an IVP and the basic idea is to run the filter in the forward direction as a "predictor" in phase one and then to run the filter in the backward direction while combining these two results to yield the corrected result in phase two. Between the proposed method and the fixed interval smoothing method, the fundamental difference is that in the proposed method the dynamics model is based on the BVP formulation and in the previous approaches, on the IVP formulation with additional smoothing.

Comparing the fixed interval smoothing technique to the proposed method, in addition to the previously mentioned points, the most significant differences are as follows.

(i) As fixed interval smoothing is run in both directions, the filter needs two process models, which can be problematic [21]. As the proposed method is based on the BVP formulation, it does not make a distinction between forward and backward directions.

(ii) In the filtering approach, each additional constraint is assumed to be attached to a single time instance [2], while the proposed method does not make such a restriction (see Section 5 for details).

(iii) Fixed interval smoothing is a two-phase method requiring numerous computations per time step [2, 15], whereas the proposed approach is a single-phase method with only few computations per an unknown.

Due to the significant differences in these two approaches, we will in this context concentrate on the proposed method. Obviously, there are situations where the two methods could both be used. Comparison of the methods in such a situation is interesting, although not addressed in this paper.

Finally, recall that problem (5) can be considered as a set of three 1D problems. Thus, let us focus on the 1D case for a while. Details of the more prevalent 3D case are considered later on. Notice that because of this, the symbols will be changed a bit: $r_0, v_0 \in \mathbb{R}$ whereas $\mathbf{r}_0, \mathbf{v}_0 \in \mathbb{R}^3$ and so on. In the following treatment, it is assumed that the accelerometer samples represent an instantaneous value of the specific force. In other words, it is assumed that the sensor output is not processed in any way before the "navigation computer."

3. Solution of a BVP

The main goal of this section is to form a linear system of equations of the form $\mathbf{Ax} = \mathbf{b}$ for the position estimates $\mathbf{x}$. These are now expressed as a vector containing the position at each discrete time instant (referred to as x_i), where the vector $\mathbf{b}$ is a function of $a(t)$. In the following treatment, the total number of the samples is N, $t_i \in [T_0, T_1]$ is the value of time instant $i \in \mathbb{Z}$ $(1 \le i \le N)$, and $h_i = t_{i+1} - t_i$.

Now, let us rephrase problem (5) as a variational equation

$$\int_{T_0}^{T_1} \ddot{r}u \, \mathrm{d}t = \int_{T_0}^{T_1} au \, \mathrm{d}t \quad \forall u \in U, \tag{6}$$

where U is a Hilbert space [7]. Second derivative of r can be eliminated by integrating the left-hand side of (6) by parts, which yields

$$\int_{T_0}^{T_1} \ddot{r}u \, \mathrm{d}t = \Big/_{T_0}^{T_1} \dot{r}u - \int_{T_0}^{T_1} \dot{r}\dot{u} \, \mathrm{d}t = \int_{T_0}^{T_1} au \, \mathrm{d}t \quad \forall u \in U, \tag{7}$$

where the notation "$\Big/$" stands for substitution. By evaluating the substitution term and rearranging (7), we get

$$\int_{T_0}^{T_1} \dot{r}\dot{u} \, \mathrm{d}t = -\int_{T_0}^{T_1} au \, \mathrm{d}t + \dot{r}(T_1)u(\mathrm{T}_1) - \dot{r}(T_0)u(T_0) \quad \forall u \in U. \tag{8}$$

While (8) is otherwise in a convenient form for our purposes, it is not well suited for practical computations, because U is an infinite-dimensional space. Thus, let us approximate U with a finite-dimensional space spanned by

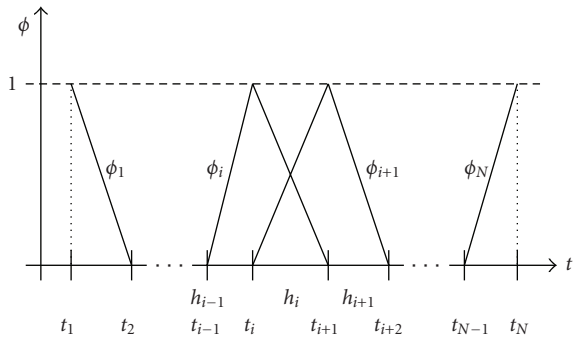

Figure 1: Lowest order basis functions ϕ_i.

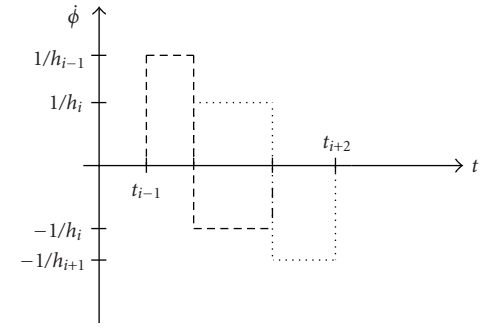

Figure 2: Time derivative of the lowest order basis functions ϕ_i (dashed line) and ϕ_{i+1} (dotted line).

piecewise affine basis functions ("affine function = linear function + a constant")

$$\phi_i = \begin{cases} 0, & t < t_{i-1}, \\ \dfrac{(t - t_{i-1})}{h_{i-1}}, & t_{i-1} \le t < t_i, \\ 1 - \dfrac{(t - t_i)}{h_i}, & t_i \le t < t_{i+1}, \\ 0, & t \ge t_{i+1}, \end{cases} \tag{9}$$

often referred to as the "hat" functions. Functions (9) for few values of i are shown in Figure 1. Also, let us use the same basis functions ϕ to discretize u and r. This choice is often referred to as the Galerkin method [7]. Note that it is also possible to choose basis functions ϕ_i different from the lowest order approximation used here, when considered necessary.

