2012 International Conference on Indoor Positioning and Indoor Navigation, 13-15th November 2012

Smoothing for ZUPT-aided INSs

David Simón Colomar, John-Olof Nilsson, and Peter Händel
Signal Processing Lab, ACCESS Linnaeus Centre
KTH Royal Institute of Technology
Stockholm, Sweden

***Abstract*—Due to the recursive and integrative nature of zero-velocity-update-aided (ZUPT-aided) inertial navigation systems (INSs), the error covariance increases throughout each ZUPT-less period followed by a drastic decrease and large state estimate corrections as soon as ZUPTs are applied. For dead-reckoning with foot-mounted inertial sensors, this gives undesirable discontinuities in the estimated trajectory at the end of each step. However, for many applications, some degree of lag can be tolerated and the information provided by the ZUPTs at the end of a step can be made available throughout the step, eliminating the discontinuities. For this purpose, we propose a smoothing algorithm for ZUPT-aided INSs. For near real-time applications, smoothing is applied to the data in a step-wise manner requiring a suggested varying-lag segmentation rule. For complete off-line processing, full data set smoothing is examined. Finally, the consequences and impact of smoothing are analyzed and quantified based on real-data.**

I. Introduction

Pedestrian dead-reckoning systems constructed around foot-mounted inertial measurement units (IMUs) have shown remarkable tracking performance [1]–[6]. The potential applications range from blue-force tracking, ambient living/smart offices, and ambulatory gait analysis. These navigation systems are commonly implemented as zero-velocity-update-aided (ZUPT-aided) inertial navigation systems (INSs). Owing to their integrative and recursive nature, the error covariance increases throughout each step and "collapses" at the end of the step when large corrections to the state estimates are applied. These large corrections complicate motion analysis and can be distracting for visualization. The situation is illustrated in Fig. 1 where multiple tracked steps are plotted beside each other. See also Fig. 5-6 in the end of the article for illustration of the behavior of the covariances. Unfortunately, for applications with tight real-time constraints, this behavior is unavoidable, since every estimate corresponds to the best estimate including all information up until that time instant. However, for many applications, some degree of lag (non-causality) can be tolerated and the information provided by the ZUPTs at the end of a step, causing the discontinuities, can be made available throughout the step. However, incorporating this information requires some non-causal filtering. Consequently, to eliminate the discontinuities and unsymmetrical covariance over the steps, the implementation of a smoothing filter for a ZUPT-aided INS is considered. To our knowledge, no treatment of smoothing for such systems has previously been presented, even though extensive literature exists on the general subject.

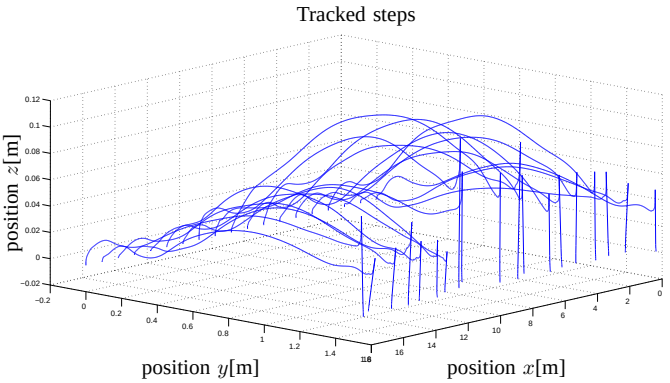

Fig. 1: Steps from a straight-line trajectory as tracked by a ZUPT-aided INS. Large corrections causing apparent discontinuities can be seen at the end of each step. Note the difference in scale between the xy-plane and the z-axis.

The remainder of the article is structured as follows. In Section II the underlaying ZUPT-aided INS is reviewed. In Section III the smoothing problem is introduced and the general smoothing formula is given. We argue that the costumary ZUPT-aided INS filtering cannot be mapped to the smoothing formula and revert to an open-loop implementation of the same. Also since the measurements (the ZUPTs) are irregularly spaced and appear in clusters, a varying-lag smoothing rule is necessary and therefore introduced. By combining all of the different considered aspects, the proposed smoothing algorithm for a ZUPT-aided INS is given. Finally, in section IV, the impact of the smoothing throughout the steps is quantified on real data.

