Universidad Zaragoza
1542

Proyecto Fin de Carrera

Step-wise smoothing para ZUPT-aided INSs

Autor

David Simón Colomar

Director: John-Olof Nilsson

Ponente: Enrique Masgrau Gómez

Director: John-Olof Nilsson
Ponente: Enrique Masgrau Gómez

Ingeniería Superior de Telecomunicación
Departamento de Ingeniería Electrónica y de Comunicaciones
Escuela de Ingeniería y Arquitectura
Septiembre de 2012

Resumen

Step-wise smoothing para ZUPT-aided INSs

Debido a la naturaleza recursiva de la mayoría de los sistemas de navegación inercial (Inertial navigation systems, INSs) foot-mounted zero-velocity-updated-aided (ZUPT-aided), la covarianza del error va incrementando a lo largo de cada paso y “colapsa” al final de éste, donde se hace la corrección debido a la ZUPT. Esto da lugar a bruscas correcciones y discontinuidades en la trayectoria estimada. Para aplicaciones con estrictas exigencias de tiempo real este comportamiento es inevitable, ya que cada estimación corresponde a la mejor estimación usando toda la informacion hasta ese instante de tiempo. Sin embargo, para muchas aplicaciones un cierto grado de retardo (no causalidad) puede ser tolerado y la información proporcionada por las ZUPTs al final del paso, que causa las correcciones abruptas, puede hacerse disponible a lo largo de todo el paso. En consecuencia la implementación de un filtro de alisado (smoothing) para un ZUPT-aided INS es considerada en esta tesis para eliminar las correcciones bruscas y la covarianza no simétrica a lo largo de los pasos. Que sepamos, no se ha presentado tratamiento formal de smoothing para sistemas ZUPT-aided INS, pese a que existe una gran variedad de literatura acerca del tema general de smoothing.

Debido al habitual filtro complementario de lazo cerrado empleado en aided INSs, las distintas técnicas de smoothing estándar no se pueden aplicar directamente. Además las medidas (las ZUPTs) están espaciadas irregularmente y aparecen en grupos. Por tanto, se requiere de algún tipo de regla de smoothing de retardo variable. En este proyecto se sugiere un método basado en una mezcla de filtro complementario de lazo abierto-cerrado combinado con un smoothing Rauch-Tung-Striebel (RTS). Se analizan distintos tipos de reglas de smoothing de retardo variable. Para aplicaciones próximas a tiempo real, el smoothing se aplica a los datos paso a paso. Los intervalos (pasos) para el smoothing se determinan en base a disponibilidad de las medidas y a umbrales de tiempo y covarianza. Por otro lado, para un procesado completo off-line, se analizan sets completos de datos. Finalmente, se cuantifican las consecuencias del smoothing y del filtro en bucle abierto-cerrado basándose en datos reales. El impacto del smoothing se ilustra y analiza a lo largo de los pasos.

Abstract

Step-wise smoothing for ZUPT-aided INSs

Due to the recursive nature of most foot-mounted zero-velocity-update-aided (ZUPT-aided) inertial navigation systems (INSs), the error covariance increases throughout each step and "collapses.^{a}t the end of the step, where the ZUPT correction is done. This gives sharp corrections and discontinuities in the estimated trajectory. For applications with tight real-time constraints, this behavior is unavoidable, since every estimate corresponds to the best estimate given all the 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, giving the sharp correction, can be made available throughout the step. Consequently, to eliminate the sharp corrections and the unsymmetrical covariance over the steps, the implementation of a smoothing filter for a ZUPT-aided INS is considered in this thesis. To our knowledge, no formal treatment of smoothing for such systems has previously been presented, even though an extensive literature on the general subject exists.

Owing to the customary closed-loop complementary filtering used for aided INS, standard smoothing techniques cannot directly be applied. Also since the measurements (the ZUPTs) are irregularly spaced and appear in clusters, some varying-lag smoothing rule is necessary. Therefore, a method based on a mixed open-closed-loop complementary filtering combined with a Rauch-Tung-Striebel (RTS) smoothing is suggested in this thesis. Different types of varying-lag smoothing rules are examined. For near real-time applications, smoothing is applied to the data in a step-wise manner. The intervals (steps) for the smoothing are determined based on measurement availability and covariance and timing thresholds. For complete off-line processing, full data set smoothing is examined. Finally, the consequences of the smoothing and the open-closed-loop filtering are quantified based on real data. The impact of the smoothing throughout the steps is illustrated and analyzed.

Abstrakt

Step-wise smoothing för ZUPT-aided INSs

På grund av den rekursiva karaktären hos de flesta nollhastighet-suppdaterade tröghetsnavigeringssystem (på engelska, ZUPT-aided INSs), ökar felets kovarians över varje steg och "kollapsar" i slutet av steget, där nollhastighetsuppdateringen görs. Detta ger kraftiga korrigeringar och diskontinuiteter i den skattade banan. För tillämpningar med håda realtidsbegränsningar, är detta beteende oundvikligt eftersom varje skattning motsvarar den bästa uppskattningen få all information fram till den tidpunkten. För många tillämpningar kan en viss grad av eftersläpning (icke-kausalitet) emellertid tolereras och den information som nollhastighetsuppdateringarna tillför vid slutet av ett steg, vilket ger kraftig korrigering, kan göras tillgängliga i hela steg. För att eliminera de kraftiga korrigeringar och osymmetriska kovariansen över stegen, behandlads i detta examensarbete implementeringen av ett glättningsfilter för ett nollhastighetsuppdaterade tröghetsnavigeringssystem. Såvitt vi vet har ingen formell behandling av glättningsfilter för sådana system tidigare lagts fram, trots att en omfattande litteratur i ämne finns.

På grund av de brukliga återkpllade filtrering som används för understödda tröghetesnavigering, kan vanliga glättningstekniker inte direkt tillämpas. Eftersom ätningarna (nollhastighetsuppdateringarna) är oregelbundet placerade och förekommer i kluster, behövs ett glättningsfilter med varierande-fördröjning användas. Därför föreslås en metod som byggs på en växelvis öppen/återkopplad filtrering i kombination med en Rauch-Tung-Striebel (RTS) glättning i detta examensarbete. Olika metoder för återkoppling och glättning undersöks. För nära realtidstilämpningar tillämpas glättningen stegvis på datan. Intervallen (stegen) för glättningen bestäms på grundval av mätningarnas tillgänglighet och filterkovariansens värden i förhållande till tröskelvärden. För tillämpningar helt utan krav på eftersläpning undersöks användet av hela datamängd för glättning. Slutligen undersöks konsekvenserna av glättningen och de öppna-slutna filtrering kvantifieras baseras på verkliga data. Effekten av glättningen längs stegen illustreras och analyseras.

Agradecimientos

Estocolmo es el lugar donde he hecho esta tesis. Pero del mismo modo, Estocolmo es el lugar donde he pasado uno de los mejores años de mi vida. Ahora es el momento de agradecer a todo el mundo que me ha ayudado a hacer esta tesis, y que tambien me ha permitido disfrutar de este increíble año en el norte.

En primer lugar, quiero mostrar mi más sincera gratitud a mi supervisor John-Olof Nilsson por su continuo apoyo durante estos últimos meses. He disfrutado realmente trabajando con él.

También quiero agradecer a Andrea Rizzi, Jean-Marie Le Bourhis, David Aguilar Pérez y en general todos los amigos que han compartido conmigo horas de estudio, trabajo, frustraciones... y sí, risas y buenas experiencias también. Sin ellos, escribir esta tesis no habría sido lo mismo.

Del mismo modo, quiero dar gracias a mis amigos, tanto Erasmus como de España, que me han hecho disfrutar de uno de los mejores periodos de mi vida y que a veces han escuchado, más allá de los límites de la relatividad, mis problemas.

Por último, pero no por ello menos importante, quiero mostrar gratitud a mi familia. Por años de paciencia, ánimo y guía. Finalmente estoy aquí, cuando mi carrera llega a su fin. Sin ellos, esto no sería posible.

Muchas gracias a todos, de corazón.

Índice general

Índice de figuras

Capítulo 1

Introducción

Un sistema de navegación inercial (inertial navigation system, INS) foot-mounted permite el dead-reckoning de personas. La naturaleza recursiva de la mayoría de estos sistemas causa que estén sujetos a errores acumulativos. Para foot-mounted INSs, el error estimado en la posición y la velocidad de la persona va incrementándose a lo largo de cada paso y "colapsa" al final de éste, donde una zero-velocity-update (ZUPT) proporciona información que permite la corrección de las estimaciones dando lugar a correcciones abruptas y por tanto a discontinuidades en la trayectoria estimada. El objetivo de este proyecto fin de carrera es diseñar un filtro de smoothing (*alisado*) que evite la aparición de estas discontinuidades. A lo largo de este primer capítulo se proporciona una visión general del sistema disponible, motivando a partir de éste la necesidad de un smoothing paso a paso para ZUPT-aided INSs. Finalmente, se indica la estructura del resto del proyecto fin de carrera, así como las principales referencias para su mejor seguimiento.

1.1. Foot-mounted ZUPT-aided INS

Dead-reckoning es el proceso de estimación de la posición actual de un cuerpo combinando la posición y velocidad previamente determinadas de éste con la velocidad conocida o estimada del cuerpo. En particular, un sistema de navegación inercial (inertial navigation system, INS) consiste en un conjunto de sensores que permiten calcular continuamente la posición, velocidad y orientación (attitude) de un cuerpo aplicando leyes físicas de movimiento a un estado inicial conocido del sistema. Los sensores inerciales que pueden vestirse en el cuerpo, como el foot-mounted INS empleado en este proyecto fin de carrera, poseen propiedades deseables para el seguimiento de personas en situaciones donde otros sistemas de localización (como el GPS) fallan, como puede ser en interiores o en bosques densos. En estas situaciones un INS da estimaciones fiables, ya que un INS emplea leyes físicas de movimiento que no requieren la detección de señales externas que podrían estar bajo el efecto de interferencias. Por tanto, el dead-reckoning de personas mediante INSs ha sido propuesto para muchas aplicaciones, como en defensa, para trabajadores de rescate y/o emergencias, análisis médico de patrones de pasos, así como para oficinas inteligentes.

OpenShoe es una implementación open source de un foot-mounted INS que incluye diseño tanto hardware como software. Es el resultado de la colaboración entre el Laboratorio de Procesado de Señal del Instituto Real de Tecnología (en sueco Kungliga Tekniska Högskolan, KTH) de Estocolmo con el Departamento de Procesado Estadístico de Señal en el Instituto de Ciencia Indio (Indian Institute of Science, IISc) en Bangalore, India. La Fig.1.1. muestra una implementación del sistema. A la derecha aparece una sección de un zapato donde se ve la posición en que se encuentra el INS, embebido en la suela del zapato. En [12] puede encontrarse una implementación del sistema completamente documentada, que es sobre la que este proyecto fin de carrera va a trabajar.

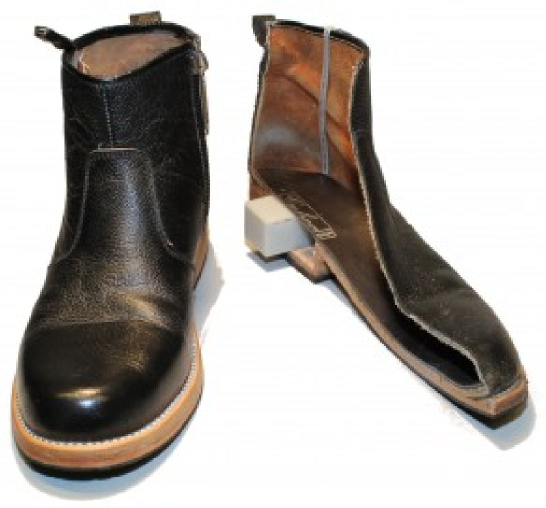


Figura 1.1: Implementación de un OpenShoe.

La principal ventaja de este sistema es que mide movimiento, mientras que la mayoría de los sistemas de navegación para personas intentan estimar el tipo y la longitud del paso que ha dado la persona. Esto se convierte en un problema cuando el movimiento es irregular. Sin embargo, como el sistema empleado mide movimiento, no importa si la persona corre, salta o camina; nuestro sistema detectará el movimiento y por tanto proporcionará una posición correctamente. El error en la estimación del ZUPT-aided INS puede ser tan bajo como el 0.14 % del trayecto total, para trayectorias en línea recta, y en general será al menos igual o mejor que sistemas equivalentes. Del mismo modo, otros valores importantes de este foot-mounted INS son su modularidad y su pequeño peso, volumen y precio en comparación con los típicos sistemas de investigación de ordenador más sensor. Estos rasgos permitirán equipar a un gran número de usuarios con unidades foot-mounted INS. El primer trabajo con foot-mounted ZUPT-aided INSs es [5], artículo que proporciona un buen trasfondo de este tema.

El objetivo de un foot-mounted INS es proporcionar un dead-reckoning fiable. Esto es, busca estimar con precisión la posición y la velocidad de una persona siguiendo los movimientos que ésta efectúa a partir de un estado inicial conocido. Este seguimiento tiene una estructura recursiva. Como el actual estado estimado depende del previo, conforme el tiempo pasa el error en la estimación aumenta. Por tanto, es necesario proporcionar una corrección a las estimaciones de posición y velocidad cada breves periodos de tiempo.

Un foot-mounted zero-velocity-updated-aided (ZUPT-aided) INS proporciona esta corrección mientras el zapato se encuentra en estado estacionario. Un estado estacionario tiene lugar mientras la suela del zapato dondel INS está embebido está en contacto con el suelo. Durante este intervalo de tiempo, el sistema de navegación inercial está estacionario, y la velocidad estimada debería ser cero.

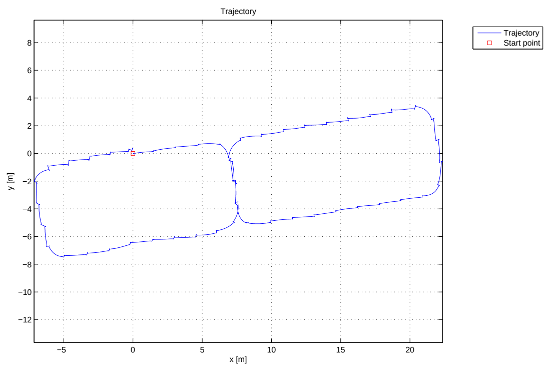
Si no es cero, la velocidad estimada por el sistema se trata como una medida del error en la velocidad y puede ser empleada para corregir la estimación de la posición y la velocidad. Como estos estados estacionarios ocurren como mucho cada pocos segundos, esta corrección puede hacerse regularmente y por tanto evitar que el error en la estimación cometido por el INS crezca demasiado.

El proceso de seguimiento de los ZUPT-aided INSs puede ser dividido en tres diferentes fases para cada instante de tiempo: primero, la captura de datos por los sensores en la unidad de medición inercial (inertial measurement unit, IMU); segundo, la detección de intervalos de ZUPT en los que el zapato está estacionario; y tercero, el procesado mediante un filtro de tipo Kalman para ZUPT-aided INS. Este filtro de Kalman proporciona para cada instante de tiempo una estimación de la posición y la velocidad de la persona combinando las nuevas medidas proporcionadas por la IMU con el estado estimado previamente, usando leyes de mecánica. Después, esta estimación es corregida por una ZUPT si se detecta que el pie está en una fase estacionaria, ya que durante esta fase el error es conocido.