Given the basis functions ϕ_i, notice that the position r can be approximated as a piecewise affine function

$$r \approx \sum_{i=1}^{N} x_i \phi_i. \tag{10}$$

From (8), we see that the derivatives of the basis functions (9) are also needed. It holds that

$$\dot{\phi}_i = \begin{cases} 0, & t < t_{i-1}, \\ \dfrac{1}{h_{i-1}}, & t_{i-1} < t < t_i, \\ -\dfrac{1}{h_i}, & t_i < t < t_{i+1}, \\ 0, & t > t_{i+1}. \end{cases} \tag{11}$$

From Figure 2, one can verify that function $\dot{\phi}_i$ is a piecewise constant function, discontinuous at points t_{i-1}, t_i, and t_{i+1}. From (8) we see that we need to *integrate* a similar term with discontinuities at the nodes over the domain. In other words, these discontinuities do not matter, which is a well-known fact from integral theory.

In total, we are now in the position to discretize equation (8), which yields

$$\sum_{i=1}^{N} x_i \int_{T_0}^{T_1} \dot{\phi}_i \dot{\phi}_j \, \mathrm{d}t = -\int_{T_0}^{T_1} a\phi_j \, \mathrm{d}t + \dot{r}(T_1)\phi_j(T_1) - \dot{r}(T_0)\phi_j(T_0) \quad \forall j = 1, 2, \ldots, N. \tag{12}$$

Now that we have discretized the problem, we are getting closer to the equation $\mathbf{Ax} = \mathbf{b}$ stated as our goal. In fact, (12) is a system of linear equations. Thus, let us start by assembling the matrix $\mathbf{A}$. Knowing that the index i in (12) refers to a certain column and j to a certain row, the elements of $\mathbf{A}$ are given as

$$\mathbf{A}[i, j] = \int_{T_0}^{T_1} \dot{\phi}_i \dot{\phi}_j \, \mathrm{d}t \quad \forall i, j = 1, 2, \ldots, N, \tag{13}$$

which are easy to compute by substitution of (11) into (13). In practice, $\mathbf{A}$ is going to have only few nonzero elements all of them at the diagonal, subdiagonal, and superdiagonal (assuming the "obvious" indexing). For equally spaced nodes with step size h, for example, we have elements $2/h$ at the diagonal and $-1/h$ at the sub-and superdiagonal.

Our next task is to compute the vector $\mathbf{b}$. As seen from (12), the task is to integrate term $a\phi_j$ over the interval $[T_0, T_1]$. To do this, we fit a piecewise affine function to $a(t)$ between every node as seen in Figure 3 with dashed line, which yields

$$\int_{t_{j-1}}^{t_{j+1}} a\phi_j \, \mathrm{d}t = \frac{h_{j-1}}{6}\left[a\left(t_{j-1}\right) + 2a\left(t_j\right)\right] + \frac{h_j}{6}\left[2a\left(t_j\right) + a\left(t_{j+1}\right)\right] \tag{14}$$

for every $j \in [2, N-1]$. For nodes $j = 1$ and $j = N$, we get

$$\int_{t_1}^{t_2} a\phi_j \, \mathrm{d}t = \frac{h_1}{6}[2a(t_0) + a(t_1)], \qquad \int_{t_{N-1}}^{t_N} a\phi_j \, \mathrm{d}t = \frac{h_{N-1}}{6}[a(t_{N-1}) + 2a(t_N)], \tag{15}$$

respectively.

Finally, let us consider how to apply the different types of boundary values into (12). At first, notice that terms $\phi_j(T_1)$ and $\phi_j(T_0)$ seen in (12) will be nonzero (evaluating to value one) only for the values $j = 1$ and $j = N$, respectively. If $\dot{r}(T_0) = v_0$ or $\dot{r}(T_1) = v_1$ of (5) is given, the respective boundary condition is called a *Neumann* boundary condition [7]. These can be applied directly by adding the given values into $\mathbf{b}$.

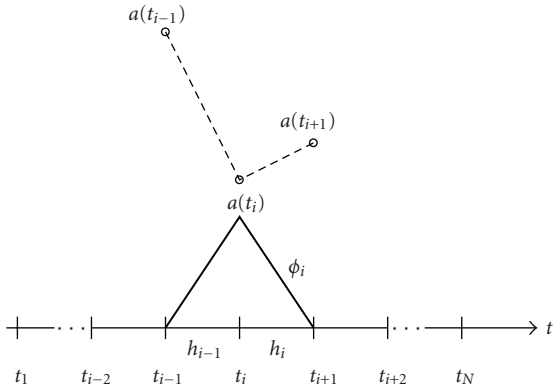

Figure 3: Basis function ϕ_i along with the corresponding acceleration values.

The other possibility for the boundary values is to have $r(T_0) = r_0$ or $r(T_1) = r_1$ fixed thus having a *Dirichlet* boundary condition [7]. As the value at the corresponding boundary is already known, it does not need to be solved. Thus, we can move all terms depending on it to $\mathbf{b}$ reducing the number of unknown terms by one. For the reduced system, the corresponding term of the last two terms of (12) will be zero, as mentioned above. Recall that it is also possible to have a Dirichlet condition on one boundary and a Neumann condition on the other boundary.