Reproducible research: A Matlab implementation of the suggested smoothing algorithm is available at www.openshoe.org.

II. ZUPT-aided INS

Conceptually, the ZUPT-aided INS consists of an inertial measurement unit (IMU), giving specific force and angular rate measurements, and a Kalman type of filter, giving navigation state estimates. In the following subsections, the customary filtering implementation is reviewed.

A. Inertial navigation

A foot-mounted IMU is most likely of strap-down type and the IMU measurements need to be transformed from the sensor

978-1-4673-1954-6/12/$31.00

frame to the *fixed* navigation frame. Therefore, in the first place the measurements taken by the gyroscope are integrated to find the relative orientation from one frame to another. The relative orientation is represented with quaternions $\mathbf{q}_n$ and updated with

$$\mathbf{q}_n = \left[\cos\left(\frac{\|\omega_n\|T_s}{2}\right)\mathbf{I}_4 + \frac{2}{\|\omega_n\|T_s}\sin\left(\frac{\|\omega_n\|T_s}{2}\right)\mathbf{\Omega}_n\right]\mathbf{q}_{n-1} \tag{1}$$

where $\omega_n = [\omega_n^x, \omega_n^y, \omega_n^z]^T$, T_s is the sampling period of the system, n is a time index, ω_n^i is the angular rate measurement around the i axis, and

$$\mathbf{\Omega}_n = \frac{T_s}{2}\begin{bmatrix} 0 & \omega_n^z & -\omega_n^y & \omega_n^x \\ -\omega_n^z & 0 & \omega_n^x & \omega_n^y \\ \omega_n^y & -\omega_n^x & 0 & \omega_n^z \\ -\omega_n^x & -\omega_n^y & -\omega_n^z & 0 \end{bmatrix} \tag{2}$$

is the quaternion update matrix. However, the orientation might be equivalently represented with the rotation matrix $\mathbf{R}_n \Leftrightarrow \mathbf{q}_n$ or the Euler angles $\boldsymbol{\theta}_n \Leftrightarrow \mathbf{q}_n$. For clarity, we will use the different representation interchangeably.

Once the current orientation is known, the specific force measured by the accelerometers $\mathbf{f}_n$ can be expressed in the navigation frame. This allows a compensation to be made for the gravitational acceleration $\mathbf{g} = [0, 0, 9.81]^T$

$$\mathbf{a}_n = \mathbf{R}_n\mathbf{f}^b - \mathbf{g} \tag{3}$$

which yields the acceleration $\mathbf{a}_n$ in the navigation coordinate frame.

Finally, the inertial acceleration $\mathbf{a}_n$ is integrated to get the position $\mathbf{p}_n$ and velocity $\mathbf{v}_n$. Since the frequency is high and the variables discrete, the acceleration $\mathbf{a}_n$ can be considered constant between two time samples and the basic equations of motion are applied as mechanization equations

$$\mathbf{p}_n = \mathbf{p}_{n-1} + \mathbf{v}_{n-1}T_s + \frac{1}{2}\,\mathbf{a}_n T_s^2 \tag{4}$$

$$\mathbf{v}_n = \mathbf{v}_{n-1} + \mathbf{a}_n T_s. \tag{5}$$

Concatenating the position, velocity, and orientation representation into a navigation state vector $\mathbf{x}_n = (\mathbf{p}_n, \mathbf{v}_n, \theta_n)$ allow us to describe equations (1)-(5) as a state space system

$$\mathbf{x}_n = f_{\text{mech}}(\mathbf{x}_{n-1}, \mathbf{f}_n, \boldsymbol{\omega}_n). \tag{6}$$

Together with the IMU, this state space system makes up the INS.

B. ZUPT-aiding

Unfortunately, the errors of the state estimates as propagated by the INS increase rapidly with time. Therefore, additional information is required for correcting the estimates. In the current scenario, pseudo-measurements in the form of ZUPTs are used. The idea underlying the ZUPTs is to detect the state when the shoe is stationary and, hence, its velocity is supposedly zero. The system is considered stationary if

$$T(\{\boldsymbol{\omega}^i, \mathbf{f}^i\}_{W_n}) < \gamma$$

where $T(\cdot)$ is some test statistics, $\{\boldsymbol{\omega}^i, \mathbf{f}^i\}_{W_n}$ is the inertial measurements over some time window W_n, and γ is some threshold. See [7] for further details about zero-velocity detection.