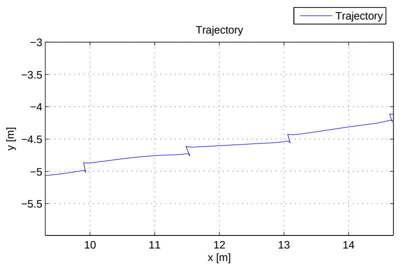
Como aplicación en tiempo real, estos tres pasos se hacen continuamente tan pronto como las medidas están disponibles, haciendo posible el seguimiento del movimiento. Sin embargo, la implementación de Matlab sobre la que este proyecto fin de carrera trabaja estos pasos se llevan a cabo por completo secuencialmente uno tras de otro. Una implementación de Matlab de este sistema se encuentra disponible en www.openshoe.org.

1.2. Motivación del proyecto fin de carrera

Debido a la propia naturaleza del sistema, en la trayectoria estimada por un ZUPT-aided INS aparecen discontinuidades . Una trayectoria estimada típica se muestra en Fig.1.2a, y una ampliación de tres pasos de esta trayectoria se muestra en 1.2b.



(a) Trayectoria estimada para los ejes xy



(b) Visión detallada de tres pasos para los ejes xy

Figura 1.2: Trayectoria estimada para los ejes xy

Del análisis de estas figuras fácilmente se puede apreciar una especie de *picos* que dan a la trayectoria estimada una apariencia no deseable. Esta situación tambien aparece en el eje z, así como en las estimaciones de la velocidad. La situación es debida a la corrección llevada a cabo por el filtro de Kalman cuando la información proporcionada por una ZUPT se hace disponible. A lo largo de un paso, el error en la estimación va aumentando y por tanto al final del paso, justo antes de que la ZUPT se detecte, el error es grande. Por tanto, cuando la información proporcionada por la ZUPT se hace disponible la corrección efectuada es grande, apareciendo los *picos* que se han mostrado para cada paso.

Para aplicaciones en estricto tiempo real este comportamiento es inevitable, ya que cada estimación corresponde a la mejor estimación considerando toda la información disponible hasta ese instante de tiempo. Sin embargo, para visualización y análisis de movimiento, estas grandes correcciones son indeseables. Como muchas aplicaciones admiten cierto grado de retardo (no causalidad) y la información provista por las ZUPTs al final del paso, causando las correcciones abruptas, puede hacerse disponible a lo largo de todo el paso, en este proyecto fin de carrera se considera la implementación de un filtro de smoothing para ZUPT-aided INSs, buscando eliminar las correcciones bruscas y la covarianza no simétrica a lo largo de los pasos.

Que sepamos, no se ha presentado tratamiento formal de smoothing para sistemas ZUPT-aided INS, pese a la gran variedad de información existente en el ámbito de filtros de smoothing. Consideramos que este filtro de smoothing puede ser muy útil para la futura investigación en el área. Por tanto, el objetivo de este proyecto fin de carrera es implementar un algoritmo específico de smoothing para ZUPT-aided INSs.

El proyecto fin de carrera se ha llevado a cabo en el Laboratorio de Procesado de Señal en el Instituto Real de Tecnología (KTH) de Estocolmo, con la supervisión de John-Olof Nilsson. El examinador es Peter Händel. El ponente en Zaragoza es Enrique Masgrau Gómez.

1.3. Definición del problema

Debido al usual filtro complementario de lazo cerrado empleado por los aided INSs, las distintas técnicas estándar de smoothing no pueden ser aplicadas directamente, ya que nuevos aspectos no considerados en la bibliografía que trata el problema general de smoothing deben ser considerados. El trabajo de este proyecto fin de carrera consiste en evaluar distintas opciones de implementación e implementar un algoritmo de smoothing para ZUPT-aided INSs basado en este análisis. Adicionalmente, la implementación de este algoritmo de smoothing se usa para caracterizar las correcciones aplicadas sobre un paso con respecto a las correcciones debidas a la implementación recursiva original.

Como las discontinuidades aparecen paso a paso, el smoothing de un único paso es considerado. Además, se desea la obtención de un sistema cuyo funcionamiento pueda ser en casi tiempo real. Los filtros de smoothing son técnicas no causales de procesado de señal, por lo que su funcionamiento en estricto tiempo real no es posible. Sin embargo, admitiendo cierto retardo (lo cual es posible para muchas aplicaciones) un funcionamiento en casi tiempo real es posible. Ello también requiere de una regla de segmentación para la creación de segmentos de datos más cortos, que pueden ser identificados con un paso. Estos segmentos cortos de datos pueden ser alisados tan pronto como todas las componentes en el segmento (un paso) están disponibles, permitiendo así un funcionamiento en casi tiempo real. Por tanto, en este proyecto fin de carrera también se motivan y diseñan las rutinas para la segmentación automática de los datos.

1.4. Visión general del proyecto fin de carrera

Este proyecto fin de carrera se divide en siete capítulos. A lo largo de este capítulo 1 se ha proporcionado una visión general de los ZUPT-aided INSs, permitiendo introducir el problema a ser resuelto. Durante los siguientes capítulos, se proporciona el conocimiento necesario para alcanzar una comprensión completa de la versión disponible del ZUPT-aided INS. Así, el capítulo 2 recorre en profundidad diferentes aspectos de la implementación, introduciendo las bases de la navegación inercial y proporcionando un modelo de error para ésta. El capítulo 3 trata el filtrado de Kalman: cuáles son sus fundamentos y cómo puede ser usado en un ZUPT-aided INS. El ZUPT-aiding es explicado a continuación, explicando cómo corrige el error acumulado en las estimaciones. Los siguientes capítulos muestran el trabajo llevado a cabo en este proyecto fin de carrera. En el capítulo 4 son analizados distintos algoritmos generales de smoothing que se usan como base para tratar con el problema de smoothing que nos concierne. Tras el análisis de distintos algoritmos, se motiva la elección de uno de éstos para su implementación. El capítulo 5 explica los distintos aspectos de implementación a ser considerados que aparecen a la hora de combinar los ZUPT-aided INSs con los algoritmos generales de smoothing. En particular, el problema de la segmentación de datos es considerado para el subsecuente smoothing de un paso. Del mismo modo, se motiva por qué una implementación en lazo abierto del típico filtro de Kalman usado por ZUPT-aided INSs es empleada para el correcto funcionamiento del sistema. Finalmente, se muestra el algoritmo de smoothing propuesto para ZUPT-aided INSs. El capítulo 6 consiste en el análisis y comparación con la implementación original de distintas implementaciones del algoritmo de smoothing propuesto. Finalmente, en el capítulo 7 se muestran las conclusiones y se propone el posible futuro trabajo a realizar en este área.

La versión completa del proyecto fin de carrera se encuentra en el anexo A, en inglés. En esta versión en castellano se encuentran revisados y por completo los capítulos 1, 5, 6 y 7, así como el algoritmo de smoothing empleado y la motivación para su elección del capítulo 4. En el capítulo 3, a modo ilustrativo para esta versión en castellano, se encuentra un resumen de las ecuaciones empleadas por el filtro de Kalman típico en ZUPT-aided INSs.

Finalmente, en el anexo B se encuentra el artículo "Smoothing for ZUPT-aided INSs". Este artículo es consecuencia del trabajo de este proyecto fin de carrera, y ha sido admitido y enviado para su publicación en la Conferencia Internacional de Posicionamiento y Navegación en Interiores (International Conference on Indoor Positioning and Indoor Navigation (IPIN)), que tendrá lugar entre el 13 y el 15 de Noviembre de 2012.

1.5. Principales referencias

Este proyecto fin de carrera está íntimamente relacionado con los sistemas de navegación inercial. Debido al amplio rango de tecnologías involucradas, a lo largo del texto se intenta motivar y tomar los resultados requeridos para sus explicaciones. Para detalles adicionales y demostraciones completas, aquí damos algunas premisas y referencias al lector.

Las dos principales fuentes que entendemos pueden proporcionar una buena comprensión teórica del sistema son [1] para los capítulos 2 y 3, y [7] para el capítulo 4. La primera proporciona un análisis completo de los INSs, mientras que la segunda introduce los distintos algoritmos generales de smoothing. Es conveniente indicar que en este proyecto fin de carrera no se presentan los sistemas de coordenadas, ecuaciones de rotación de planos, las simplificaciones subyacentes en el uso de los cuaterniones ni las ecuaciones de sistema de espacio-estado; las cuales son la base en las que el modelo de error está basado (las tres primeras), y la base desde la que derivar el filtro de Kalman (la última).

Capítulo 3

ZUPT-aided INS

Este capítulo pretende resumir de forma concisa el funcionamiento del ZUPT-aided INS sobre el que esta tesis trabaja. Para ello, en primer lugar se resume brevemente el modo en que se realiza la navegación inercial. Posteriormente, se motiva cómo el ZUPT-aiding funciona para finalmente dar las ecuaciones empleadas por el sistema disponible, basado en un filtro de tipo Kalman. Para detalles adicionales, los capítulos 2 y 3 del documento original de esta tesis, incluidos en el anexo A, proporcionan una visión y motivación mucho más completa del sistema empleado y sus aproximaciones.

3.1. ZUPT-aided INS

Conceptualmente, un ZUPT-aided INS consiste en una unidad de medición inercial (inertial measurement unit, IMU), que proporciona medidas de fuerza especifica y velocidad angular; y un filtro de tipo Kalman, que proporciona estimaciones del estado de navegación.

En concreto, la mayoría de las foot-mounted IMUs pertenecen al tipo strap-down, donde los ejes del sensor están alineados con los del móvil, en este caso el zapato. Por tanto, las medidas tomadas por la IMU necesitan ser convertidas desde los ejes de coordenadas del sensor a los ejes de coordenadas *fijos* de navegación. En consecuencia, en primer lugar las medidas tomadas por el giroscopio se integran, permitiendo conocer la orientación relativa entre los ejes de coordenadas. La orientación relativa se representa empleando cuaterniones $\mathbf{q}_n$ y se actualiza con

$$\mathbf{q}_n = \left[\cos\Big(\frac{\|\omega_n\|T_s}{2}\Big)\mathbf{I}_4 + \frac{2}{\|\omega_n\|T_s}\sin\Big(\frac{\|\omega_n\|T_s}{2}\Big)\boldsymbol{\Omega}_n\right]\mathbf{q}_{n-1} \tag{3.1}$$

donde $\omega_n = [\omega_n^x, \omega_n^y, \omega_n^z]^T$, T_s es el periodo de muestreo del sistema, n es un índice de tiempo, ω_n^i son las velocidades angulares en torno al eje i, y

$$\boldsymbol{\Omega}_n = \frac{T_s}{2}\begin{bmatrix} 0 & \omega_n^z & -\omega_n^y & \omega_n^x \\ -\omega_n^z & 0 & \omega_n^x & \omega_n^y \\ \omega_n^y & -\omega_n^x & 0 & \omega_n^z \\ -\omega_n^x & -\omega_n^y & -\omega_n^z & 0 \end{bmatrix} \tag{3.2}$$

es la matriz de actualización del cuaternión. La orientación puede ser representada equivalentemente con la matriz de rotación $\mathbf{R}_n \Leftrightarrow \mathbf{q}_n$ o los ángulos de

Euler $\boldsymbol{\theta}_n \Leftrightarrow \mathbf{q}_n$. Por motivos de claridad, emplearemos las distintas notaciones equivalentemente.

Conociendo la orientación actual, la fuerza específica medida por los acelerómetros $\mathbf{f}_n$ se puede expresar en los ejes de navegación. Sobre estos ejes, se puede compensar la aceleración gravitacional $\mathbf{g} = [0, 0, 9{,}81]^T$

$$\mathbf{a}_n = \mathbf{R}_n \mathbf{f}^b - \mathbf{g} \tag{3.3}$$

obteniendo así la aceleración $\mathbf{a}_n$ en los ejes de coordenadas de navegación.

Finalmente, la aceleración inercial $\mathbf{a}_n$ se integra para obtener la posición $\mathbf{p}_n$ y la velocidad $\mathbf{v}_n$. Debido a que la frecuencia del sistema es alta y las variables empleadas discretas, la aceleración $\mathbf{a}_n$ puede considerarse constante entre dos instantes de tiempo, permitiendo emplear las ecuaciones básicas del movimiento como ecuaciones de mecanización

$$\mathbf{p}_n = \mathbf{p}_{n-1} + \mathbf{v}_{n-1} T_s + \frac{1}{2}\,\mathbf{a}_n T_s^2 \tag{3.4}$$

$$\mathbf{v}_n = \mathbf{v}_{n-1} + \mathbf{a}_n T_s. \tag{3.5}$$

Concatenando las estimaciones de posición, velocidad y orientación en un vector de estado de navegación $\mathbf{x}_n = (\mathbf{p}_n, \mathbf{v}_n, \theta_n)$ podemos describir las ecuaciones (3.1)-(3.5) como un sistema de espacio de estados

$$\mathbf{x}_n = f_{\text{mech}}(\mathbf{x}_{n-1}, \mathbf{f}_n, \boldsymbol{\omega}_n). \tag{3.6}$$

Junto con la IMU, este sistema de espacio de estados conforman el INS. Desafortunadamente, el error en la estimación del estado estimado por el INS incrementa rápidamente con el tiempo. Por tanto, se requiere de información adicional para corregir las estimaciones. Nuestro sistema emplea pseudo-medidas en la forma de ZUPTs. La idea subyacente bajo el concepto de ZUPT es detectar el estado en el que el zapato se encuentra estacionario, siendo su velocidad supuestamente cero. El sistema se considera estacionario si

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

donde $T(\cdot)$ son estadísticas de testeo, $\{\boldsymbol{\omega}^i, \mathbf{f}^i\}_{W_n}$ son las medidas inerciales sobre una ventana temporal W_n, y γ es un umbral. Para información adicional sobre la detección de intervalos de velocidad cero, ver [14].

Cuando el sistema está estacionario, la velocidad estimada por (3.6) se puede tratar como una pseudo-medida del error en la estimación de la velocidad. Junto con un modelo de desviación de (3.1)-(3.5)

$$\delta\mathbf{x}_n = \mathbf{F}_n \delta\mathbf{x}_{n-1} + \mathbf{u}_n \tag{3.7}$$

donde $\delta\mathbf{x}_n$ es la desviación de los estados de navegación estimados con respecto a los verdaderos, esta fórmula puede emplearse para estimar $\delta\mathbf{x}_n$ con un filtro de tipo Kalman, proporcionando así el ZUPT-aiding. Para detalles adicionales, véase [1].