We have now means to assembly an equation of the form

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{16}$$

for the (1D) position of the object. Depending on the type of the applied boundary conditions, the number of unknowns n is equal to $N-1$ or $N-2$. Matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ is known to be symmetric and positive definite [22]. With the chosen basis functions, $\mathbf{A}$ is also a tridiagonal matrix. In practice, these properties guarantee that (16) can be solved with linear time complexity [8]. In other words, when doubling the amount of unknowns N, the time needed to solve the system is also doubled (approximately), which is certainly not true for a general system of linear equations.

From the position data $\mathbf{x}$, it is now a straightforward task to compute the velocity using a suitable numerical differentiation formula. Since the position data is relatively smooth due to the "double integration" process, we have not experienced any problems in computing the derivative with an adequate accuracy.

4. Generalization to 3D

In this section, we will generalize the method described in the previous section to the 3D case. For this, we introduce a coordinate transformation matrix $\mathbf{C}(t)$ used to transform the specific force measurements $\mathbf{f}^{\mathrm{b}}(t)$ into the accelerations $\mathbf{a}^{\mathrm{n}}(t)$ represented in a suitable *navigation* frame as follows:

$$\mathbf{a}^{\mathrm{n}}(t) = \mathbf{C}(t)\mathbf{a}^{\mathrm{b}}(t). \tag{17}$$

An element-wise representation of $\mathbf{C}(t)$ is

$$\mathbf{C}(t) = \begin{bmatrix} c_{11}(t) & c_{12}(t) & c_{13}(t) \\ c_{21}(t) & c_{22}(t) & c_{23}(t) \\ c_{32}(t) & c_{32}(t) & c_{33}(t) \end{bmatrix}. \tag{18}$$

In this paper, we will assume that an estimate of $\mathbf{C}(t)$ for each time instant is available. Note that $\mathbf{C}(t)$ depends only on the attitude, which is independent of the position and velocity, when the assumptions considered in the first section are valid [2]. Furthermore, let vector $\mathbf{g}^n$ be the acceleration due to the gravity represented in the navigation frame. With these notions, the specific force measurements f made by the accelerometers can be "converted" into accelerations as

$$\begin{aligned} a_1^{\mathrm{n}} &= c_{11}f_1^{\mathrm{b}} + c_{12}f_2^{\mathrm{b}} + c_{13}f_3^{\mathrm{b}} + g_1^{\mathrm{n}}, \\ a_2^{\mathrm{n}} &= c_{21}f_1^{\mathrm{b}} + c_{22}f_2^{\mathrm{b}} + c_{23}f_3^{\mathrm{b}} + g_2^{\mathrm{n}}, \\ a_3^{\mathrm{n}} &= c_{31}f_1^{\mathrm{b}} + c_{32}f_2^{\mathrm{b}} + c_{33}f_3^{\mathrm{b}} + g_3^{\mathrm{n}}, \end{aligned} \tag{19}$$

where the time dependencies have not been explicitly stated.

Now, following from (19) and the right-hand side of (12), we have an equation

$$\begin{aligned} \mathbf{b}_i[j] &= -\int_{T_0}^{T_1} a_i^{\mathrm{n}}\phi_j \,\mathrm{d}t \\ &= -\int_{T_0}^{T_1} \left[c_{i1}f_1^{\mathrm{b}} + c_{i2}f_2^{\mathrm{b}} + c_{i3}f_3^{\mathrm{b}} + g_i^{\mathrm{n}}\right]\phi_j \,\mathrm{d}t \end{aligned} \tag{20}$$

for the j th component of vector $\mathbf{b}_i \ \forall\, i = [1,2,3]$. Thus, the linear equation for the 3D position is

$$\begin{bmatrix} \mathbf{A}_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_2 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{A}_3 \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \end{bmatrix} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \mathbf{b}_3 \end{bmatrix}. \tag{21}$$

As discussed in the previous section, matrix $\mathbf{A}_i$ is a tridiagonal matrix. Then, also the system matrix of (21) is not only tridiagonal, but a block diagonal matrix. For instance, if boundary conditions are fixed in the same way for all dimensions, it follows that $\mathbf{A}_1 = \mathbf{A}_2 = \mathbf{A}_3$ holds.

With the equations derived in this section, it is possible to solve the 3D BVP using the presented FEM method. The solution of this more general problem can also be found with linear time complexity, as in the 1D case.

5. Using a Number of Additional Constraints to Model Sensor Errors

So far we have constructed a method to compute position and velocity data with certain boundary conditions. We will now propose a way to exploit a number of additional constraints concerning the velocity or the position information to estimate sensor errors. For this, let us now precisely define the term "additional constraint": *an additional constraint is a linear equation that bounds the position or the velocity of the object at an arbitrary number of time instances* $t_i \in [T_0, T_1]$.

At simplest, the previous definition could simply mean that $v_1(t_i)$ is fixed. On the other hand, a less intuitive equation $\sum_{i=1}^{N}[r_1(t_i) + r_2(t_i)] = z_1$ is also a valid constraint. This way, we can for example exploit constraints like $x_i = x_j$ for any i and j without saying anything about the absolute position at these points. It could be hard to exploit these kind of constraints properly in traditional filtering problems.

In this section, we will use constraints systematically to enhance accuracy. To make the concepts presented in this section clear, let us first present the equations for the 1D case.

5.1. Treating Additional Constraints. Let us first assume that $\mathbf{A} \in \mathbb{R}^{N\times N}$ holds. That is, given a Dirichlet boundary condition, we add an auxiliary equation to the system instead of removing the corresponding boundary value from the system. This is a necessary procedure when exploiting additional constraints.