When the system is stationary, the estimated velocity as of (6) can be treated as a pseudo-measurement of the velocity estimation error. Together with a deviation model of (1)-(5)

$$\delta\mathbf{x}_n = \mathbf{F}_n\delta\mathbf{x}_{n-1} + \mathbf{w}_n \tag{7}$$

where $\delta\mathbf{x}_n$ is the deviation of the estimated navigation states form the true states, this can be used to estimate $\delta\mathbf{x}_n$ with a Kalman type filter. This gives the so called INS aiding. For further details on this see [8]. Note that, as argued in [9], systematic sensor errors are difficult to model and estimate and therefore no such states are included in $\delta\mathbf{x}_n$.

The final equations used by our ZUPT-aided INS are

Initialization: $\hat{\mathbf{x}}_0 \leftarrow \mathrm{E}[\mathbf{x}_0]$, $\mathbf{P}_0 \leftarrow \mathrm{cov}(\mathbf{x}_0)$
Loop: n = 1 **to end of data**

% Time update
$\hat{\mathbf{x}}_n = f_{\text{mech}}(\hat{\mathbf{x}}_{n-1}, \mathbf{f}_n, \omega_n)$
$\mathbf{P}_n = \mathbf{F}_n\mathbf{P}_{n-1}\mathbf{F}_n^T + \mathbf{G}\mathbf{Q}\mathbf{G}^T$
% Measurement update
if $T(\{\boldsymbol{\omega}^i, \mathbf{f}^i\}_{W_n}) < \gamma$
  $\mathbf{K}_n = \mathbf{P}_n\mathbf{H}^T(\mathbf{H}\mathbf{P}_n\mathbf{H}^T + \mathbf{R})^{-1}$
  $\delta\hat{\mathbf{x}}_n = \mathbf{K}_n\hat{\mathbf{v}}_n$ (8)
  $\mathbf{P}_n \leftarrow \mathbf{P}_n(\mathbf{I} - \mathbf{K}_n\mathbf{H})$
  % Compensate internal states
  $\begin{bmatrix} \hat{\mathbf{p}}_n \\ \hat{\mathbf{v}}_n \end{bmatrix} \leftarrow \begin{bmatrix} \hat{\mathbf{p}}_n \\ \hat{\mathbf{v}}_n \end{bmatrix} + \begin{bmatrix} \delta\hat{\mathbf{p}}_n \\ \delta\hat{\mathbf{v}}_n \end{bmatrix}$
  $\hat{\mathbf{R}}_n \leftarrow (\mathbf{I}_3 - \mathbf{\Delta}_n)\hat{\mathbf{R}}_n$
  $\delta\hat{\mathbf{x}}_n \leftarrow \mathbf{0}$

where $\mathbf{P}_n = \mathrm{cov}(\delta\hat{\mathbf{x}}_n)$ is the error covariance matrix, $\mathbf{G}$ is the process noise matrix, $\mathbf{Q} = \mathrm{cov}(\mathbf{w}_k)$, $\mathbf{H} = [\mathbf{0}_3\,\mathbf{I}_3\,\mathbf{0}_3]$ is the observation matrix, $\mathbf{K}$ is the Kalman gain, and

$$\mathbf{\Delta}_n = \begin{bmatrix} 0 & -\delta\mathbf{x}_n^{yaw} & \delta\mathbf{x}_n^{pitch} \\ \delta\mathbf{x}_n^{yaw} & 0 & -\delta\mathbf{x}_n^{roll} \\ -\delta\mathbf{x}_n^{pitch} & \delta\mathbf{x}_n^{roll} & 0 \end{bmatrix}.$$

The above algorithm is of closed-loop complementary type where for each iteration n, the estimated state $\hat{\mathbf{x}}_n$ is corrected by the additional measurement (the ZUPT) through the estimated deviation $\delta\hat{\mathbf{x}}_n$. This is the customary way of dead-reckoning with a ZUPT-aided INS. However, direct implementation of a smoothing algorithm is not possible. The next section explains the motivation and introduces the modifications done to the algorithm in order to achieve a smoothed ZUPT-aided INS.