Las ecuaciones finales empleadas por nuestro ZUPT-aided INS son

```
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
    % Actualización temporal
    $\hat{\mathbf{x}}_n = f_{\mathrm{mech}}(\hat{\mathbf{x}}_{n-1}, \mathbf{f}_n, \omega_n)$
    $\mathbf{P}_n = \mathbf{F}_n \mathbf{P}_{n-1} \mathbf{F}_n^T + \mathbf{G}\mathbf{Q}\mathbf{G}^T$
    % Actualización por medición
    if $T(\{\boldsymbol{\omega}^i, \mathbf{f}^i\}_{W_n}) < \gamma$
        $\mathbf{K}_n = \mathbf{P}_n \mathbf{H}^T (\mathbf{H}\mathbf{P}_n \mathbf{H}^T + \mathbf{R})^{-1}$
        $\delta\hat{\mathbf{x}}_n = \mathbf{K}_n \hat{\mathbf{v}}_n$
        $\mathbf{P}_n \leftarrow \mathbf{P}_n (\mathbf{I} - \mathbf{K}_n \mathbf{H})$
        % Compensación de estados interna
        $\begin{bmatrix} \hat{\mathbf{p}}_n \\ \hat{\mathbf{v}}_n \end{bmatrix} \leftarrow \begin{bmatrix} \hat{\mathbf{p}}_n \\ \hat{\mathbf{v}}_n \end{bmatrix} + \begin{bmatrix} \delta\hat{\mathbf{p}}_n \\ \delta\hat{\mathbf{v}}_n \end{bmatrix}$
        $\hat{\mathbf{R}}_n \leftarrow (\mathbf{I}_3 - \boldsymbol{\Delta}_n)\hat{\mathbf{R}}_n$
        $\delta\hat{\mathbf{x}}_n \leftarrow \mathbf{0}$
                                                                    (3.8)
```

donde $\mathbf{P}_n = \mathrm{cov}(\delta\hat{\mathbf{x}}_n)$ es la matriz de covarianza del error, $\mathbf{G}$ es la matriz de ruido del proceso, $\mathbf{Q} = \mathrm{cov}(\mathbf{u}_k)$, $\mathbf{H} = [\mathbf{0}_3\ \mathbf{I}_3\ \mathbf{0}_3]$ es la matriz de observación, $\mathbf{K}$ es la ganancia de Kalman, y

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

El algoritmo aquí explicado es de tipo complementario en lazo cerrado, donde para cada iteración n, el estado estimado $\hat{x}_n$ se corrige por una medida adicional (la ZUPT) a través de la desviación estimada $\delta\hat{\mathbf{x}}_n$. Esta es la forma habitual en que los ZUPT-aided INSs hacen el dead-reckoning.

Capítulo 4

Algoritmos de smoothing

El objetivo de este proyecto es implementar un algoritmo de smoothing específico para ZUPT-aided INSs que evite la aparición de correcciones abruptas en los estados estimados por el sistema. Que sepamos, no existen tales algoritmos y por tanto se debe diseñar un algoritmo específico para ello. Este capítulo introduce los fundamentos del smoothing y el algoritmo general de smoothing elegido como base para la implementación de un algoritmo de smoothing específico para ZUPT-aided INSs, así como la motivación para su elección con respecto a los otros algoritmos generales. En el capítulo equivalente en el anexo A se pueden encontrar el resto de algoritmos analizados. En el siguiente capítulo, el algoritmo elegido será adaptado a los aspectos específicos de un ZUPT-aided INS para obtener el algoritmo de smoothing para ZUPT-aided INS buscado.

4.1. Fundamentos de smoothing

El objetivo de un proceso de estimación smoothing (también llamado proceso de estimación no causal) es determinar el vector de estado estimado $\hat{\mathbf{x}}_{n|N},\ n < N$ tal que su varianza del error sea minimizada proporcionando información proviniente del futuro. Este es el fundamento de un *problema de smoothing.*

Hay distintas aproximaciones para resolver un problema de smoothing. El problema de smoothing de *intervalo fijo* es elmás conocido y ampliamente usado. Busca calcular $\hat{\mathbf{x}}_{n|N}$ a partir de un conjunto fijo de medidas $\{\tilde{\mathbf{y}}_0, \tilde{\mathbf{y}}_1 ... \tilde{\mathbf{y}}_N\}$ para cada $n \in \{0, 1, ..., N\}$. La aproximación de *punto fijo* mantiene n fijo mientras N va aumentando; y la aproximación de *retardo fijo* hace variar n a la vez que N, por lo que $\forall n,\ n + L = N,\ L \in \mathbb{N}$ y fijo.

En el ámbito de este proyecto, únicamente métodos pertenecientes al problema de intervalo fijo se van a analizar. Esto se debe a que se considera que la estructura de la señal disponible se corresponde con este método. Cada paso tiene una duración distinta de N_i muestras, con la ZUPT en las últimas muestras. Por tanto, si cada paso se divide en segmentos de N_i muestras, para cada segmento obtenido el problema de smoothing de intervalo fijo puede resolverse. Cómo dividir adecuadamente estos segmentos es un problema que será discutido en el siguiente capítulo. Si la información que proporciona la ZUPT al final de los pasos, que causa las correcciones abruptas, se hace ahora disponible a lo largo de cada paso calculando $\hat{\mathbf{x}}_{n|N}$ para cada $n \in \{0, 1, ..., N\}$ mediante un filtro de smoothing, se espera que las correcciones abruptas al final del paso no ocurran, alcanzando así el deseado efecto de smoothing.

4.2. La fórmula Rauch-Tung-Striebel (RTS)

Como el algoritmo de Bryson-Frazer (BF) (ver anexo A, capítulo 4.2.) , la fórmula de Rauch-Tung-Striebel (RTS) también aprovecha la estructura del modelo de estado de espacios. Si $\mathbf{x}_n$, $\hat{\mathbf{x}}_n$ y $\mathbf{r}_{n|N}$ seguían en las fórmulas BF ecuaciones recursivas que permitían el cálculo de $\hat{\mathbf{x}}_{n|N}$ a partir de ellas, parece lógico suponer que $\hat{\mathbf{x}}_{n|N}$ pueda ser directamente obtenido mediante una estructura iterativa. Esta es la idea subyacente bajo las recursiones RTS: obtener una fórmula iterativa que permita calcular $\hat{\mathbf{x}}_{n|N}$ a partir de $\hat{\mathbf{x}}_{n+1|N}$.

Del mismo modo que el algoritmo BF, el algoritmo RTS consiste en un algoritmo de smoothing en "dos pasadas". Hacia delante, se calcula $\hat{\mathbf{x}}(n|n)$ aplicando el filtro de Kalman correspondiente, sobre el que se hablara en el siguiente capítulo. Después, se hace una segunda pasada *hacia atrás*, que es donde se utiliza la fórmula RTS.

Existen distintas formas para las recursiones RTS. En las siguientes líneas puede encontrarse la así llamada *fórmula de smoothing RTS original*, la cual presenta la ventaja de implementación en comparación con los otros modelos de RTS de que sólo requiere para cada iteración la inversión de la matriz $\mathbf{P}_{n+1|n}$. Los otros métodos de RTS requieren en cada iteración la inversión de dos matrices, $\mathbf{P}_{n+1|n}$ y $\mathbf{F}_n$, lo que es computacionalmente más costoso.

Con condiciones $\forall n \in [0,\ ...,\ N]\ \exists\ \mathbf{F}_n^{-1}$, $E[\mathbf{u}_n\mathbf{n}_l^T] = 0$ and $\mathbf{P}_n > 0$, la fórmula RTS es

$$\begin{aligned}
&for\ n = N-1,\ ...,\ 0\\
&\qquad \mathbf{A}(n) = \mathbf{P}(n|n)\mathbf{F}^T\mathbf{P}^{-1}(n+1|n)\\
&\qquad \hat{\mathbf{x}}(n|N) = \hat{\mathbf{x}}(n|n) + \mathbf{A}(n)[\hat{\mathbf{x}}(n+1|N) - \hat{\mathbf{x}}(n+1|n)]\\
&\qquad \mathbf{P}(n|N) = \mathbf{P}(n|n) + \mathbf{A}(n)[\mathbf{P}(n+1|N) - \mathbf{P}(n+1|n)]\mathbf{A}^T(n)
\end{aligned} \tag{4.1}$$

con condiciones iniciales $\hat{\mathbf{x}}(N|N), \mathbf{P}(N|N)$ obtenidas del filtro Kalman hacia delante; y donde por claridad se ha escrito $\hat{\mathbf{x}}(n|n)$ en lugar de $\hat{\mathbf{x}}(n)$

Nótese que para ZUPT-aided INSs, $\hat{\mathbf{x}}(n|n-1)$ y $\mathbf{P}(n|n-1)$ se corresponden a la actualización temporal del filtro de Kalman hacia delante $\hat{\mathbf{x}}^-(n)$ y $\mathbf{P}^-(n)$, mientras que $\hat{\mathbf{x}}(n|n)$ y $\mathbf{P}(n|n-1)$ se corresponden a la actualización de velocidad cero que, en el caso de que un estado de velocidad cero (zero velocity, ZV) no se detecte, son idénticas a la actualización de tiempo.

4.3. Comparación de algoritmos y elección

En la versión equivalente de este capítulo en el apéndice A, los algoritmos de smoothing más relevantes son analizados para decidir cuál se va a usar como base para la implementación de un smoothed ZUPT-aided INS. Ahora se requiere fijar un criterio para la elección de un algoritmo de smoothing.

Además, no existe trabajo previo documentado para el filtrado de smoothing sobre sistemas ZUPT-aided INS, lo cual implica la carencia de fuentes sobre las que basar la elección del algoritmo de smoothing para el sistema. Debido a que la implementación de un algoritmo de smoothing paso a paso para ZUPT-aided

INSs conlleva la consideración de ciertos aspectos específicos de implementación, tal y como se mostrará en el próximo capítulo, en lo subsiguiente se sigue esencialmente un criterio de elección: la simplicidad del algoritmo. Esto permite concentrarse en los problemas específicos de implementación para un filtro de smoothing para ZUPT-aided INS.

Como el sistema va a ejecutarse en Matlab, por ahora los requisitos de memoria no se consideran como restricción. Del mismo modo, aspectos relacionados con la implementación en tiempo real no son completamente consideradas debido a que el sistema se ejecuta en Matlab cuando todas las medidas del sensor se han tomado y por tanto están disponibles. Sin embargo, se debe remarcar que en esta tesis se está implementando un algoritmo de smoothing *paso a paso* para ZUPT-aided INS, lo cual hará que el algoritmo propuesto en un futuro cercano se implemente para una aplicación en casi tiempo real. Por tanto, en lo que sigue este aspecto no es completamente despreciado.

Considerando todos estos aspectos, puede afrontarse la elección del algoritmo de smoothing. Debido a los problemas de implementación que presentan las fórmulas de dos filtros, éstas son descartadas. Chequeando los algoritmos restantes, el de BF y el de RTS, es evidente que el algoritmo RTS presenta una implementación más sencilla. Adicionalmente, la recursión BF requiere la inversión de dos matrices, $\mathbf{P}_n$ y $\mathbf{F}_n$, mientras que la recursión RTS requiere únicamente la inversión de la matriz $\mathbf{P}_n$. Es más, el algoritmo RTS no requiere el vector de innovación $\mathbf{e}_n$, mientras que el BF lo hace. Este vector no está directamente disponible en nuestra versión del sistema, requiriendo de su cálculo.

Como conclusión, la implementación del método RTS es directa y presenta ciertas ventajas con respeto a la del BF y es por tanto el algoritmo RTS el elegido para la implementación de un filtro de smoothing para ZUPT-aided INSs, que es el objetivo de esta tesis. Aunque no es un requisito en esta tesis, puede apreciarse en la ecuación 4.1 que las matrices $\mathbf{P}(n|n), \mathbf{P}(n|n-1)$ and vectors $\hat{\mathbf{x}}(n|n), \hat{\mathbf{x}}(n|n-1)$ deben ser almacenadas en memoria en la pasada hacia delante del filtro de Kalman ya que el algoritmo de smoothing requiere de ellas en la pasada hacia atrás, incrementando notablemente los requisitos de memoria. En caso de que la memoria sea una restricción, como la covarianza va incrementando prácticamente de forma lineal a lo largo del paso, unos pocos valores de covarianza $\mathbf{P}$ pueden ser almacenados y usados para interpolar el resto. Además, la complejidad computacional es la misma en la pasada hacia atrás que en la pasada hacia delante, por lo que el número de operaciones se duplica.

Capítulo 5

Smoothed ZUPT-aided INS

Hasta ahora se ha explicado por un lado el ZUPT-aided INS sobre el que trabajamos como modelo típico de ZUPT-aided INS; y por el otro se han introducido los algoritmos generales de smoothing. De su combinación se obtendrá el algoritmo de smoothing para ZUPT-aided INS que buscamos. Sin embargo, esta combinación requiere considerar ciertos aspectos que necesitan ser resueltos. Este capítulo trata estos aspectos: dónde aparecen y cuál es la solución propuesta. El primer punto de este capítulo explica por qué se requiere y cómo se hace una implementación en lazo abierto del usual filtro de Kalman en lazo cerrado para ZUPT-aided INSs. La segunda sección del capítulo explica la segmentación de datos propuesta para alcanzar una situación en la que un algoritmo de smoothing de intervalo fijo pueda ser aplicado. Como los pasos no están uniformemente espaciados, se requiere y por tanto introduce una regla de retardo variable.

5.1. Implementación open-loop

La identificación de los términos de la fórmula RTS (4.1) con los de la recursión de Kalman usual para ZUPT-aided INSs que se ha expuesto (ver eq.(3.8)) no es una identificación directa ya que los estados estimados $\hat{\mathbf{x}}_n$ no pueden ser relacionados directamente con las componentes de la fórmula RTS. Esto es debido a que los ZUPT-aided INS consisten típicamente en filtros en lazo cerrado. Esto significa que para cada iteración el filtro proporciona una estimación del estado completo $\hat{\mathbf{x}}_n$ a través de lo que hemos denominado *compensación interna de estados.* Smoothing directo sobre el vector $\hat{\mathbf{x}}_n$ no es posible ya que la matriz de covarianza disponible es la matriz de covarianza *del error* $\mathbf{P}$, no la matriz de covarianza de estados.

Por tanto, para una identificación directa con los términos en la fórmula RTS abrir el filtro de Kalman es necesario permitiendo así propagar por un lado una estimación del estado $\hat{\mathbf{x}}^-$ empleando las medidas tomadas por la IMU y usando las ecuaciones de mecanización (3.1)-(3.5) . Por otro lado se propaga el error estimado a través del filtrado de Kalman $\delta\hat{\mathbf{x}}$. Es éste vector $\delta\hat{\mathbf{x}}$ el que contiene las correcciones a ser alisadas. Como la matriz de covarianza del error $\mathbf{P}$ está disponible, el smoothing sí puede hacerse sobre $\delta\hat{\mathbf{x}}$. De este modo, las ecuaciones del filtro de Kalman se modifican a las incluidas en las ecuaciones 5.3, donde se muestra una implementación en filtro abierto del ZUPT-aided INS.

La actualización temporal sigue la ecuación estándar de propagación de Kalman

$$\delta\hat{\mathbf{x}}_n^- = \mathbf{F}_n \delta\hat{\mathbf{x}}_{n-1} \tag{5.1}$$

Del mismo modo, el estado estimado $\hat{\mathbf{x}}_n^-$ se propaga durante la actualización temporal siguiendo las mismas ecuaciones de mecanización usadas hasta ahora. Sin embargo, debido a la naturaleza en lazo abierto de la implementación, ahora no se proporciona realimentación a $\hat{\mathbf{x}}_n^-$ durante la actualización por medida.