In general, all linear constraints can be represented in the form

$$\mathbf{D}\mathbf{x} = \mathbf{e}, \tag{22}$$

where $\mathbf{D} \in \mathbb{R}^{p\times N}$ and $\mathbf{e} \in \mathbb{R}^{p}$ hold, where p is the number of the constraint equations. For a practical example closely related to the example 1, consider that (5) is solved with Dirichlet boundary values. If the Neumann boundary values are also known and the samples are equally spaced, one can construct (22) as

$$\mathbf{D} = \begin{bmatrix} -\frac{3}{2h} & \frac{2}{h} & -\frac{1}{2h} & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & \frac{1}{2h} & -\frac{2}{h} & \frac{3}{2h} \end{bmatrix} \tag{23}$$

$$\mathbf{e} = \begin{bmatrix} v_0 \\ v_1 \end{bmatrix}. \tag{24}$$

Matrix $\mathbf{D}$ is formed using a three-point differentiation method based on polynomial interpolation [10] to determine Neumann boundary values from the displacement data.

These constraints could be exploited just by computing the linear least squares problem constructed by adding these additional equations to (16). Usually, however, it is favorable to use the additional constraints to *model sensor errors*, at least when one has any knowledge of the types of the errors included in the measurements. For more information on the assumptions related to the sensor error modeling, recall Section 2.1. In this text, we will present a linear sensor error model.

5.2. Sensor Error Model. In typical situations, bias offset (i.e., the sensor shows nonzero output when no forces are acting upon it) and scaling factor are the two terms which have to beknown very accurately in order to have any realistic position or velocity estimate as a solution. Because of the run-to-run variations of these errors, they cannot be assumed to be constant between two separate events. Thus, they should be treated as unknowns. Other typical errors are caused by misalignment of the axes, changing temperature and drifting bias.

As a practical example used in the example 1 later in Section 6, let us model the sensor errors as follows:

$$\hat{f}(t) = sf(t) + b, \tag{25}$$

where $\hat{f}(t)$ is the corrected specific force measurement, s is some constant scaling factor, and b is some constant bias term correcting the erroneous measurement. By replacing the measured $f(t)$ treated in the previous section with the corrected acceleration $\hat{f}(t)$, it is easy to form an equation of the form

$$\mathbf{b} = \mathbf{F}\mathbf{l} \tag{26}$$

for vector $\mathbf{b}$ seen in (16). Matrix $\mathbf{F}$ is in this case an $N \times 2$ matrix, whose elements are formed by computing the integrals $-\int_{T_0}^{T_1} a\phi_j$ and $-\int_{T_0}^{T_1} \phi_j$ from (12). Vector $\mathbf{l}$ is simply $[\, s \; b \,]^T$.

Let us now replace the right-hand side of (16) with

$$\mathbf{b}_0 + \mathbf{b} = \mathbf{b}_0 + \mathbf{F}\mathbf{l}, \tag{27}$$

where $\mathbf{b}_0$ depends only on the given boundary conditions (which are treated as exact) and corresponding boundary values. This distinction is necessary, since one should not modify the given boundary values by applying error modeling on them.

In general, a problem of linear constraints used to model linear sensor errors can be stated as

$$\begin{cases} \mathbf{A}\mathbf{x} = \mathbf{b}_0 + \mathbf{F}\mathbf{l}, \\ \mathbf{D}\mathbf{x} = \mathbf{e}, \end{cases} \tag{28}$$

where $\mathbf{F} \in \mathbb{R}^{N\times q}$, $\mathbf{l} \in \mathbb{R}^{q}$ hold, and q is the number of modeled sensor error terms.

Now, since $\mathbf{A}$ is known to be invertible, one gets an equation

$$\mathbf{D}\mathbf{A}^{-1}\mathbf{F}\mathbf{l} = \mathbf{e} - \mathbf{D}\mathbf{A}^{-1}\mathbf{b}_0 \tag{29}$$

for $\mathbf{l}$. Note that the matrix $\mathbf{D}\mathbf{A}^{-1}\mathbf{F}$ has dimension $p \times q$, that is, it is typically a very small matrix compared to $\mathbf{A}$. With $\mathbf{l}$ known, it is easy to solve for $\mathbf{x}$.

In the case where $p > q$ (more constraints than model parameters), one can also take reliability of different measurements into account by solving a *weighted least squares problem* (WLS) [22]. In general, (29) has a unique solution $\mathbf{l}$ only when $p = q$ holds and the row rank of $\mathbf{D}$ is full. Typically, making sure that $p \geq q$ and that the row rank of $\mathbf{D}$ is at least q, (29) has either a unique or a least squares solution for $\mathbf{l}$. In each case, the upper equation of (28) is treated as an exact equation.

5.3. Additional Constraints in 3D. Let us now consider the use of additional constraints in a 3D case. Using the

notions form the previous section and (22) as an example, constraints can be represented as

$$\mathbf{D}\begin{bmatrix}\mathbf{x}_1\\ \mathbf{x}_2\\ \mathbf{x}_3\end{bmatrix} = \mathbf{e}. \tag{30}$$

In addition to the sensor errors discussed in the 1D example, it is now also possible to model errors like attitude errors, unknown value of the acceleration due to the gravity, and cross-correlation of different sensors. A particularly useful method is to treat the acceleration due to the gravity as an unknown three-dimensional vector, which is then subtracted from the specific force measurements given in the global coordinates. This reduces the systems sensitivity to initial attitude errors.