III. SMOOTHING

The customary algorithm (8) gives the behavior illustrated in Fig. 1. The problem is that the information provided by the

ZUPTs is abruptly introduced at the end of the step. This can be mitigated by smoothing.

A. General smoothing

The goal of a smoothing estimation process is to determine the estimated state vector $\hat{\mathbf{x}}_{n|N}$, where a subscript $n|N$ is used to denote the estimate of the nth time instant given all information (in our case, the ZUPTs) up to N where $n < N$. This is the so called *smoothing problem*. We have analyzed different algorithms englobed in the *fixed-interval* smoothing problem, which aims to calculate $\hat{\mathbf{x}}_{n|N}$ from a fixed set of measurements $\{\tilde{\mathbf{y}}_0, \tilde{\mathbf{y}}_1 ... \tilde{\mathbf{y}}_N\}$ for every $n \in \{0, 1, ..., N\}$. Fixed-interval smoothing problems have been the considered smoothing algorithms because the structure of the signal is considered to fit well with this method. By dividing the signal in segments directly related to a step, the information provided by the ZUPT at the end of the step giving the sharp corrections can be made available throughout the whole step via a smoothing algorithm. For many applications, some degree of lag (non-causality) can be tolerated such that smoothing can be carried out step-by-step and, thus, a near real-time behavior of the smoothing is achieved. Among the different types of fixed-interval smoothing algorithms, the Rauch-Tung-Striebel (RTS) formula has been used, since it presents a straightforward relationship with the previous customary ZUPT-aided INS algorithm. The RTS formula is

$$\begin{aligned}
&\textbf{Loop: } n = s_{\text{end}} - 1 \textbf{ to } s_{\text{start}} \\
&\quad \mathbf{A}_n = \mathbf{P}_{n|n}\boldsymbol{\Gamma}_n^T \mathbf{P}_{n+1|n}^{-1} \\
&\quad \hat{\boldsymbol{\chi}}_{n|s_{\text{end}}} = \hat{\boldsymbol{\chi}}_{n|n} + \mathbf{A}_n(\hat{\boldsymbol{\chi}}_{n+1|s_{\text{end}}} - \hat{\boldsymbol{\chi}}_{n+1|n}) \\
&\quad \mathbf{P}_{n|s_{\text{end}}} = \mathbf{P}_{n|n} + \mathbf{A}_n(\mathbf{P}_{n+1|s_{\text{end}}} - \mathbf{P}_{n+1|n})\mathbf{A}_n^T
\end{aligned} \tag{9}$$

where $\hat{\boldsymbol{\chi}}_{n|n}$ is some arbitrary state vector, $\boldsymbol{\Gamma}_n$ is some related system matrix, and the smoothing has been applied over the interval $[s_{\text{end}}, s_{\text{start}}]$ where $s_{\text{end}} > s_{\text{start}}$. The initial conditions $\hat{\boldsymbol{\chi}}_{n|n}$ and $\mathbf{P}_{n|n}$ are provided by the forward Kalman filter.

The RTS formula (9) cannot be directly applied to the estimation (8). This is because of the internal compensation in (8) changing the value of $\delta\hat{\mathbf{x}}_n$ preventing smoothing from being directly applied to it. This problem can be solved by simply avoiding the internal compensation and instead propagate (8) open-loop. In this case the deviation state estimates need to be propagated with

$$\delta\hat{\mathbf{x}}_{n|n-1} = \mathbf{F}_n \delta\hat{\mathbf{x}}_{n-1|n-1}$$

since they are no longer set to zero. By running the filter open-loop, the estimation formula can directly be applied to the deviation states $\delta\hat{\mathbf{x}}_{n-1}$

$$\begin{aligned}
&\textbf{Loop: } n = s_{\text{end}} - 1 \textbf{ to } s_{\text{start}} \\
&\quad \mathbf{A}_n = \mathbf{P}_{n|n}\mathbf{F}^T \mathbf{P}_{n+1|n}^{-1} \\
&\quad \delta\hat{\mathbf{x}}_{n|s_{\text{end}}} = \delta\hat{\mathbf{x}}_{n|n} + \mathbf{A}_n(\delta\hat{\mathbf{x}}_{n+1|s_{\text{end}}} - \delta\hat{\mathbf{x}}_{n+1|n}) \\
&\quad \mathbf{P}_{n|s_{\text{end}}} = \mathbf{P}_{n|n} + \mathbf{A}_n(\mathbf{P}_{n+1|s_{\text{end}}} - \mathbf{P}_{n+1|n})\mathbf{A}_n^T.
\end{aligned}$$

However, once the smoothing has been applied, nothing prevents the internal compensation from being implemented.