En lugar de proporcionar la realimentación sobre $\hat{\mathbf{x}}_n^-$, durante la actualización por medida se corrige el error estimado $\delta\hat{\mathbf{x}}_n^-$. Durante una ZUPT el *error* es la diferencia entre la velocidad estimada y cero, siendo esta la corrección aplicada en la versión en lazo cerrado del filtro. Es esta corrección la cantidad que el error *estimado* $\delta\hat{\mathbf{x}}_n^-$ debería disminuir durante la ZUPT. Así, la ecuación de actualización por medida para la versión en lazo abierto del filtro es

$$\delta\hat{\mathbf{x}}_n = \delta\hat{\mathbf{x}}_n^- + \mathbf{K}_n(-\mathbf{v}) \tag{5.2}$$

Por tanto, se propagan un estado estimado $\hat{\mathbf{x}}_n^-$ sin ser corregido y una estimación del error $\delta\hat{\mathbf{x}}_n$. Tras la segmentación de datos aplicando el criterio explicado en la siguiente sección, el filtro de smoothing puede aplicarse al segmento de estimaciones de error de estado $\delta\hat{\mathbf{x}}$. Este segmento es el que contiene las drásticas correcciones a ser eliminadas y que puede ser directamente identificado con los términos en la fórmula RTS (4.1). Tras aplicar el smoothing sobre el error estimado $\delta\hat{\mathbf{x}}$, no hay nada que impida aplicar la compensación interna de estados para cada muestra del segmento. De este modo, el estado estimado final $\hat{\mathbf{x}}_n$ se calcula para cada muestra del segmento. Sin embargo, antes de dar las ecuaciones completas para un smoothed ZUPT-aided INS, en la siguiente sección se trata con el tema de la segmentación de datos.

```
**Inicialización:** $\hat{\mathbf{x}}_0 = E[\mathbf{x}_0], \delta\hat{\mathbf{x}}_0 = 0, \mathbf{P}_0 = var(\mathbf{x}_0)$
 **Loop:** $n = s_{\text{start}}$ **to** $s_{\text{end}}$
    % Actualización temporal
    $\hat{\mathbf{x}}_n = f_{\text{mech}}(\hat{\mathbf{x}}_{n-1}, \mathbf{u}_n)$
    $\delta\hat{\mathbf{x}}_n^- = \mathbf{F}_n \delta\hat{\mathbf{x}}_{n-1}$
    $\mathbf{P}_n^- = \mathbf{F}_n \mathbf{P}_{n-1} \mathbf{F}_n^T + \mathbf{G}\mathbf{Q}\mathbf{G}^T$
    % Actualización por medición
    **if** $T(\{\boldsymbol{\omega}^i, \mathbf{f}^i\}_{W_n}) < \gamma$
        $\mathbf{K}_n = \mathbf{P}_n^- \mathbf{H}^T (\mathbf{H}\mathbf{P}_n^- \mathbf{H}^T + \mathbf{R})^{-1}$          (5.3)
        $\delta\hat{\mathbf{x}}_n = \delta\hat{\mathbf{x}}_n^- - \mathbf{K}_n(\delta\hat{\mathbf{v}}_n - \hat{\mathbf{v}}_n)$
        $\mathbf{P}_n = \mathbf{P}_n^-(\mathbf{I} - \mathbf{K}_n \mathbf{H})$
 % Compensación de estados interna
 **Loop:** $n = s_{\text{start}}$ **to** $s_{\text{end}}$
    $\begin{bmatrix} \hat{\mathbf{p}}_n \\ \hat{\mathbf{v}}_n \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{p}}_n^- \\ \hat{\mathbf{v}}_n^- \end{bmatrix} + \begin{bmatrix} \delta\hat{\mathbf{p}}_n \\ \delta\hat{\mathbf{v}}_n \end{bmatrix}$
    $\hat{\mathbf{R}}_n = (\mathbf{I}_3 - \boldsymbol{\Delta}_n)(\hat{\mathbf{R}}_n)^-$
```

5.2. Segmentación de los datos

En el capítulo previo mostramos que se buscaba resolver un problema de smoothing de *intervalo fijo*. Este problema toma un segmento fijo de medidas $\{\tilde{\mathbf{y}}_0, \tilde{\mathbf{y}}_1, ..., \tilde{\mathbf{y}}_N\}$ y calcula $\hat{\mathbf{x}}_{n|N}$ para cada muestra en el segmento. Consideramos que esta manera de enfocar el problema de smoothing se corresponde muy bien con el sistema disponible, ya que cada segmento de datos puede asociarse con un paso de la persona y la información que proporciona la ZUPT al final del paso, que causa las discontinuidades, puede hacerse disponible a lo largo de todo el paso. Adicionalmente, dividir los datos en segmentos de corta duración como es un paso permitirá la implementación de un smoothed ZUPT-aided INS que funcione en casi tiempo real.

En consecuencia, buscando alcanzar las condiciones de problema de smoothing de intervalo fijo, los datos deben ser segmentados. Para ello, ciertos problemas deben ser solucionados. Es sencillo entender que dos pasos distintos no tienen la misma longitud y por tanto no duran la misma cantidad de muestras. Por tanto, para una correcta segmentación de los datos se requiere de una regla de segmentación que pueda considerar la diferente duración de cada paso. Los siguientes párrafos motivan la decidida para esta tesis.

Un pensamiento intuitivo es aprovecharse de los intervalos de ZV y dividir el paso por identificación directa con éstos, ya que hay una fase estacionaria por paso. Sin embargo, esto no es posible, al menos no directamente. La detección de estados de ZV se hace aplicando umbrales γ a la magnitud medida de aceleración, a la magnitud medida por el giroscopio y/o a la varianza de la aceleración local. Cuando se está entre todos estos umbrales (AND lógica), un estado de ZV es decidido. Incluso aunque se aplican métodos para evitar la detección errónea de un estado de ZV, como el promediado de muestras adyacentes, experimentalmente se muestra que durante la fase estacionaria de un paso se deciden fases de no-ZV cuando no deberían. Esto se ilustra en el vector de ZUPT que se puede ver en Fig. 5.1, donde en torno al instante 1.4 seg. hay una decisión errónea de no-ZUPT. Cuanto más alta sea la velocidad de la persona, más probable es que esto ocurra. Por tanto, emplear directamente la decisión de ZV como método de segmentación de datos no es posible, ya que un paso no se corresponde a la secuencia *"no ZV detectada - ZV detectado = 1 paso"*.

Pese a todo, ciertos aspectos de la idea de detección de ZV se aprovechan para obtener la regla de segmentación empleada en esta tesis, donde umbrales de covarianza y tiempo se emplean. Para motivarla, en primer lugar analicemos la evolución de la covarianza del error de velocidad a lo largo de un paso. En la Fig. 5.1 se muestra como la covarianza del error en la velocidad disminuye drásticamente tan pronto como una ZUPT se detecta. Cuando el pie alcanza un estado estacionario, la corrección tomando medidas externas se hace y por tanto las dependencias entre las distintas variables disminuyen y se mantienen mínimas mientras la ZUPT ocurre. Tan pronto como las ZUPTs acaban, la covarianza va incrementándose monótonamente de nuevo.

De la evolución del segmento analizado de covarianza del error en la velocidad del previo párrafo, se decide fijar un umbral de covarianza γ_s cuyo cruce implicará decidir segmentación. ¿Dónde y cómo fijar la evaluación de este umbral de covarianza γ_s?

Se consideran tres puntos distintos para la segmentación, marcados en la Fig.5.2.

- A, punto donde la covarianza del error en la velocidad está disminuyendo drásticamente debido a la ZUPT.

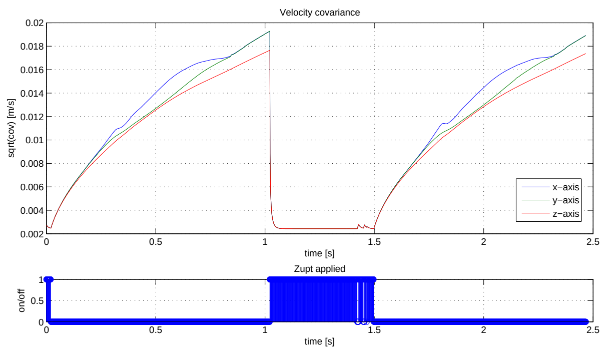


Figura 5.1: Covarianza del error en la velocidad para dos pasos

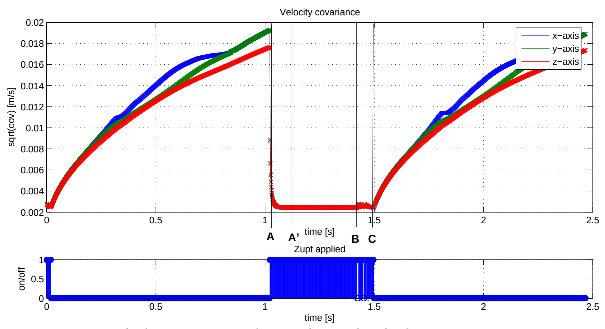


Figura 5.2: Covarianza del error en la velocidad durante una ZUPT. Cada muestra aparece marcada.

- B, punto donde la primera discontinuidad en el vector de ZUPT se detecta en la fase estacionaria del paso.
- C, punto donde la fase estacionaria del paso termina.

Desafortunadamente, ninguno de estos puntos presenta las características adecuadas para ser el punto donde segmentar:

- En el punto A las mayores correcciones se están haciendo sobre los valores de error de la covarianza y los estados. Como el sistema todavía no ha convergido, un smoothing adecuado no es posible ya que la información proporcionada por la ZUPT no está todavía completamente disponible al final del segmento obtenido cortando en A.
- La segmentación en B, donde la primera secuencia de ZV detectadas acaba, implica el mismo problema que en A si la covarianza todavía no ha convergido. Ademas, caracterizar el sistema es dificil ya que este punto está situado aleatoriamente en el segmento de la ZUPT (de hecho, este evento suele ocurrir hacia el final de la fase estacionaria)
- Debido a la aleatoriedad de la aparicion de las discontinuidades en el vector de ZUPT, segmentar en C es imposible sin admitir cierto grado de no causalidad.

A pesar de esto, de la situación A se obtiene una regla de segmentación válida si tras cruzar el umbral, se crea el segmento un tiempo constante más tarde, en A'. En A la covarianza del error en la velocidad todavía no ha convergido. Esto significa que la información de la ZUPT no está completamente disponible. Si la segmentación se decide un tiempo constante más tarde, en A', la covarianza del error en la velocidad ya ha convergido casi completamente en un mínimo y la información proporcionada por la ZUPT está completamente disponible permitiendo así un correcto smoothing. Para determinar la distancia entre A' y A, un umbral de tiempo τ_s debe ser decidido.

Ciertos problemas pueden aparecer con el umbral de covarianza γ_s. El cruce a través del umbral γ_s ha de ser detectado cuando la covarianza decrece. De otro modo, cuando la covarianza está aumentando, podría tomarse una decisión de segmentación errónea.. Por otro lado, durante la fase estacionaria de un paso, puede tomarse una decisión errónea de no-ZUPT y la covarianza ir aumentando su valor. Estos segmentos erróneamente decididos son cortos y el valor de la covarianza no incrementa demasiado. Pese a todo, el valor de γ_s debe ser lo suficientemente grande como para evitar que este umbral sea atravesado cuando la ZUPT vuelve a detectarse. Problemas de detección adicionales no necesitan ser considerados, ya que la covarianza del error en la velocidad decrece monótonamente durante una ZUPT y crece monótonamente en caso contrario.

En consecuencia, la regla de segmentación propuesta se divido en dos pasos: primero, la detección del instante en que se cruza un umbral de covarianza del error en la velocidad γ_s. τ_s muestras (segundos) más tarde, se hace la segmentación y un paso es correctamente obtenido para su posterior smoothing. En la Fig. 5.3. se muestra un resumen de la regla de segmentación, donde las distintas partes del gráfico se han exagerado para facilitar la explicación de la regla de segmentación.

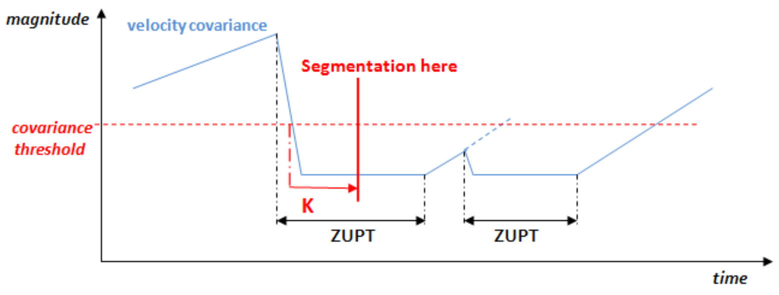


Figura 5.3: Resumen de la regla de segmentación. La covarianza del error en la velocidad aumenta a lo largo del paso y disminuye drásticamente cuando una ZUPT se decide. Durante la fase estacionaria de un paso, puede tomarse una decisión errónea de no-ZUPT. Estos segmentos erróneamente decididos son cortos, y la covarianza aumenta pero no traspasa el umbral elegido de covarianza.

Por último, la evaluación de esta regla de segmentación debe hacerse en algún punto durante la iteración del filtro de Kalman. Como la regla de segmentación requiere el conocimiento del valor de covarianza actual, hemos decidido evaluar esta regla al final de la iteración, tal y como se muestra en Alg. 1 al final de este capítulo.

5.2.1. Elección de los valores de los umbrales

Fijar valores para τ_s y γ_s es necesario. Para ello, considérense los siguientes valores típicos del sistema para trayectos de prueba grabados para una persona caminando a 3.5 km/h, con frecuencia del sistema de $F_s = 820$ Hz. Bajo estas condiciones,

- la longitud media de la fase no estacionaria del paso es de ∼800 muestras (0.976 sec.).

- la longitud media de la fase estacionaria sin detecciones erróneas de no-ZV es de ~400 muestras (0.488 sec.).
- la media del máximo de la suma de la covarianza del error en la velocidad (el valor justo antes de que la ZUPT empiece disminuyendo el valor de la covarianza) es $9{,}58 \cdot 10^{-4}$.
- la media del máximo de la suma de las componentes de la covarianza del error en la velocidad en los segmentos cortos erróneamente decididos de no-ZV durante la fase estacionaria del paso es $2{,}23 \cdot 10^{-5}$.
- el valor pico de la suma de las componentes de la covarianza del error en la velocidad en los segmentos cortos erróneamente decididos de no-ZV durante la fase estacionaria del paso es $4{,}16 \cdot 10^{-5}$.
- el valor mínimo de la suma de las componentes de la covarianza del error en la velocidad es $1{,}78 \cdot 10^{-5}$.

En primer lugar, un valor para γ_s se decide. Ha de ser lo suficientemente pequeño como para que primero la covarianza haya convergido prácticamente cuando se detecte (evitando así el problema discutido para el punto A), pero lo suficientemente grande como para que segundo no sea afectada por el aumento del valor de la covarianza cuando un estado de no-ZV se detecta durante la fase estacionaria del paso, ver Fig. 5.3. Como el umbral de tiempo ha de fijarse para permitir la convergencia una vez el umbral de covarianza se ha cruzado, el factor crítico para fijar γ_s es el segundo (ya que si no, se segmentarían dos pasos donde sólo hay uno). Se elige un valor de $\gamma_s = 1{,}50 \cdot 10^{-4}$, que está varias veces por encima del valor medio y del valor pico detectado en las simulaciones para el máximo de la suma de las componentes de la covarianza del error en la velocidad en los segmentos cortos erróneamente decidios de no-ZV durante la fase estacionaria del paso. El valor elegido es *suma* de componentes para compensar los distintos efectos que pudieran ocurrir en los distintos ejes.

La constante de tiempo τ_s se fija en 30 muestras (0.037 seg.), que experimentalmente muestra ser suficiente para permitir la convergencia del sistema y por tanto un smoothing posterior correcto. Además, el valor decidido de τ_s proporciona suficiente margen como para permitir trabajar con personas que caminen más rápido, lo que implica una duración más corta de la fase estacionaria del paso. Nótese que la duración media de la fase estacionaria para una persona caminando a 3.5 km/h es 0.488 seg.