As an example, let us now derive the equations for sensor error of (25) for each axis in addition to the unknown acceleration due to the gravity discussed above. Thus, we have a total of 9 unknown model parameters. As one could expect, we must take the effects of the coordinate transformation matrix into account when deriving the necessary equations. By plugging (25) into (21), we have

$$\begin{aligned}\mathbf{b}_i[j] = &-\int_{T_0}^{T_1}\Big[c_{i1}\left(s_1 f_1^{\mathrm{b}} + b_1\right) + c_{i2}\left(s_2 f_2^{\mathrm{b}} + b_2\right)\\ &- c_{i3}\left(s_3 f_3^{\mathrm{b}} + b_3\right) + g_i^{\mathrm{n}}\Big]\phi_j\,\mathrm{d}t\\ = &-s_1\int_{T_0}^{T_1} c_{i1} f_1^{\mathrm{b}}\phi_j\,\mathrm{d}t - b_1\int_{T_0}^{T_1} c_{i1}\phi_j\,\mathrm{d}t\\ &- s_2\int_{T_0}^{T_1} c_{i2} f_2^{\mathrm{b}}\phi_j\,\mathrm{d}t - b_2\int_{T_0}^{T_1} c_{i2}\phi_j\,\mathrm{d}t\\ &- s_3\int_{T_0}^{T_1} c_{i3} f_3^{\mathrm{b}}\phi_j\,\mathrm{d}t - b_3\int_{T_0}^{T_1} c_{i3}\phi_j\,\mathrm{d}t\\ &- g_i^{\mathrm{n}}\int_{T_0}^{T_1}\phi_j\,\mathrm{d}t.\end{aligned} \tag{31}$$

The several integrals in (31) are scalars for each $j \in [1, 2, \ldots N]$, easily evaluated with (14) and (15), since no unknown terms appear inside the integrals. Let us now gather the results into vectors

$$\mathbf{p}_{klmn}[j] = -\int_{T_0}^{T_1} c_{kl}^{m}\left(f_l^{\mathrm{b}}\right)^{n}\phi_j\,\mathrm{d}t \tag{32}$$

for all $j \in [1, 2, \ldots, N]$. With these notions, we get, for example

$$\begin{aligned}\mathbf{p}_{kl11}[j] &= -\int_{T_0}^{T_1} c_{kl} f_l^{\mathrm{b}}\phi_j\,\mathrm{d}t,\\ \mathbf{p}_{kl1}[j] &= -\int_{T_0}^{T_1} c_{kl}\phi_j\,\mathrm{d}t,\\ \mathbf{p}[j] &= -\int_{T_0}^{T_1}\phi_j\,\mathrm{d}t,\end{aligned} \tag{33}$$

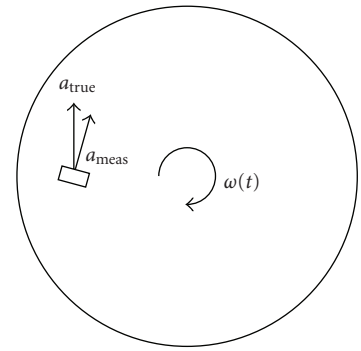

Figure 4: Measurement setup of example 1.

where only the relevant indices are shown. Thus, we get size

$$\mathbf{F} = \begin{bmatrix}\mathbf{p}_{1111} & \mathbf{p}_{111} & \mathbf{p}_{1211} & \mathbf{p}_{121} & \mathbf{p}_{1311} & \mathbf{p}_{131} & \mathbf{p} & \mathbf{0} & \mathbf{0}\\ \mathbf{p}_{2111} & \mathbf{p}_{211} & \mathbf{p}_{2211} & \mathbf{p}_{221} & \mathbf{p}_{2311} & \mathbf{p}_{231} & \mathbf{0} & \mathbf{p} & \mathbf{0}\\ \mathbf{p}_{3111} & \mathbf{p}_{311} & \mathbf{p}_{3211} & \mathbf{p}_{321} & \mathbf{p}_{3311} & \mathbf{p}_{331} & \mathbf{0} & \mathbf{0} & \mathbf{p}\end{bmatrix}, \tag{34}$$

$$\mathbf{l} = \begin{bmatrix}s_1 & b_1 & s_2 & b_2 & s_3 & b_3 & g_1^{\mathrm{n}} & g_2^{\mathrm{n}} & g_3^{\mathrm{n}}\end{bmatrix}^{\mathrm{T}}. \tag{35}$$

For the nine unknown model parameters we need at least nine constraints. This can be covered, for example, with the knowledge of the velocity of the object at three different points. In total, we have equations identical to (28), with dimension N replaced by $3N$.

6. Example 1

To demonstrate the use of the proposed method, an example using readily available consumer grade accelerometers [23] was created. An accelerometer was mounted on a horizontally rotating rate table, whose angular velocity (rotation rate) can be controlled. The accelerometer was mounted in such a way that its measurement direction was *approximately* the same as the direction of the tangential acceleration caused by angular acceleration of the rate table, as seen in Figure 4.

The aim of this example was not to get position and velocity as accurately as possible, but to compare different solution methods in a situation where one needs to get reasonable position and velocity estimates regardless of the fact that the measurement contains significant errors. The angular velocity of the rate table was set to follow function illustrated in the Figure 5 a certain number of times. With this kind of a setup, the true acceleration, velocity, and displacement of the mounted sensor were known up to the accuracy of the motor rotating the rate table. The errors caused by the rate table were observed using the angular rate output of the motor and found to be negligible compared to other error sources.

In first test, the rate table was set to repeat exactly the function represented in Figure 5 ten times, leading to seven full revolutions (or 10.4 meters) in $T = 20$ seconds. In the other test, the function of the same form was scaled in such a way that 50 repeats lead to total of 62 full revolutions (or 92.3 meters) in $T = 100$ seconds. The sampling frequency of

Table 1: Comparison of different methods computing object's velocity and position. "I" stands for time-stepping, "II" for FEM, and "III" for FEM with sensor error modeling.