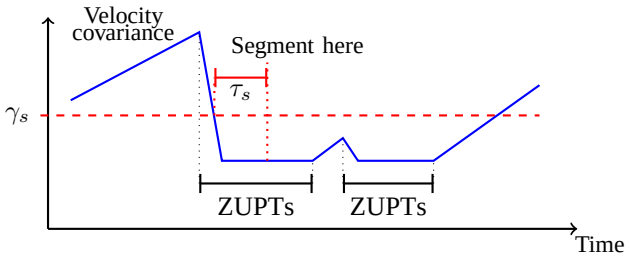

Fig. 2: Velocity error covariance increases along a step and decreases drastically when the ZUPTs are applied. During the steady phase of a step, a no-ZUPT decision can be made. These erroneously decided segments are short; the covariance increases but does not trespass the selected covariance threshold.

B. Data segmentation

The aim of choosing a fixed-interval smoothing problem is to implement a near to real-time smoothing algorithm by applying the smoothing in a step-wise manner. As pointed out, the fixed-interval problem fits with the step by step structure of the signal. However, some kind of segmentation rule that allows to create the data segments to be smoothed is still needed. Since the measurements (the ZUPTs) are irregularly spaced and appear in clusters, we propose a varying-lag smoothing rule based on measurement availability and covariance and timing thresholds. Throughout a step, the velocity error covariance monotonically increases until the stationary phase of the step is detected, when the error covariance drastically decreases. However, the detection of the ZUPT intervals is not perfect and during the stance phase a no-ZUPT decision can be made. Nevertheless, these determined no-ZUPT segments during the steady phase of the step do not last for a long time. Soon, a ZUPT is detected again and the covariance decreases once more. Since these erroneously determined no-ZUPT segments are short, the velocity error covariance cannot increase as much as during the non-stationary phase of the step. Therefore, to properly segment a step, a sum of the velocity error covariances threshold γ_s to be crossed top-down is fixed to decide the segmentation, as shown in Fig.2. The threshold must be high enough to not be affected by the error covariance increase during these erroneously decided short no-ZUPT segments.

Unfortunately, direct segmentation at the point where the velocity error covariance threshold γ_s is trespassed leads to an incorrect behavior of the smoothing algorithm. At this point, the velocity error covariance has not converged yet and hence the information provided by the ZUPT is not fully available in the segment. Despite this, if the segmentation is applied a constant time after the crossing of this covariance threshold, the information given by the ZUPT is available. Therefore, a time threshold τ_s [s] is fixed. When the covariance threshold γ_s is trespassed, the Kalman filter continues running normally for the next τ_s seconds, when the segment is cut. The proposed segmentation rule is summarized in Fig. 2.

C. Suggested 3-pass algorithm

Considering the specific features shown along this section, the proposed smoothing algorithm is given in Alg. 1. The algorithm is a 3-pass algorithm. On the 1st pass, it runs the open-loop ZUPT-aided INS. The forward run is temporarily suspended, by the data segmentation, giving a 2nd backward pass adding the smoothing. Finally, the algorithm performs a 3rd forward pass in which the estimated deviations are used to correct the navigational states. The last pass continues up to the point where the 1st forward pass stopped, where the 1st pass forward continues again. The 3rd forward pass effectively closes the loop and, therefore, the algorithm can be viewed as mixed open-closed-loop filtering.

It should be noted that compared to (8), the memory requirements increase, since for each segment storing $\delta\hat{\mathbf{x}}$ and $\mathbf{P}$ is necessary for the smoothing. However, the covariance increases rather linearly and if memory is a concern, a subset of the covariance values $\mathbf{P}$ could be stored and used to interpolate the rest on the backward pass.