Para comprobar si los valores elegidos son adecuados, se grabaron nuevos sets de datos con una persona caminando a una velocidad media de 6.5 km/h. Desgraciadamente, el sistema del que disponemos no es wirelss y grabar datos a velocidades más altas es complejo. Para los datasets obtenidos, los valores típicos fueron

- la longitud media de la fase no estacionaria del paso es de ~600 muestras (0.731 sec.).
- la longitud media de la fase estacionaria sin detecciones erróneas de no-ZV es de ~150 muestras (0.182 sec.), con alta desviación típica.
- la media del máximo de la suma de la covarianza del error en la velocidad (el valor justo antes de que la ZUPT empiece disminuyendo el valor de la covarianza) es $6{,}84 \cdot 10^{-4}$.
- la media del máximo de la suma de las componentes de la covarianza del error en la velocidad en los segmentos cortos erróneamente decididos de no-ZV durante la fase estacionaria del paso es $3{,}45 \cdot 10^{-5}$
- el valor pico de la suma de las componentes de la covarianza del error en la velocidad en los segmentos cortos erróneamente decididos de no-ZV durante la fase estacionaria del paso es $0{,}966 \cdot 10^{-4}$.

- el valor mínimo de la suma de las componentes de la covarianza del error en la velocidad es $1{,}78 \cdot 10^{-5}$.

De estos valores se puede concluir que los valores elegidos para τ_s y γ_s tienen suficiente margen para trabajar a mayores velocidades: τ_s es cinco veces mayor que la media del máximo de la suma de las componentes de la covarianza del error en los segmentos cortos erróneamente decididos de no-ZV durante la fase estacionaria del paso, y cuatro veces más pequeña que la media del máximo de la suma de componentes de la covarianza del error. Finalmente, aunque estos valores tienen suficiente margen, para mayores velocidades de la persona estos valores deben reconsiderarse. De acuerdo con nuestros tests, el valor de τ_s puede disminuirse sin que las estimaciones se vean afectadas. Además, modificando los umbrales para la detección de estados de ZV, la detección de estos estados puede mejorarse. Por tanto, para mayores velocidades de la persona, un proceso de tuning para los umbrales de la regla de smoothing y de detección de fases de ZV ha de ser considerado.

5.3. Implementación del smoothed ZUPT-aided INS

Este capítulo ha mostrado y resuelto los distintos aspectos a considerar para la implementación de un smoothed ZUPT-aided INS. Así, añadiendo las consideraciones llevadas a cabo en la anterior sección y las fórmulas de smoothing 4.1 sobre la versión en lazo abierto del usual ZUPT-aided INS

Considerando estos aspectos, sobre la versión en lazo abierto del filtro de Kalman (ecuaciones (5.3)) el algoritmo de smoothing que proponemos en este proyecto para ZUPT-aided INSs se proporciona en Alg. 1. Este algoritmo se puede considerar como un algoritmo en 3 pasadas. En la primera, se ejecuta el ZUPT-aided INS en lazo abierto. La ejecución hacia delante se suspende temporalmente por la segmentación de los datos, ejecutando una segunda pasada hacia atrás que añade el smoothing. Finalmente, el algoritmo hace una tercera pasada hacia delante en la que el error alisado se emplea para corregir los estados de navegación. Después, se vuelve a iniciar la primera pasada para el siguiente segmento de datos. La tercera pasada cierra el bucle permitiendo ver el algoritmo propuesto como un filtro de lazo combinado abierto-cerrado.

A efectos de análisis, también se considera un smoothing off-line de los datos, donde la información de todas las ZUPT del futuro se encuentra disponible para el smoothing. Ello implica no realizar segmentación de los datos, por lo que de las ecuaciones en Alg. 1 no se ejecuta la parte correspondiente a la regla de segmentación.

Por último, debido a las ventajas de estabilidad que presentan los filtros lazo cerrado, también se ha implementado una versión en lazo cerrado del algoritmo de smoothing para ZUPT-aided INS, la cual se resume en el Alg. 2, que es una adaptación del algoritmo propuesto. De nuevo consiste en un algoritmo en tres pasadas. El filtro de Kalman en lazo cerrado se ejecuta normalmente, y tras la segmentación, se aplica el smoothing sobre el vector de error $\delta\hat{\mathbf{x}}$. Después se realiza el compensado de estados interno. Sin embargo, existe un problema en la configuración del filtro que no hemos sido capaces de localizar todavía, pero los resultados obtenidos son razonablemente buenos. Actualmente seguimos trabajando sobre él.

Algorithm 1 Pseudo código para el algoritmo en 3 pasadas propuesto

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

Loop while $s_{\text{start}} < s_{\text{end}}$
  %Kalman filter hacia delante
  Loop: $n = s_{\text{start}}$ to $s_{\text{end}}$
    % Actualización temporal
    $\hat{\mathbf{x}}_n = f_{\text{mech}}(\hat{\mathbf{x}}_{n-1}, \mathbf{u}_n)$
    $\delta\hat{\mathbf{x}}_n^- = \mathbf{F}_n \delta\hat{\mathbf{x}}_{n-1}$
    $\mathbf{P}_n^- = \mathbf{F}_n \mathbf{P}_{n-1} \mathbf{F}_n^T + \mathbf{GQG}^T$
    % Actualización por medición
    if $T(\{\boldsymbol{\omega}^i, \mathbf{f}^i\}_{W_n}) < \gamma$
      $\mathbf{K}_n = \mathbf{P}_n^- \mathbf{H}^T (\mathbf{HP}_n^- \mathbf{H}^T + \mathbf{R})^{-1}$
      $\delta\hat{\mathbf{x}}_n = \delta\hat{\mathbf{x}}_n^- - \mathbf{K}_n(\delta\hat{\mathbf{v}}_n - \hat{\mathbf{v}}_n)$
      $\mathbf{P}_n = \mathbf{P}_n^-(\mathbf{I} - \mathbf{K}_n\mathbf{H})$
    % Eval. regla de segmentación
    if $c > 0$
      $c = c + T_s$
    if $\|\text{diag}(\mathbf{P}_{n-1}^v)\| > \gamma_s \ \wedge \ \|\text{diag}(\mathbf{P}_n^{vel})\| \leq \gamma_s \ \wedge c = 0$
      $c = T_s$
    if $c > \tau_s$
      $s_{\text{end}} \leftarrow n$
      break loop
  % Smoothing
  Loop: $n = s_{\text{end}} - 1$ to $s_{\text{start}}$
        $\mathbf{A}_n = \mathbf{P}_{n|n} \mathbf{F}^T \mathbf{P}_{n+1|n}^{-1}$
    $\delta\hat{\mathbf{x}}_{n|s_{\text{end}}} = \delta\hat{\mathbf{x}}_{n|n} + \mathbf{A}_n(\delta\hat{\mathbf{x}}_{n+1|s_{\text{end}}} - \delta\hat{\mathbf{x}}_{n+1|n})$
     $\mathbf{P}_{n|s_{\text{end}}} = \mathbf{P}_{n|n} + \mathbf{A}_n(\mathbf{P}_{n+1|s_{\text{end}}} - \mathbf{P}_{n+1|n})\mathbf{A}_n^T$
  % Compensación de estados interna
  Loop: $n = s_{\text{start}}$ to $s_{\text{end}}$
    $\begin{bmatrix} \hat{\mathbf{p}}_n \\ \hat{\mathbf{v}}_n \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{p}}_n^- \\ \hat{\mathbf{v}}_n^- \end{bmatrix} + \begin{bmatrix} \delta\hat{\mathbf{p}}_n \\ \delta\hat{\mathbf{v}}_n \end{bmatrix}$
    $\hat{\mathbf{R}}_n = (\mathbf{I}_3 - \boldsymbol{\Delta}_n)(\hat{\mathbf{R}}_n)^-$
    $\delta\hat{\mathbf{x}}_n \leftarrow \mathbf{0}$
  $s_{\text{start}} = s_{\text{end}} + 1$, $s_{\text{end}} =$ "end of data", $c = 0$
```

Algorithm 2 Pseudo código para el algoritmo en 3 pasadas: Lazo cerrado

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

Loop while $s_{\text{start}} < s_{\text{end}}$
    %Kalman filter hacia delante
    Loop: $n = s_{\text{start}}$ to $s_{\text{end}}$
        % Actualización temporal
        $\hat{\mathbf{x}}_n = f_{\text{mech}}(\hat{\mathbf{x}}_{n-1}, \mathbf{u}_n)$
        $\delta\hat{\mathbf{x}}_n = 0$
        $\mathbf{P}_n^- = \mathbf{F}_n\mathbf{P}_{n-1}\mathbf{F}_n^T + \mathbf{GQG}^T$
        % Actualización por medición
        if $T(\{\boldsymbol{\omega}^i, \mathbf{f}^i\}_{W_n}) < \gamma$
            $\mathbf{K}_n = \mathbf{P}_n^-\mathbf{H}^T(\mathbf{HP}_n^-\mathbf{H}^T + \mathbf{R})^{-1}$
            $\delta\hat{\mathbf{x}}_n = -\mathbf{K}_n(\delta\hat{\mathbf{v}}_n - \hat{\mathbf{v}}_n)$
            $\mathbf{P}_n = \mathbf{P}_n^-(\mathbf{I} - \mathbf{K}_n\mathbf{H})$
            % Compensación de estados interna
            $\begin{bmatrix} \hat{\mathbf{p}}_n \\ \hat{\mathbf{v}}_n \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{p}}_n^- \\ \hat{\mathbf{v}}_n^- \end{bmatrix} + \begin{bmatrix} \delta\hat{\mathbf{p}}_n \\ \delta\hat{\mathbf{v}}_n \end{bmatrix}$
            $\hat{\mathbf{R}}_n = (\mathbf{I}_3 - \boldsymbol{\Delta}_n)(\hat{\mathbf{R}}_n)^-$
        % Eval. regla de segmentación
        if $c > 0$
            $c = c + T_s$
        if $\|\text{diag}(\mathbf{P}_{n-1}^v)\| > \gamma_s \ \wedge \ \|\text{diag}(\mathbf{P}_n^{vel})\| \leq \gamma_s \ \wedge c = 0$
            $c = T_s$
        if $c > \tau_s$
            $s_{\text{end}} \leftarrow n$
            break loop
    % Smoothing
    Loop: $n = s_{\text{end}} - 1$ to $s_{\text{start}}$
        $\mathbf{A}_n = \mathbf{P}_{n|n}\mathbf{F}^T\mathbf{P}_{n+1|n}^{-1}$
        $\delta\hat{\mathbf{x}}_{n|s_{\text{end}}} = \delta\hat{\mathbf{x}}_{n|n} + \mathbf{A}_n(\delta\hat{\mathbf{x}}_{n+1|s_{\text{end}}} - \delta\hat{\mathbf{x}}_{n+1|n})$
        $\mathbf{P}_{n|s_{\text{end}}} = \mathbf{P}_{n|n} + \mathbf{A}_n(\mathbf{P}_{n+1|s_{\text{end}}} - \mathbf{P}_{n+1|n})\mathbf{A}_n^T$
    % Compensación de estados interna
    Loop: $n = s_{\text{start}}$ to $s_{\text{end}}$
        $\begin{bmatrix} \hat{\mathbf{p}}_n \\ \hat{\mathbf{v}}_n \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{p}}_n^- \\ \hat{\mathbf{v}}_n^- \end{bmatrix} + \begin{bmatrix} \delta\hat{\mathbf{p}}_{n|s_{\text{end}}} \\ \delta\hat{\mathbf{v}}_{n|s_{\text{end}}} \end{bmatrix}$
        $\hat{\mathbf{R}}_n = (\mathbf{I}_3 - \boldsymbol{\Delta}_n)(\hat{\mathbf{R}}_n)^-$
        $\delta\hat{\mathbf{x}}_n \leftarrow \mathbf{0}$
    $s_{\text{start}} = s_{\text{end}} + 1$, $s_{\text{end}}$ = “end of data”, $c = 0$
```

Capítulo 6

Evaluación y análisis de la implementación

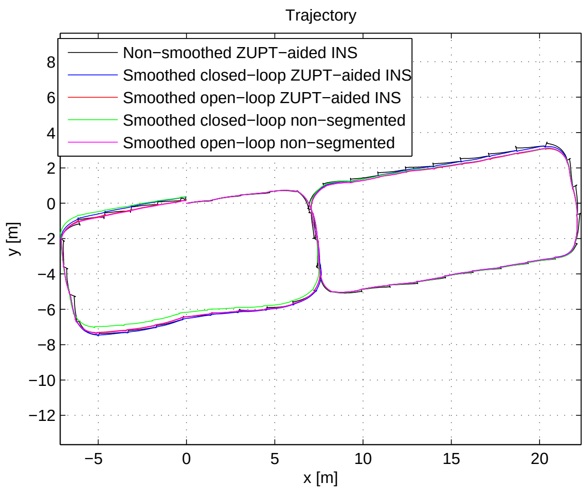
Considerando los aspectos específicos que presentan los ZUPT-aided INSs, hemos propuesto un algoritmo de smoothing específico para este tipo de INSs, consiste en la segmentación de los datos y el aplicado de un algoritmo de smoothing RTS sobre el segmento obtenido. Este capítulo analiza los cambios que las distintas implementaciones de este algoritmo producen en el comportamiento del sistema con respecto al funcionamiento del sistema original. De este modo, la primera sección introduce las distintas implementaciones disponibles del sistema. Después, se analiza el comportamiento para cada implementación del sistema sobre los estados estimados y la covarianza del error. Finalmente se muestran las conclusiones derivadas del análisis.

6.1. Preliminares

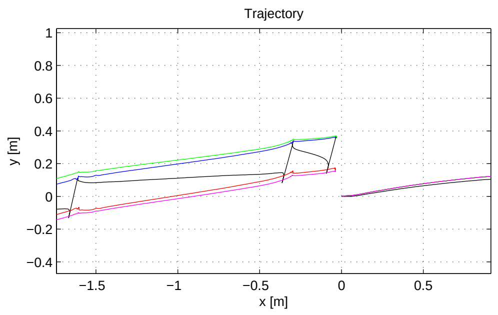
A partir de la implementación de un ZUPT-aided INS estándar, se ha implementado un algoritmo de smoothing para ZUPT-aided INS ejecutando el filtro de Kalman en que habitualmente consisten en lazo abierto. A partir de ella, se han desarrollado distintas implementaciones basadas en el algoritmo de smoothing que hemos propuesto en el capítulo anterior, tal y como se indicó en la sección 5.3. Este capítulo evalúa el efecto del smoothing en las implementaciones programadas, analizando la mejora respecto a la implementación original sin smoothing así como las diferencias entre las distintas implementaciones. A modo recopilatorio, éstas son

- El ZUPT-aided INS original (ver ecuaciones 3.8).
- Smoothed ZUPT-aided INS en lazo cerrado (ver Alg. 2).
- Smoothed ZUPT-aided INSen lazo abierto (ver Alg. 1).
- Smoothed ZUPT-aided INS no segmentado (no paso a paso) en lazo cerrado (ver Alg. 2, sin regla de segmentación).
- Smoothed ZUPT-aided INS no segmentado (no paso a paso) en lazo abierto (ver Alg. 1, sin regla de segmentación).

La evaluación llevada a cabo en este capítulo se centra en el análisis y comparación con el ZUPT-aided INS original para analizar cualitativa y cuantitati-



(a) Trayectoria completa estimada ejes xy.