	Test 1			Test 2		
Vel. error [m/s]	t_1	t_2	t_3	t_1	t_2	t_3
I	0.07	0.10	0.12	33.24	65.47	97.17
II	0.01	0.04	0.06	−0.67	0.50	1.15
III	0.04	0.05	0.04	0.33	0.54	0.44
Pos. error [m]	t_1	t_2	t_3	t_1	t_2	t_3
I	0.10	0.49	1.00	510.96	1992.99	4434.70
II	−0.31	−0.29	−0.15	−42.34	−43.42	−16.59
III	−0.10	−0.07	−0.05	−6.78	−4.72	−0.61

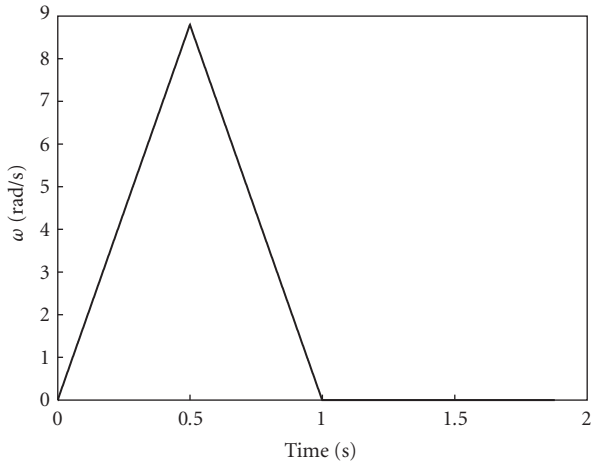

Figure 5: Angular velocity of the rate table.

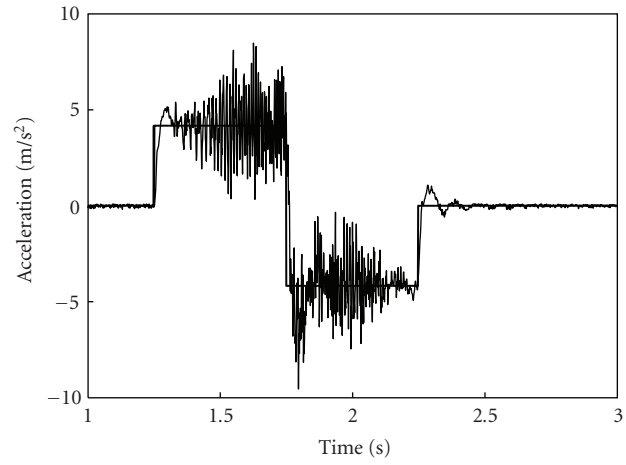

Figure 6: Ideal (thick line) and measured acceleration (thin line).

the sensor was set to 1000 Hz and an accurate (16 bit) analog to digital converter was used. Accelerations were measured using a consumer grade 12 g accelerometer [23].

The raw accelerometer data was mapped to accelerations with the scale factor given by the manufacturer. From this acceleration data, position and velocity were computed by a number of different methods:

(i) "Traditional" IVP (double integration with trapezoid rule),

(ii) FEM with Dirichlet boundary conditions (BCs),

(iii) FEM with Dirichlet BCs combined with additional Neumann BCs to supply the error model (25).

In methods I and II, the bias error of the sensor was estimated by averaging the output while the sensor was at rest. This was done in order to make method I (and to some extent, method II) comparable to the method III by reduction of the large bias error. This is rarely possible in general and only method III can be used to reliably detect any remaining bias (example 2 in Section 7 is an example of this case). As a reference, the ideal (ideal driving motor) and the measured accelerations (accelerometer bias removed) of the first spin are plotted in Figure 6.

Table 1 shows the main results of the tests. In each test, the computed results were compared to the known value in three separate points by computing the difference between the computed and the real value. Point t_1 was located at $T/3$, t_2 at $T/2$, and t_3 at $2T/3$.

As seen in Table 1, method I seems to increase the error with increasing time, as expected. Method II on the contrary, thanks to the basic property of the variational technique, does not increase the error but distributes it over the whole time period. The difference between these two methods is clearly seen in the velocity and position errors of the longer test (test 2), where method II gives much better estimates than method I. In each test, method III clearly outperforms methods I and II, which is expected due to the provided two additional constraints.

Figures 7 and 8 demonstrate the differences between velocity and position estimates given by methods I and II during test 1. Estimates given by method III coincide with the reference plots. Figure 7 shows only the last spin of the test 1 for better view of the differences.

7. Example 2

This example considers the computation of the velocity and position of a ski jumper during a single jump. As compared to the previous example, this is a more realistic and general inertial navigation problem with six degrees of freedom.

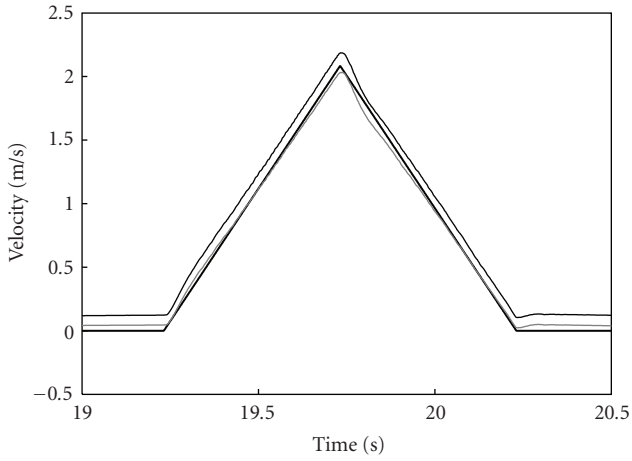

Figure 7: Velocity estimates given by methods I (thin black line) and II (thin gray line) compared to the ideal velocity (thick line).