IV. Experimental results

We have compared the smoothing effect over two different implementations of the smoothing algorithm. The first is a *segmented* smoothed ZUPT-aided INS which corresponds to the formulas shown in Alg.1 with $\tau_s = 0.04$ [s.]; whereas the second corresponds to a *non-segmented* smoothed ZUPT-aided INS which corresponds to the same formulas but without evaluating the segmentation rule ($\tau_s = \infty$). Hence, the non-segmented smoothed ZUPT-aided INS corresponds to a smoothing of the whole data set (off-line processing of the data). Thus, consequences of near real-time processing can be compared with off-line processing of the data, where the information provided by all of the future ZUPTs is known. All data has been collected with OpenShoe units [5].

The effect of the smoothing algorithm in an estimated trajectory is shown in Fig. 3. This figure shows the achieved smoothing effect compared with the estimated trajectory by the customary (8). The estimated smoothed trajectories are almost perfectly overlapping and the differences between the segmented and non-segmented implementations are small. Fig. 4 shows the result of the smoothing over multiple steps, for the segmented ZUPT-aided INS implementation. The graph shows the aligned xy-evolution of 20 steps of a pedestrian walking at 3.5 km/h. It can be seen how the sharp correction from the customary ZUPT-aided INS are nearly negligible for the smoothed ZUPT-aided INS.

In the implementation some sharp corrections in the smoothing have been experienced. These appear when there are large accelerations and rotations and large cross-couplings between heading and the position states. These problems are believed to be due to problems with the linearization in (7). They can be mitigated by zeroing out the cross-coupling between heading and position states. However, in this case the covariance estimates in the filter will not be correct, even though the state estimates do not change significantly.

Algorithm 1 Pseudo code for the proposed 3-pass smoothing algorithm.

Initializ.: $\hat{\mathbf{x}}_0 = E[\mathbf{x}_0]$, $\delta\hat{\mathbf{x}}_0 = 0$, $\mathbf{P}_0 = var(\mathbf{x}_0)$, $c = 0$, $s_{\text{start}} = 1$, s_{end} =“end of data”

Loop while $s_{\text{start}} < s_{\text{end}}$
- % Forward Kalman filter
- **Loop:** $n = s_{\text{start}}$ **to** s_{end}
  - % Time update
  - $\hat{\mathbf{x}}_n = f_{\text{mech}}(\hat{\mathbf{x}}_{n-1}, \mathbf{f}_n, \omega_n)$
  - $\delta\hat{\mathbf{x}}_{n|n-1} = \mathbf{F}_n\delta\hat{\mathbf{x}}_{n-1|n-1}$
  - $\mathbf{P}_{n|n-1} = \mathbf{F}_n\mathbf{P}_{n-1|n-1}\mathbf{F}_n^T + \mathbf{G}\mathbf{Q}\mathbf{G}^T$
  - % Measurement update
  - **if** $T(\{\boldsymbol{\omega}^i, \mathbf{f}^i\}_{W_n}) < \gamma$
    - $\mathbf{K}_n = \mathbf{P}_{n|n-1}\mathbf{H}^T(\mathbf{H}\mathbf{P}_{n|n-1}\mathbf{H}^T + \mathbf{R})^{-1}$
    - $\delta\hat{\mathbf{x}}_{n|n} = \delta\hat{\mathbf{x}}_{n|n-1} - \mathbf{K}_n(\delta\hat{\mathbf{v}}_{n|n-1} - \hat{\mathbf{v}}_n)$
    - $\mathbf{P}_{n|n} = \mathbf{P}_{n|n-1}(\mathbf{I} - \mathbf{K}_n\mathbf{H})$
  - % Segmentation rule eval.
  - **if** $c > 0$
  - $c = c + T_s$
  - **if** $\|\text{diag}(\mathbf{P}^v_{n-1})\| > \gamma_s \ \wedge \ \|\text{diag}(\mathbf{P}^{vel}_n)\| \leq \gamma_s \ \wedge c = 0$
  - $c = T_s$
  - **if** $c > \tau_s$
    - $s_{\text{end}} \leftarrow n$
    - **break loop**
- % Smoothing
- **Loop:** $n = s_{\text{end}} - 1$ **to** s_{start}
  - $\mathbf{A}_n = \mathbf{P}_{n|n}\mathbf{F}^T\mathbf{P}^{-1}_{n+1|n}$
  - $\delta\hat{\mathbf{x}}_{n|s_{\text{end}}} = \delta\hat{\mathbf{x}}_{n|n} + \mathbf{A}_n(\delta\hat{\mathbf{x}}_{n+1|s_{\text{end}}} - \delta\hat{\mathbf{x}}_{n+1|n})$
  - $\mathbf{P}_{n|s_{\text{end}}} = \mathbf{P}_{n|n} + \mathbf{A}_n(\mathbf{P}_{n+1|s_{\text{end}}} - \mathbf{P}_{n+1|n})\mathbf{A}_n^T$
- % Internal state compensation
- **Loop:** $n = s_{\text{start}}$ **to** s_{end}
  - $\begin{bmatrix} \hat{\mathbf{p}}_n \\ \hat{\mathbf{v}}_n \end{bmatrix} \leftarrow \begin{bmatrix} \hat{\mathbf{p}}_n \\ \hat{\mathbf{v}}_n \end{bmatrix} + \begin{bmatrix} \delta\hat{\mathbf{p}}_{n|s_{\text{end}}} \\ \delta\hat{\mathbf{v}}_{n|s_{\text{end}}} \end{bmatrix}$
  - $\hat{\mathbf{R}}_n \leftarrow (\mathbf{I}_3 - \boldsymbol{\Delta}_{n|s_{\text{end}}})(\hat{\mathbf{R}}_n)$
  - $\delta\hat{\mathbf{x}}_n \leftarrow \mathbf{0}$
- $s_{\text{start}} = s_{\text{end}} + 1$, s_{end} = “end of data”, $c = 0$