(b) Vista detallada del principio y el final de la trayectoria xy estimada

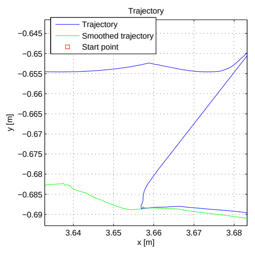
Figura 6.1: Trayectorias estimadas xy para distintas implementaciones

vamente (donde sea posible) los cambios que ocurren en el funcionamiento del sistema. Después, se comparan las distintas comparaciones de smoothing. En particular, las dos últimas implementaciones no segmentan los datos, si no que aplican la fórmula RTS al vector de error estimado $\delta\hat{\mathbf{x}}$ una vez que todo el filtro de Kalman ha procesado todo el conjunto de los datos. Estos métodos *no segmentados* se corresponden con un procesado off-line de los datos donde toda la información del futuro se encuentra disponible. Se busca su comparación con la implementación paso a paso propuesta para analizar la diferencia que aparecería en el smoothing si toda la información del futuro se encuentra disponible en comparación con que sólo se encuentre disponible la información del paso actual. Si las diferencias son asumibles, la aproximación seguida segmentando paso a paso será válida y se podrá implementar el sistema para aplicaciones en casi tiempo real.

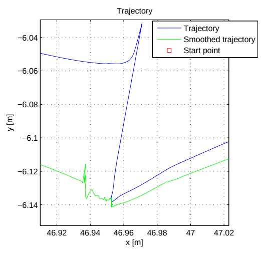
En la siguiente sección los efectos sobre los estados estimados para las distintas implementaciones son evaluados, mientras que la sección subsecuente se evalúan los cambios producidos en la covarianza. Todo esto proporciona una visión de los cambios y mejoras que produce en el sistema el algoritmo de smoothing propuesto.

6.2. Análisis de los estados estimados

La Fig. 6.1 muestra los efectos de las distintas implementaciones del algoritmo de smoothing planteado en comparación con la trayectoria estimada por el ZUPT-aided INS sin smoothing. Nótese cómo se consigue el efecto de smoothing deseado. Pese a ello, ciertos aspectos deben ser analizados. Las siguientes subsecciones consisten en un análisis cualitativo del efecto del smoothing en las estimaciones de posición, velocidad y orientación; así como una comparación de las distintas implementaciones. Adicionalmente, se cuantifica la mejora alcanzada para las discontinuidades.



(a) Implementación segmentada en lazo cerrado. 2do paso.



(b) Implementación segmentada en lazo cerrado. 34to paso.

Figura 6.2: Efecto del tiempo sobre la implementación segmentada en lazo cerrado del smoother

6.2.1. Análisis de la estimación de las posiciones

De la Fig. 6.1a se pueden analizar algunos aspectos de la trayectoria estimada. En primer lugar, muestra que la implementación smoothed sin segmentar en lazo cerrado comete un gran error que invalida la implementación. Su origen lo atribuimos a ciertos errores en su implementación, que se han de localizar. Por tanto, se requiere análisis adicional para este sistema, y la implementación sin segmentar en lazo cerrado queda invalidada para otras aplicaciones. Sin embargo, la implementación smoothed segmentada en lazo cerrado parece proveer una buena estimación. Admitiendo que cierto error se comete en la implementación, este error no incrementa a lo largo de los pasos y puede ser considerado despreciable. Por tanto, asumiendo este error, podemos usar la implementación segmentada en lazo cerrado, que presenta propiedades de estabilidad que la hacen deseable. Debido a estas propiedades de estabilidad, se debe realizar análisis adicional al hecho en esta tesis en esta implementación. La razón es que la covarianza del error en la posición aumenta a lo largo del tiempo, y este valor de covarianza se requiere para el smoothing. Esto se muestra en la Fig. 6.2b, mostrando que cuando suficiente tiempo ha pasado, pese al smoothing la corrección en la trayectoria es apreciable. En la Fig. 6.2 se muestra el comportamiento en las transiciones abruptas de la trayectoria estimada por la implementación en lazo cerrado segmentada, para un trayecto de prueba en línea recta. La Fig.6.2a es el segundo paso de la persona en esta trayectoria, mientras que Fig.6.2b es el 34to.

Se puede efectuar un análisis cualitativo de las distintas implementaciones. En la Fig. 6.3 se muestran superpuestas cinco realizaciones alineadas del paso de una persona caminando sobre una trayectoria en línea recta , para distintas implementaciones. Nótese que los ejes no son iguales por motivos ilustrativos. Aquí se muestra que para el procesado de un paso no aparecen grandes diferencias entre distintas implementaciones del algoritmo de smoothing.

Aunque la Fig. 6.3 muestra el efecto de smoothing, éste es ilustrado mejor en la Fig.6.4, donde son mostrados gráficos en 3D de pasos alineados para una trayectoria en línea recta, para la implementación inicial y la smoothed en lazo abierto segmentado del ZUPT-aided INS. Estos mismos pasos se muestran superpuestos en la Fig.6.5. Estos gráficos prueban que la gran corrección efectuada al final del paso en el ZUPT-aided INS sin smoothing desaparece para las implementaciones de nuestro algoritmo de smoothing. Para una trayectoria de ∼50 metros en línea recta, las correcciones llevadas a cabo aumentan a lo largo del tiempo, del mismo modo que lo hace la covarianza del error en la po-

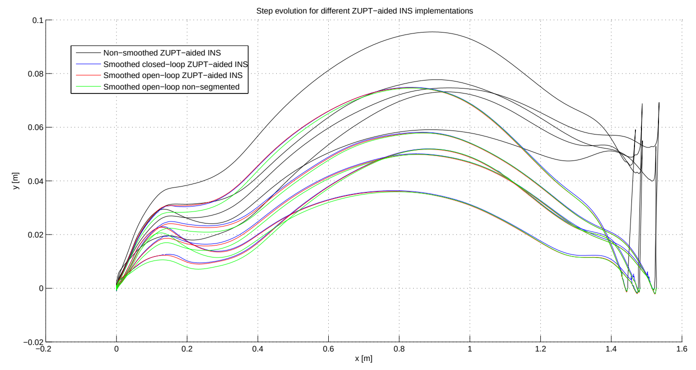


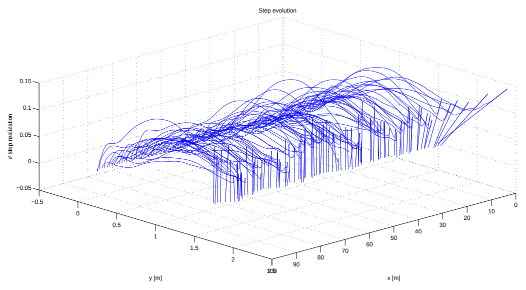
Figura 6.3: 5 realizaciones de pasos para distintas implementaciones

sicion. Al final de esta trayectoria, para la implementación inicial el orden de las correcciones es de $1 \cdot 10^{-1}$ a $1{,}5 \cdot 10^{-1}$ m. Para las implementaciones en lazo cerrado, el orden de las correcciones es de $5 \cdot 10^{-2}$ m. Para la implementación en lazo abierto segmentada, donde al final de la trayectoria de 50 m. las correcciones efectuadas son del orden de 1 a $2 \cdot 10^{-2}$ m. Para la implementación en lazo abierto no segmentada, el orden de estas correcciones es de 1 a $1{,}5 \cdot 10^{-2}$ m. Nótese que estas correcciones son ligeramente menores que para la implementación segmentada equivalente, pese a que toda la información de los futuros pasos se encuentra disponible para la implementación sin segmentar. Por tanto, el sistema se puede emplear para aplicaciones en casi tiempo real paso a paso sin grandes variaciones con respecto a un procesado completo off-line de todos los datos. Esta pequeña diferencia se debe a que el valor de la covarianza es ligeramente menor para la implementación sin segmentar que en la segmentada, ya que la implementación sin segmentar conoce toda la información del futuro. De todos modos, se requiere análisis adicional sobre trayectorias más largas para analizar el efecto del crecimiento de la covarianza a lo largo del tiempo. El análisis posterior a la realización de este proyecto parece indicar que este es un problema de linearización, que desaparece si el acoplo entre la orientación y la posición se fuerza a cero, a costa de afectar a la covarianza global. De momento no hemos alcanzado una solución completamente óptima.

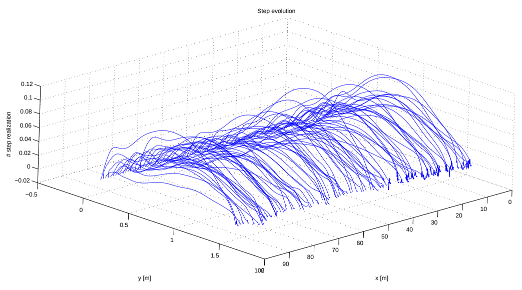
Por otro lado, la Fig. 6.6 muestra cómo la trayectoria en torno a la corrección abrupta toma ahora una forma típica de *pico* en torno al punto donde la corrección tenía lugar. Durante la ZUPT la velocidad del peatón es prácticamente cero para un gran segmento de datos, lo que produce una gran concentración de posiciones estimadas en torno a un punto. Smoothing sobre esta parte del segmento no es posible. Sin embargo, mientras el zapato no está estacionario hay una deriva en la estimación (el error va incrementándose a lo largo del paso), deriva que cuando se corregía en la implementación original se hacia mediante la corrección abrupta. El smoothing no compensa por completo esta deriva, y la trayectoria tiende a ir hacia la dirección de la deriva original, hasta que el efecto de la gran concentración de puntos durante la fase estacionaria fuerza a la trayectoria alisada a regresar a la trayectoria original.

En la Fig. 6.6 también se puede ver una vista en detalle del final de una trayectoria, que es distinto para cada implementación. Esto se debe a la distinta manera en que las estimaciones se construyen.

Un análisis de la trayectoria sobre el eje z muestra el smoothing de cada paso, donde su comportamiento básico se corresponde con lo esperado para un filtro de smoothing, tal y como se muestra a lo largo de un paso en la Fig. 6.7. Sin



(a) Implementación original



(b) Implementación segmentada open-loop

Figura 6.4: Plot en 3D de pasos alineados

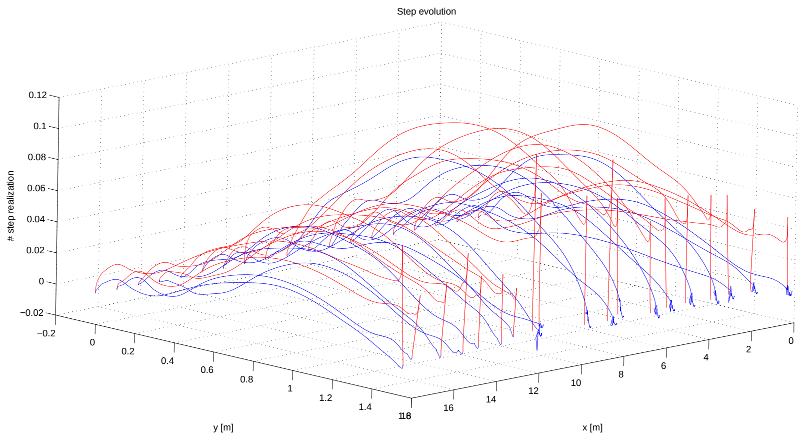


Figura 6.5: Overlapped 3D plot of aligned steps

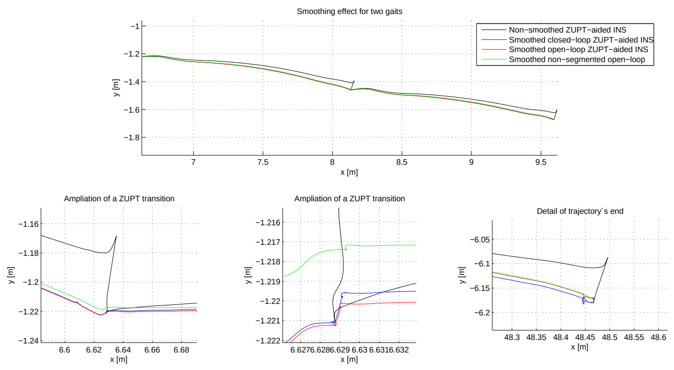


Figura 6.6: Realización típica de dos pasos. Las subfiguras muestran distintos detalles.

embargo, trayectorias que varían en el eje z no han sido testeadas, aunque no esperamos grandes diferencias.

6.2.2. Análisis de la estimación de velocidades

En la Fig.6.8a se puede observar que no existen grandes diferencias para las velocidades estimadas tras aplicar el smoothing para las distintas implementa-

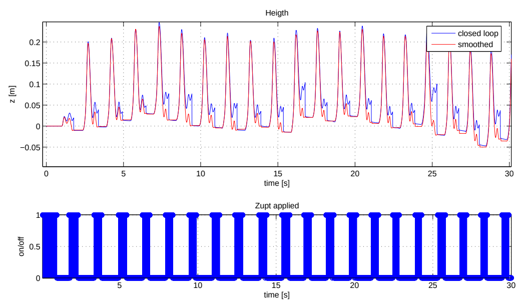
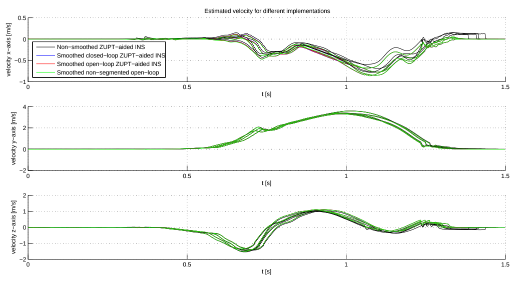
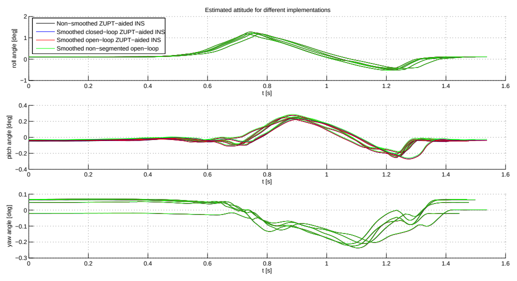


Figura 6.7: Evolución del estado estimado para el eje z.



(a) 5 realizaciones de la velocidad estimada para distintas implementaciones



(b) 5 realicaciones de la orientación estimada para distintas implementaciones

Figura 6.8: 5 realizaciones para la velocidad y la orientación estimada con distintas implementaciones

ciones propuestas. Esto se debe a que la velocidad vuelve a cero en cada fase estacionaria, decorrelando así las componentes de velocidad para cada paso sin importar qué implementación se use. Por otro lado, para esta realización específica que mostramos el smoothing se lleva a cabo principalmente en el eje x, puesto que este es el eje donde tiene lugar el mayor desplazamiento. Obsérvese ahí que la corrección drástica que aparecía para la estimación de la velocidad no aparece en las implementaciones del filtro de smoothing.

6.2.3. Análisis de las estimaciones de la orientación

La Fig.6.8b muestra distintas realizaciones de un paso para las distintas implementaciones del algoritmo de smoothing. No hay grandes diferencias entre implementaciones para los estados correspondientes a la orientación, incluso pese al smoothing. Esto es debido a que las componentes de la orientación están débilmente correladas con las estimaciones de velocidad y posición.