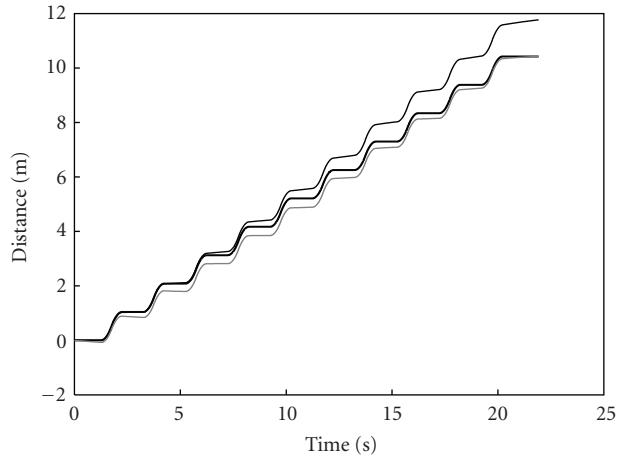

Figure 8: Position estimates given by methods I (thin black line) and II (thin gray line) compared to the ideal position (thick solid line).

Computation of the attitude of the object is based on the data given by three standard consumer grade gyroscopes [24]. The position and velocity were then measured with three standard consumer grade accelerometers [23] and computed with the proposed method with a sensor error model similar to the one presented in Section 5.3. The needed additional constraints contained information about

(i) the location of the jumper at five points evenly spread on the inrun hill (in Figure 9, the part of thick black line with negative x- and positive y-coordinates)

(ii) the trajectory of the jumper after the landing, which should coincide with the linearization of the landing hill (in Figure 9, the part of thick black line with x-coordinates greater than 100 m).

In Figure 9, two-dimensional trajectories of two jumpers are plotted along with the known profile of the hill. At first, the trajectories follow the profile of the hill until the jumpers

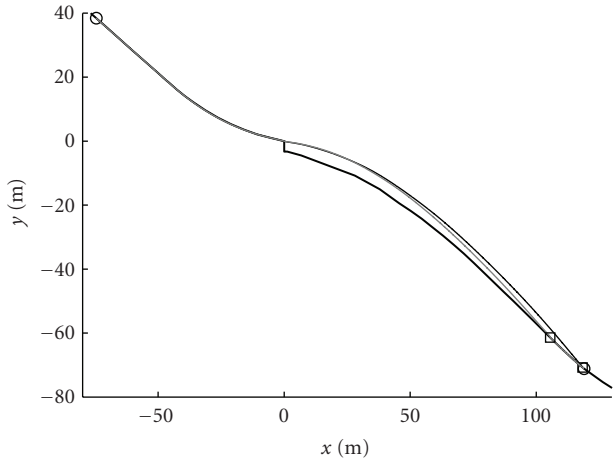

Figure 9: Computed two-dimensional trajectories of two independent events (thin gray and thin black lines) along with the known profile of the hill (thick black line).

take off and eventually land at some point of the landing hill. Drawn circles at both ends of the trajectories demonstrate the start and the end of the navigation period and the given Dirichlet boundary conditions. Drawn squares represent the landing points, in this case quite different for the two events, agreeing well with the recorded jump lengths. Notice that the trajectories are computed using two distinct measurement systems, each containing unique error sources.

Figure 10 shows that the vertical component of the velocity of the same two jumps is plotted as a function of horizontal displacement. Notice how the absolute values of the vertical velocity substantially differ while the small changes in the velocity are practically identical. One might first consider the small changes as typical stochastic errors caused by the inertial navigation system, especially when dealing with low performance sensors.

Given that two independent measurement systems show the same variations in the velocity at the same locations, it is evident that there is actually only a negligible amount of stochastic errors present. Instead, the small variations are caused by deterministic sources, namely, in this particular application the uneven inrun hill.

Unfortunately, the estimates cannot be compared with a reference trajectory, because such data are not available. Thus, we cannot give the exact amount of error present in the position and velocity estimates. We do however claim that the achieved accuracy is something one does not typically expect from consumer grade inertial sensors.

8. Conclusion

The work was motivated by applications, where it is natural to encounter BVPs instead of IVPs. In many cases, it is also possible to formulate an IVP as a BVP, given that the results are not required in real time.

Finite element method is utilized to solve inertial navigation problems formulated as BVPs. As a result, we get a linear system of equations for the position estimates, whose

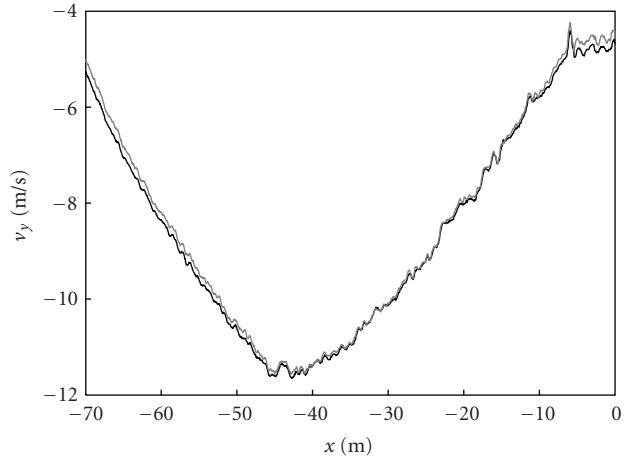

FIGURE 10: Vertical velocity (v_y) as a function of horizontal displacement (x) of the two independent measurements.

solution can be found with linear time complexity. It is demonstrated that solving a BVP rather than an "equivalent" IVP gives more accurate results.