Fig. 5 shows the smoothing effect over the velocity error covariance. For (8), the error covariance increases along a step until the information provided by the ZUPT becomes available, where the error covariance decreases drastically. For the smoothed implementations, the information provided by the future ZUPTs is also available. Therefore, the highest error covariance value is in the middle of the step, which is

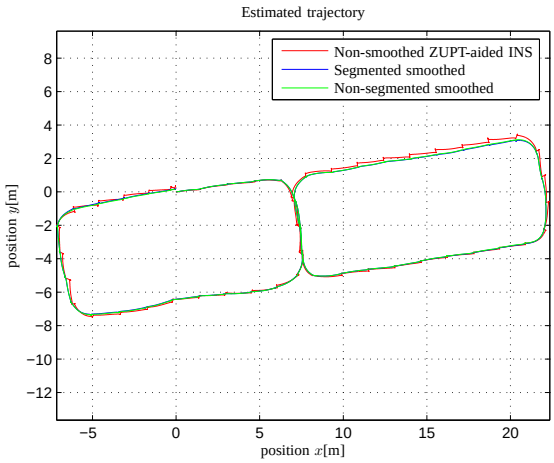

Fig. 3: Effect of smoothing over a trajectory. The large corrections at the end of each step have been smoothed. Note that the segmented and non-segmented smoothed trajectories are essentially overlapping.

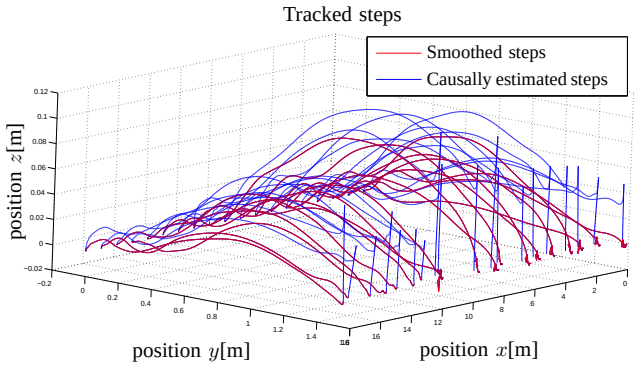

Fig. 4: Effect of smoothing over multiple steps. The large corrections at the end of each step have been smoothed. Note the difference in scale between the xy-plane and the z-axis.

the furthest point from any ZUPT, and the covariance looks symmetrical over the step.