6.3. Análisis de la covarianza del error

El hecho de recalcular los estados estimados implica modificar la relación entre los distintos estados, esto es, modificar la matriz de covarianza entre las variables. Tal y como se muestra gráficamente en las siguientes subsecciones, para el ZUPT-aided INS original la covarianza del error incrementa mientras no se detecta ZUPT (no hay estado estacionario). Esto es debido a que el filtro hace una estimación empleando datos de previas estimaciones sin ninguna otra corrección adicional. Por tanto, cuando se detecta una ZUPT y se aplica la corrección empleando el conocimiento de la velocidad del sistema, la covarianza del error decrece drásticamente de la misma manera que las correcciones abruptas aparecían sobre los estados estimados.

Sin embargo, el smoothing modifica el carácter de la covarianza del error. Como indicamos en la sección 4.1., un problema de smoothing busca minimizar la varianza del error del vector de estados estimados $\hat{\mathbf{x}}_{n|N}$. Ahora, todo el segmento de datos se encuentra disponible desde el principio del proceso de filtrado. Esto es, la información proporcionada por la ZUPT, que permite corregir el error en la estimación, se encuentra disponible desde el principio y por tanto puede proporcionarse a cada muestra del segmento. Por tanto, es lógico esperar que la covarianza del error se minimice, si la información que reduce el error es conocida por todas las muestras del segmento.

A través de las siguientes subsecciones, se muestra un análisis cualitativo del carácter de la covarianza del error tras el smoothing, comparando las distintas implementaciones del algoritmo de smoothing con el ZUPT-aided INS del que hemos partido.Por motivos de claridad en la explicación, la covarianza del error en la velocidad abre este análisis.

6.3.1. Análisis de la covarianza del error en la velocidad

La Fig.6.9 ejemplifica la covarianza del error en la velocidad a lo largo de dos pasos de una realización típica para la implementación inicial y la smoothed en lazo abierto del ZUPT-aided INS. Para la implementación inicial el valor de la covarianza aumenta monótonamente a lo largo del tiempo mientras no se detecta ZUPT y, cuando ésta se detecta, una medida externa (la velocidad cero) proporciona información adicional que hace que la relación entre las componentes disminuya drásticamente y continúe decreciendo monótonamente durante el segmento de ZUPTs hasta un mínimo, que se mantiene hasta el final del segmento de ZUPTs.

Es sencillo percatarse que ahora hay una gran diferencia entre la covarianza del error de la implementación original y de la smoothed. ¿Por qué ocurre esto? Tras la segmentación, todos los datos se encuentran disponibles y por tanto la información proporcionada por una ZUPT, decorrelando las componentes, se hace disponible para cada muestra del paso. Esto se debe a que el filtro de smoothing proporciona información del futuro. Todos los segmentos siguen la secuencia *hay ZUPT- no hay ZUPT - hay ZUPT*. Esto significa que a ambos lados del segmento hay ZUPT y que la información proporcionada por ellos es conocida por las muestras en las fases del paso en que no hay ZUPT. Lógicamente, durante esta fase en que no hay ZUPT cuanto más cerca se esté de un punto con ZUPT, más decorreladas estan las componentes ya que mas próximo se está a la información que da la ZUPT y decorrela las componentes. Esto es la causa de que la evolución de la covarianza a lo largo del tiempo posea esta forma de "semicircunferencia"durante las fases en que no hay ZUPT: como hay ZUPTs a ambos lados del segmento, cuanto más lejos se esté de una fase con ZUPT (que da la corrección), más alto será el valor de la covarianza del error (puesto que mayor será el error). Por tanto, esta forma de semicircunferencia muestra claramente que la información poporcionada por las ZUPT se encuentra disponible para todo el paso. La Fig. 6.10 muestra esto de manera más intuitiva. La covarianza del error aumenta monótonamente durante una fase no estacionaria del paso. Si el sistema se ejecuta de manera anticausal, la covarianza del error

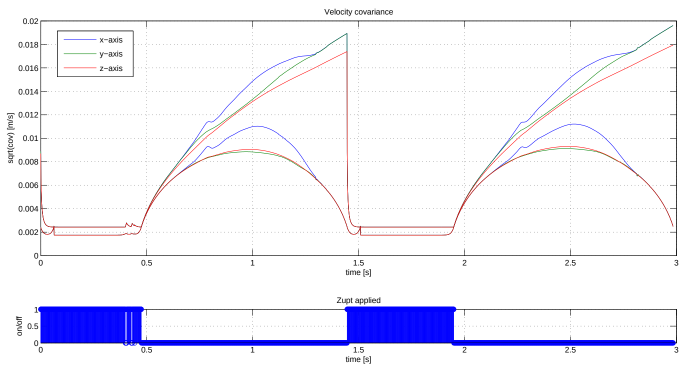


Figura 6.9: Evolución típica de la covarianza del error en la velocidad para dos pasos.

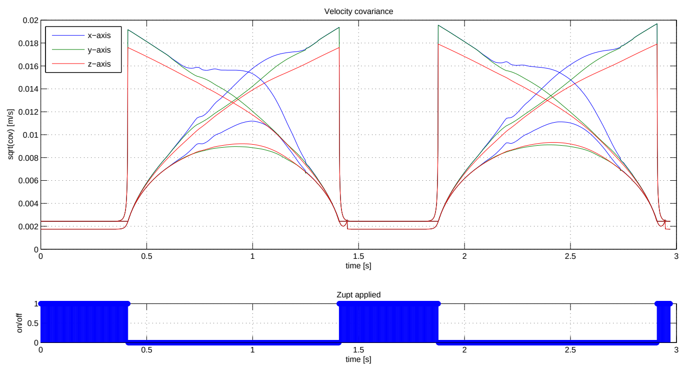


Figura 6.10: Evolucion típica de la covarianza del error en la velocidad para dos pasos. Superpuesto con la realización anticausal.

aumenta del futuro hacia el pasado. Cuando se aplica el smoothing, la información del futuro y el pasado se hace disponible a la vez para cada muestra. Por tanto, la covarianza del error adopta la forma simétrica que se ha motivado en este párrafo.

Otros dos aspectos pueden remarcarse de la Fig. 6.9. En primer lugar, hay una discontinuidad en la covarianza cada vez que un segmento se corta, como puede verse en $t = 1{,}5$ seg. El valor del extremo izquierdo se corresponde a la primera iteración del algoritmo de smoothing, el cual procesa los datos hacia atrás, véase la ecuación 4.1. Por tanto, la muestra alisada no tiene información adicional del futuro $\hat{\mathbf{x}}(N|N)$ y el valor de la covarianza de la implementación inicial es el mismo para la implementación inicial y la smoothed. Conforme los datos se van procesando hacia atrás, la información del futuro se hace disponible para las siguientes muestras $\hat{\mathbf{x}}(N-1|N), \hat{\mathbf{x}}(N-2|N), ...$ y así la covarianza disminuye a un mínimo. Nótese que si el umbral de tiempo empleado fuera menor, no habría información suficiente disponible de la proporcionada por la ZUPT y el smoothing realizado no sería correcto.

La segunda situación a remarcar de la Fig. 6.9 es el valor de la covarianza durante la ZUPT, puesto que éste es menor que para la implementación inicial sin smoothing. Para esta implementación, disminuye de $2{,}432 \cdot 10^{-3}$ a $1{,}746 \cdot 10^{-3}$ para cada eje. El conocimiento muestras adicionales del futuro explica esta disminución.

La Fig.6.11. muestra diez pasos para cada implementación del algoritmo de

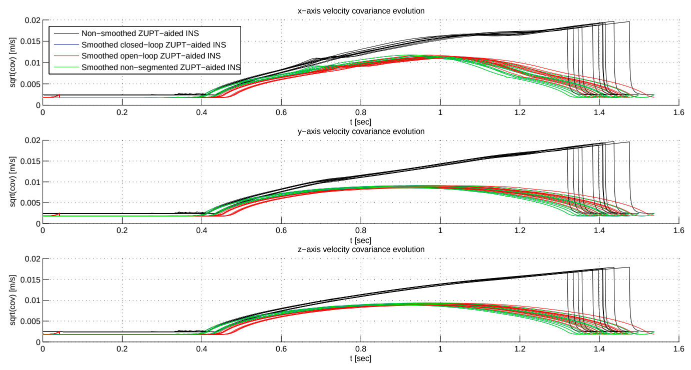


Figura 6.11: Covarianza del error en la velocidad para distintas implementaciones. 10 realizaciones por implementación.

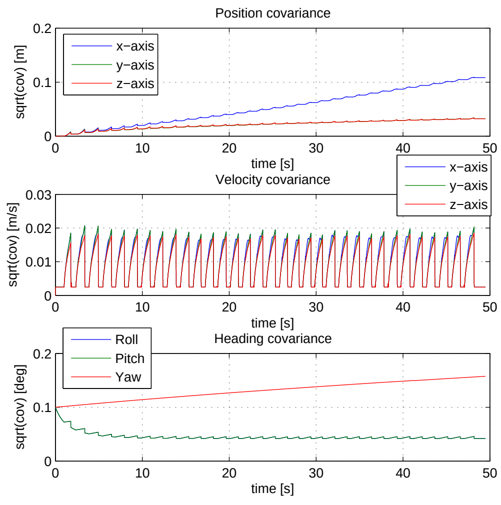
smoothing para cada eje. De ahí puede concluirse que no hay diferencias significativas entre implementaciones en términos de comportamiento de covarianza y por tanto, el análisis llevado a cabo para la implementación en lazo abierto puede hacerse extensiva a otras implementaciones. Los valores mínimos de la covarianza son los mismos para cada implementación. El hecho de que la implementación en lazo abierto aparezca ligeramente retrasada se debe a que los puntos de segmentación decididos son distintos que los decididos para la implementación en lazo abierto. Además, la implementación sin segmentar no presenta la discontinuidad que aparece para la segmentada, puesto que no hay segmentación.

6.3.2. Análisis de la covarianza del error en la posición

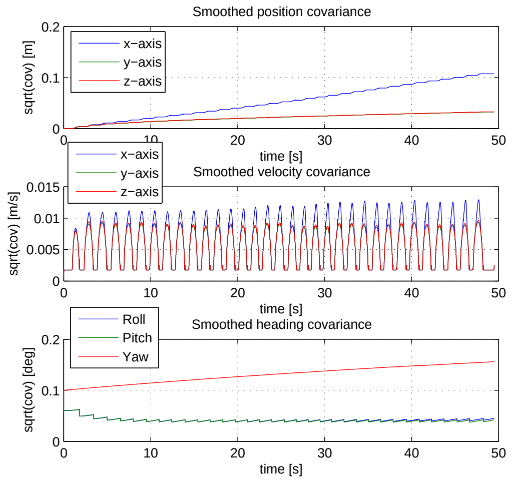
Fig. 6.12. muestra la evolución temporal para la covarianza del error en la posición, velocidad y orientación para una realización completa del sistema; y la Fig. 6.13 ejemplifica la evolución de la covarianza del error en la posición a lo largo del tiempo para dos pasos de una realización para la implementación inicial y el smoothed no segmentado en lazo abierto ZUPT-aided INS

En primer lugar, en la Fig. 6.12 se observa que la covarianza del error en la posición aumenta a lo largo de tiempo para distintas implementaciones. Una ZUPT estima que la velocidad debería ser cero, y a partir de esta estimación corrige el resto de las componentes. La corrección sobre la covarianza en la posición es apreciable, pero no fija el error en la posición a cero a diferencia del error en la velocidad. Por tanto, durante un segmento de ZUPTs las componentes de la covarianza del error en la posición no se decorrelan casi por completo como sí ocurría con las de la velocidad. Pese a todo, la covarianza cuando un segmento de ZUPT comienza sigue decreciendo drásticamente pese a no decorrelarse completamente, como se puede ver en la Fig 6.13.

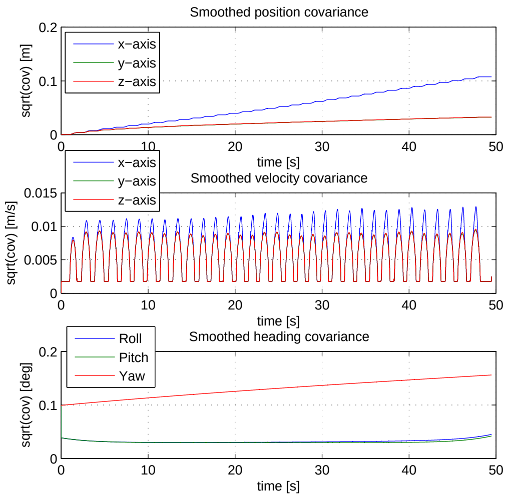
La evolución de la covarianza del error en la posición a lo largo de un paso se ilustra en la Fig. 6.13. Esta evolución a lo largo de un paso es distinta que para la covarianza del error en la velocidad. En la implementación original, la covarianza sólo aumenta cuando no hay ZUPT y, tan pronto como la ZUPT ocurre, disminuye drásticamente pero no hasta el valor que tenía al inicio del segmento. Así, conforme el tiempo avanza el valor de la covarianza del error en la posición va aumentando. Para la implementación en lazo abierto segmentada, el efecto del smoothing se aprecia en el hecho de que ahora no existe la drástica disminución que había en la covarianza de la posición: como la información provista



(a) Original



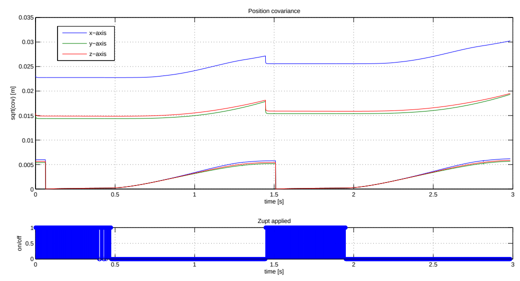
(b) En lazo-abierto segmentado



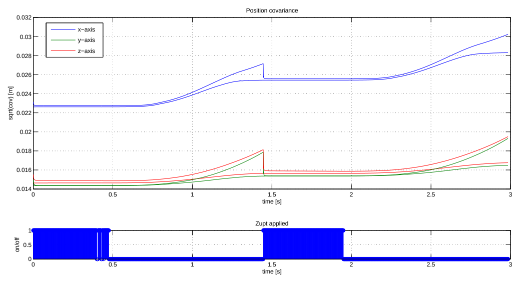
(c) En lazo abierto no segmentado

Figura 6.12: Evolucion temporal de la covarianza del error en la posición, velocidad y orientación

por las ZUPT se encuentra disponible para todo el segmento, es lógico que la covarianza aumente mientras no hay ZUPT pero sin disminuir en ningún momento cuando hay alguna corrección por la ZUPT, puesto que esta información ya se ha hecho disponible a lo largo del paso. Finalmente, para la implementación no segmentada en lazo abierto la covarianza evoluciona igual que para la implementación segmentada. Sin embargo, como toda la información del futuro se encuentra disponible el valor de la covarianza del error en la posición es ligeramente menor, pues más información es conocida. Sin embargo, como casi toda la información la aporta el propio paso, la disminución es prácticamente despreciable, haciendo por tanto válida la aproximación paso a paso seguida para lograr una implementación en casi tiempo real.



(a) Covarianza del error en la posición para dos pasos. Implementación original vs. lazo abierto segmentada.



(b) Covarianza del error en la posición para dos pasos. Implementación original vs. lazo abierto no segmentada

Figura 6.13: Covarianza del error en la posición típica para dos pasos

6.3.3. Análisis de las covarianzas en el error en la orientación

En la Fig. 6.12 puede observarse la evolución temporal de la covarianza del error en la orientación para distintas implementaciones. Puede observarse cómo la implementación original presenta algunas correcciones drásticas, particularmente para el yaw, que no aparecen para la implementacion en lazo abierta segmentada, donde estas correcciones se han alisado. En general el valor medio de la covarianza es ligeramente menor para la implementación en lazo abierto no segmentada debido al conocimiento de muestras del futuro proporcionado por el smoothing.