For further accuracy enhancements, an efficient way of combining inertial measurements with possible additional constraints is created. This gives us a possibility to model constant sensor errors, known to limit the achievable accuracy of the system. While the error model significantly enhances the accuracy of the system, it is kept computationally simple and easily adoptable.

In practice, the accuracy improvements allow us to exploit inertial sensors of certain performance level in more challenging applications. For this, it is necessary to see that the concept of inertial navigation does not invariably imply an IVP, but a BVP as well. Then, the use of FEM will provide an efficient way to compute position and velocity estimates not prone to the accumulation of errors.

In larger scale, the current paper serves as an introduction to the idea of formulating inertial navigation problems as BVPs. As a consequence, further studies are needed to address problems to which the presented tools do not provide an obvious solution. These include, for example, stochastic errors, reliability of the possible additional constraints (as compared to the accuracy of the IMU), and coupling of position and attitude errors.

References

[1] I. Y. Bar-Itzhack, "Navigation computation in terrestrial strapdown inertial navigation systems," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 13, no. 6, pp. 679–689, 1977.

[2] D. H. Titterton and J. L. Weston, *Strapdown Inertial Navigation Technology*, Institution of Engineering and Technology, London, UK, 2nd edition, 2004.

[3] A. B. Chatfield, *Fundamentals of High Accuracy Inertial Navigation*, American Institute of Aeronautics and Astronautics, Reston, Va, USA, 1997.

[4] M. S. Grewal, L. R. Weill, and A. P. Andrews, *Global Positioning Systems, Inertial Navigation, and Integration*, John Wiley & Sons, New York, NY, USA, 2001.

[5] J. A. Farrell and M. Barth, *The Global Positioning System & Inertial Navigation*, McGraw–Hill, New York, NY, USA, 1999.

[6] L. F. Shampine, I. Gladwell, and S. Thompson, *Solving ODEs with Matlab*, Cambridge University Press, Cambridge, UK, 2003.

[7] S. Larsson and V. Thome, *Partial Differential Equations with Numerical Methods*, Springer, New York, NY, USA, 2003.

[8] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes in C (The Art of Scientific Computing)*, Cambridge University Press, Cambridge, UK, 2nd edition, 1992.

[9] M. Virmavirta, T. Nieminen, T. Tarhasaari, and L. Kettunen, "High precision inertial measurement for tracking the trajectories of ski jumpers "Smart Boot Project"," in *Proceedings of the 13th Congress European College of Sport Science*, p. 572, Estoril, Portugal, July 2008.

[10] D. Kincaid and W. Cheney, *Numerical Analysis: Mathematics of Scientific Computing*, BROOKS/COLE, Florence, Ky, USA, 3rd edition, 2002.

[11] N. El-Sheimy, H. Haiying, and N. Xiaoji, "Analysis and modeling of inertial sensors using allan variance," *EEE Transactions on Instrumentation and Measurement*, vol. 57, no. 1, pp. 140–149, 2008.

[12] A. Gelb, *Applied Optimal Estimation*, MIT Press, Cambridge, Mass, USA, 1974.

[13] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*, Chapman & Hall / CRC, London, UK, 2004.

[14] J. L. Crassidis and J. L. Junkins, *Optimal Estimation of Dynamic Systems*, Chapman & Hall / CRC, London, UK, 2004.

[15] P. D. Groves, *Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems*, Artech House, Norwood, Mass, USA, 2008.

[16] D. Fraser and J. Potter, "The optimum linear smoother as combination of two optimum linear filters," *IEEE Transactions on Automatic Control*, vol. 14, no. 4, pp. 387–390, 1969.

[17] H. E. Rauch, F. Tung, and C. T. Striebel, "Maximum likelihood estimates of linear dynamic systems," *AIAA Journal*, vol. 3, no. 8, pp. 1445–1450, 1965.

[18] K. Gade, "Navlab, a generic simulation and postprocessing tool for navigation," *European Journal of Navigation*, vol. 2, no. 4, pp. 21–59, 2004.

[19] Y. Yang, Z. Jin, W. Tian, and F. Qian, "Application of fixed interval smoothing to gps/dr integrated navigation system," in *Proceedings of the IEEE Intelligent Transportation Systems*, vol. 2, pp. 1027–1031, October 2003.

[20] A. B. Willumsen and ∅. Hegrenæs, "The joys of smoothing," in *Proceedings of the IEEE Bremen: Balancing Technology with Future Needs (OCEANS '09)*, pp. 1–7, Bremen, Germany, May 2009.

[21] M. Klaas, M. Briers, N. de Freitas, A. Doucet, S. Maskell, and D. Lang, "Fast particle smoothing: if i had a million particles," in *Proceedings of the 23rd International Conference on Machine Learning (ICML '06)*, vol. 148, pp. 481–488, Pittsburgh, Pa, USA, 2006.

[22] J. W. Demmel, *Applied Numerical Linear Algebra*, SIAM, Philadelphia, Pa, USA, 1st edition, 1997.

[23] VTI. *VTI SCA620-CHCV1A Datasheet*, 2006. Revision 2/2, Checked on 22.12.2009.

[24] Analog Devices. *AD ADXRS300ABG Datasheet*, 2004. Revision B, Checked on 22.12.2009.