On the other side, Fig. 6 shows the smoothing effect over the position error covariance. In the non-smoothed ZUPT-aided INS, the position error covariance increases over time and decreases when there is a ZUPT. However, the ZUPT does not completely decorrelate the position components (in contrast with the velocity components). Hence, the position error covariance increases over time. Besides, the position error covariance evolution throughout the step is increasing until the ZUPT is detected, where it suddenly decreases. For the smoothed implementations, the information provided by the ZUPT is available along a step. Thus, even though the position error covariance increases along the whole trajectory, it has no sharp corrections at the end of each step.

The proposed smoothing method is dependent on the length of the ZUPT segments detected, as well as the velocity of the pedestrian. Therefore, depending on the application, some tuning of the varying-lag rule and ZUPT detection thresholds could be necessary.

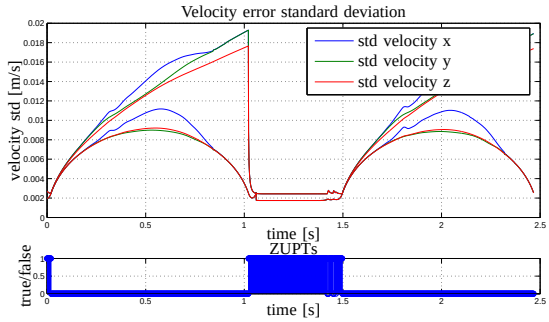

Fig. 5: Typical effect of smoothing over the velocity error covariance throughout two steps.

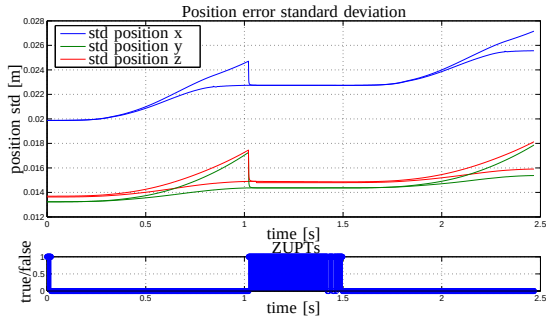

Fig. 6: Typical effect of smoothing over the position error covariance throughout two steps.

V. Conclusion

In this article we have suggested an smoothing algorithm for ZUPT-aided INSs, which has been shown to eliminate the discontinuities at the end of each step. The proposed method is based on a 3-pass mixed open-closed-loop filter. Consequences of smoothing have been illustrated, analyzed and quantified over a test trajectory, over multiple steps and over the error covariance values.

References

[1] E. Foxlin, "Pedestrian tracking with shoe-mounted inertial sensors," *Computer Graphics and Applications, IEEE*, vol. 25, pp. 38 –46, 2005.

[2] L. Ojeda and J. Borenstein, "Non-GPS navigation for security personnel and first responders," *Journal of Navigation*, vol. 60, pp. 391–407, 2007.

[3] S. Godha and G. Lachapelle, "Foot mounted inertial system for pedestrian navigation," *Measurement Science and Technology*, vol. 19, 2008.

[4] Ö. Bebek, M. Suster, S. Rajgopal, M. Fu, X. Huang, M. Çavuşoğlu, D. Young, M. Mehregany, A. van den Bogert, and C. Mastrangelo, "Personal navigation via high-resolution gait-corrected inertial measurement units," *Instrumentation and Measurement, IEEE Transactions on*, vol. 59, pp. 3018 –3027, nov. 2010.

[5] J.-O. Nilsson, I. Skog, P. Händel, and K. Hari, "Foot-mounted INS for everybody – An open-source embedded implementation," in *Proc. IEEE/ION PLANS*, pp. 140 –145, april 2012.

[6] X. Yun, J. Calusdian, E. Bachmann, and R. McGhee, "Estimation of human foot motion during normal walking using inertial and magnetic sensor measurements," *Instrumentation and Measurement, IEEE Transactions on*, vol. 61, pp. 2059 –2072, july 2012.

[7] I. Skog, P. Händel, J.-O. Nilsson, and J. Rantakokko, "Zero-velocity detection – An algorithm evaluation," *Biomedical Engineering, IEEE Transactions on*, vol. 57, pp. 2657 –2666, nov. 2010.

[8] J. Farrell and M. Barth, *The global positioning system & Inertial Navigation*. Mc Graw Hill, 1998.

[9] J.-O. Nilsson, I. Skog, and P. Händel, "A note on the limitations of ZUPTs and the implications on sensor error modeling," in *Proc. IPIN*, 2012.