Mención especial requiere la diferencia entre las covarianzas del roll y el pitch, superpuestas en los gráficos, con la del yaw. ¿Por qué estas correcciones drásticas aparecen para la covarianza del yaw pero no para las del roll y el pitch? El roll y el pitch son observables por las medidas del INS. Sin embargo, el ángulo de yaw no es observable, reflejándose así en un valor más alto de covarianza del error.

6.4. Conclusiones del análisis

A partir del análisis realizado en las anteriores secciones, ciertas conclusiones pueden alcanzarse. En primer lugar, para conseguir una implementación para casi tiempo real, el algoritmo propuesto en lazo abierto segmentado funciona adecuadamente, proporcionando resultados prácticamente idénticos a los que se obtendrían con todo el set de datos, que se correspondería con un procesado off-line de los datos equivalente a la implementación en lazo abierto no segmentada realizada, y que también hemos comprobado funciona adecuadamente. Los efectos en los estados estimados han sido analizados y la mejora en las correcciones, cuantificada. Análisis adicional se requiere para la posibilidad de implementar una versión en lazo cerrado segmentada del smoothed ZUPT-aided INS empleando el algoritmo de smoothing propuesto, debido a las propiedades de estabilidad que la implementación en lazo cerrado presenta.

Por otro lado, se ha llevado a cabo un análisis para la covarianza del error para las distintas implementaciones. Distintos efectos debidos a la segmentación y a la implementación en lazo abierto son analizados, así como el cambio de comportamiento de la covarianza para el smoothed ZUPT-aided INS, alcanzando ahora una forma simétrica que muestra que el smoothing está hecho correctamente, puesto que esta simetría se ha alcanzado mediante el conocimiento de la información que las ZUPT futuras proporcionan.

Capítulo 7

Conclusiones y futuro trabajo

El análisis de un foot-mounted ZUPT-aided INS y de varios algoritmos generales de smoothing han permitido combinarlos e implementar un smoothed ZUPT-aided INS. Para su implementación, se han considerado y solucionado distintos problemas y se ha efectuado un análisis de diferentes implementaciones del algoritmo de smoothing propuesto. Este último capítulo resume las conclusiones alcanzadas tras este proceso y sugiere futuro trabajo a ser realizado en el área.

7.1. Conclusiones

A lo largo de este proyecto hemos motivado y propuesto un algoritmo de smoothing paso a paso para ZUPT-aided INSs. Primero se ha efectuado un análisis del sistema disponible, una implementación típica de un ZUPT-aided INS consistente en un filtro en lazo cerrado de Kalman. A continuación, se ha llevado a cabo un análisis de distintos algoritmos de smoothing, eligiendo como base para el algoritmo propuesto las fórmulas de smoothing RTS. Para combinar el sistema disponible con este algoritmo de smoothing, se requiere la implementación en lazo abierto del habitual filtro de Kalman en lazo cerrado que usan los ZUPT-aided INSs permitiendo así la identificación directa de los términos de la fórmula de smoothing RTS con las variables del filtro. Además, debido a que los pasos están espaciados irregularmente y las medidas (las ZUPTs) aparecen en grupos, una regla de segmentación de datos ha sido propuesta para permitir el procesado de datos paso a paso. Esta regla está basada en umbrales de tiempo y de covarianza del error. Considerando estos dos aspectos, se ha propuesto un algoritmo de smoothing en lazo abierto paso a paso para ZUPT-aided INSs. Después se ha llevado a cabo un análisis cualitativo de ésta y otras implementaciones de smoothing. El algoritmo propuesto será implementado para aplicaciones en casi tiempo real.

7.2. Futuro trabajo

Se considera que el smoothing para foot-mounted ZUPT-aided INSs será muy útil para la futura investigación en navegación y seguimiento de personas. Hasta ahora, las correcciones en la trayectoria estimada por los ZUPT-aided INSs se llevaban a cabo empleando sistemas adicionales tales como cámaras también cargadas por la persona, lo cual no es práctico. El algoritmo propuesto no requiere ningún sensor adicional, haciéndolo así una solución más apropiada para navegación inercial.

En el ámbito de este proyecto, se espera realizar trabajo adicional. Por ejemplo, se requiere testeo adicional para la regla de segmentación para desplazamientos más rápidos, donde un proceso de tuning de los umbrales de la regla de segmentación y de detección de ZUPTs para estas situaciones debe ser considerado.

En la versión original de este proyecto se indicaba que debido a la divergencia a lo largo del tiempo de los algoritmos en filtro abierto, se debía realizar análisis adicional para periodos de tiempo más largos. Hemos testeado trayectorias en línea recta de entre cinco y diez minutos de duración, y el algoritmo funciona correctamente. Del mismo modo, hemos intentado minimizar el problema del rizado que aparece tras el filtrado de smoothing y que crece con el tiempo. Como hemos indicado previamente, parece ser un problema de linearización que se puede eliminar desacoplando las componentes de orientación y posición. Sin embargo, de este modo la covarianza global se ve afectada. Todavía seguimos trabajando intentando resolver este problema de forma más conveniente.

Por otro lado, el código empleado en nuestras simulaciones ha de ser ahora programado para un INS real, permitiendo además su funcionamiento en casi tiempo-real, tal y como hemos incidido a lo largo de la tesis.

Finalmente, el trabajo realizado debe ser presentado a la comunidad científica, puesto que esperamos que otros investigadores en el ámbito de la navegación puedan estar interesados en este trabajo para facilitar su labor en el área. Para ello, tras la realización de este proyecto hemos escrito el artículo *Smoothing for ZUPT-aided INSs* para la Conferencia Internacional de Posicionamiento y Navegación en Interiores (International Conference on Indoor Positioning and Indoor Navigation (IPIN)), que tendrá lugar entre el 13 y el 15 de Noviembre de 2012, y que se encuentra adjunto en el apéndice B. Del mismo modo, este trabajo se ha presentado en conferencias del sector en Suecia durante los últimos meses.

Bibliografía

[1] Jay A. Farrell, Matthew Barth; *The Global Positioning System & Inertial Navigation*; McGraw Hill Professional, Dec 1998.

[2] Jay A. Farrell; *Aided navigation - GPS with high rate sensors*; McGraw Hill Professional, 1st edition, Apr 2008.

[3] J.L. Farrell; *Integrated Aircraft Navigation*; Academic, New York, 1976.

[4] Carl Fischer, Poorna Talkad Sukumar, Mike Hazas; *Tutorial: implementation of a pedestrian tracker using foot-mounted inertial sensors*; IEEE Pervasive Computing, 09 Jan. 2012. IEEE computer Society Digital Library.

[5] Eric Foxlin; *Pedestrian tracking with shoe-mounted inertial sensors*; IEEE Computer Graphics and Applications, vol. 25, no.6, Nov-Dec 2005.

[6] A. Jiménez, F. Seco, J. Prieto, and J. Guevara; *Indoor Pedestrian Navigation using an INS/EKF framework for Yaw Drift Reduction and a Foot-mounted IMU*; in Proc. of WPNC, 2010.

[7] Thomas Kailath, Ali H. Sayed, Babak Hassibi; *Linear estimation*; Prentice Hall, 1st edition, 2000.

[8] Thomas Kailath, Lennart Ljung; *Two Filter Smoothing Formulae by Diagonalization of the Hamiltonian Equations*; Stanford univ. CA information systems lab., 1982.

[9] Steven M. LaValle; *Planning Algorithms*; Cambridge Univ. Press, 2006.

[10] Anthony Lawrence; *Modern Inertial Technology: Navigation, Guidance, and Control*; Springer, mechanical engineering series, 2nd edition, 1998.

[11] Maciej Niedźwiecki; *Locally Adaptive Cooperative Kalman Smoothing and Its Application to Identification of Nonstationary Stochastic Systems*; IEEE transactions of signal processing, vol. 60, no.1, Jan 2012.

[12] John-Olof Nilsson, Isaac Skog & others; *OpenShoe*, `www.openshoe.org`; 2011.

[13] PooGyeon Park, Thomas Kailath; *New Square-Root Smoothing Algorithms*; IEEE transactions of signal processing, vol. 41, no. 5, May 1996.

[14] Isaac Skog, John-Olof Nilsson, Peter Händel, Jouni Rantakokko; *Zero-Velocity Detection - An Algorithm Evaluation*; IEEE Transactions on Biomedical Engineering, vol. 57, no.11, Nov. 2010.

[15] Joseph E. Wall, Jr., Alan S. Willsky, Nils R. Sandell, Jr.; *On the Fixed-Interval Smoothing Problem*; Gordon and Breach Science Publishers Inc., *Stochastics*, 1981, Vol. 5, pp. 1-41.

Anexos

Anexo A

Documento original de la tesis

Step-wise smoothing of ZUPT-aided INS

DAVID SIMÓN COLOMAR

KTH Electrical Engineering

Master's Degree Project
Stockholm, Sweden August 27, 2012

XR-EE-SB 2012:011

Abstract

Due to the recursive nature of most foot-mounted zero-velocity-update-aided (ZUPT-aided) inertial navigation systems (INSs), the error covariance increases throughout each step and “collapses” at the end of the step, where the ZUPT correction is done. This gives sharp corrections and discontinuities in the estimated trajectory. For applications with tight real-time constraints, this behavior is unavoidable, since every estimate corresponds to the best estimate given all the 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, giving the sharp correction, can be made available throughout the step. Consequently, to eliminate the sharp corrections and the unsymmetrical covariance over the steps, the implementation of a smoothing filter for a ZUPT-aided INS is considered in this thesis. To our knowledge, no formal treatment of smoothing for such systems has previously been presented, even though an extensive literature on the general subject exists.

Owing to the customary closed-loop complementary filtering used for aided INS, standard smoothing techniques cannot directly be applied. Also since the measurements (the ZUPTs) are irregularly spaced and appear in clusters, some varying-lag smoothing rule is necessary. Therefore, a method based on a mixed open-closed-loop complementary filtering combined with a Rauch-Tung-Striebel (RTS) smoothing is suggested in this thesis. Different types of varying-lag smoothing rules are examined. For near real-time applications, smoothing is applied to the data in a step-wise manner. The intervals (steps) for the smoothing are determined based on measurement availability and covariance and timing thresholds. For complete off-line processing, full data set smoothing is examined. Finally, the consequences of the smoothing and the open-closed-loop filtering are quantified based on real data. The impact of the smoothing throughout the steps is illustrated and analyzed.

Abstrakt

På grund av den rekursiva karaktären hos de flesta nollhastighetsuppdaterade tröghetsnavigeringssystem (på engelska, ZUPT-aided INSs), ökar felets kovarians över varje steg och “ kollapsar ” i slutet av steget, där nollhastighetsuppdateringen görs. Detta ger kraftiga korrigeringar och diskontinuiteter i den skattade banan. För tillämpningar med håda realtidsbegränsningar, är detta beteende oundvikligt eftersom varje skattning motsvarar den bästa uppskattningen få all information fram till den tidpunkten. För många tillämpningar kan en viss grad av eftersläpning (icke-kausalitet) emellertid tolereras och den information som nollhastighetsuppdateringarna tillför vid slutet av ett steg, vilket ger kraftig korrigering, kan göras tillgängliga i hela steg. För att eliminera de kraftiga korrigeringar och osymmetriska kovariansen över stegen, behandlads i detta examensarbete implementeringen av ett glättningsfilter för ett nollhastighetsuppdaterade tröghetsnavigeringssystem. Såvitt vi vet har ingen formell behandling av glättningsfilter för sådana system tidigare lagts fram, trots att en omfattande litteratur i ämne finns.

På grund av de brukliga återkpllade filtrering som används för understödda tröghetesnavigering, kan vanliga glättningstekniker inte direkt tillämpas. Eftersom ätningarna (nollhastighetsuppdateringarna) är oregelbundet placerade och förekommer i kluster, behövs ett glättningsfilter med varierande-fördröjning användas. Därför föreslås en metod som byggs på en växelvis öppen/återkopplad filtrering i kombination med en Rauch-Tung-Striebel (RTS) glättning i detta examensarbete. Olika metoder för återkoppling och glättning undersöks. För nära realtidstilämpningar tillämpas glättningen stegvis på datan. Intervallen (stegen) för glättningen bestäms på grundval av mätningarnas tillgänglighet och filterkovariansens värden i förhållande till tröskelvärden. För tillämpningar helt utan krav på eftersläpning undersöks användet av hela datamängd för glättning. Slutligen undersöks konsekvenserna av glättningen och de öppna-slutna filtrering kvantifieras baseras på verkliga data. Effekten av glättningen längs stegen illustreras och analyseras.

Acknowledgements

Stockholm is the place where I have done this thesis. But Stockholm is as well the place where I have spent one of the best years of my life. Now it is time to thank all the people that helped me to do this thesis, and that allowed me to enjoy this amazing year in the north.

First, I would like to sincerely show my gratitude to my supervisor John-Olof Nilsson for his continuous support during these last months. I really enjoyed working with him.

I would also like to thank Andrea Rizzi, David Aguilar Pérez and in general all the friends who shared with me hours of studying, work, frustrations... and yes, laughs and joy as well. Without them, writing this thesis would not have been the same.

In the same way, I want to thank my friends, Erasmus and from Spain, who made me enjoy of one of the best periods of my life and sometimes listened, beyond the relativity limits, to my problems.

Last, but not least, I would like to show gratitude to my family. For years of patience, encouragement and guidance. Finally I am here, where my degree comes to an end. Without them, this would not be possible.

Thank you all, and thank you very much.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

A foot-mounted inertial navigation system (INS) allows pedestrian dead reckoning. Owing to the recursive nature of most of these systems, they are subject to cumulative errors. For a foot-mounted INS, the position and velocity estimated error increases along each step and "collapses" at the step's end, when a zero-velocity-update (ZUPT) gives information which allows the correction of the estimates. This gives sharp corrections and therefore discontinuities in the estimated trajectory. The aim of this thesis is to design a smoothing filter to avoid these discontinuities. Throughout this first chapter an overview of the already implemented system is provided, motivating the need of a step-wise smoother for a ZUPT-aided INS. Finally, the structure of the rest of the thesis is provided and the main references are indicated.

1.1 Foot-mounted ZUPT-aided INS

Dead-reckoning is the process of estimating the current position of a body by using previously determined position and velocity values, as well as known or estimated speeds of the body. Particularly, an inertial navigation system (INS) consists of a set of sensors that allow to continuously calculate position, velocity and attitude (orientation) of a body by applying physical laws of motion to an initial known state. Inertial sensors which can be worn, as the foot-mounted INS used in this thesis, have desirable properties to track pedestrians in situations where other localization systems (as GPS) fail, as it can be in indoors environments or dense forests. As an INS applies laws of motion that do not require the detection of external signals that could be affected by interference, an INS provides reliable estimations in these situations. Thus, pedestrian dead-reckoning has been proposed for many applications, such as defense, for emergency rescue workers or for smart offices.

OpenShoe is an open source embedded foot-mounted INS implementation which includes both hardware and software design. It is the result of the colaboration between the signal processing department in KTH, and the statistical signal processing in the Indian Institute of Science (IISc) in Bangalore, India. Fig.1.1. shows an implementation of the system. There, rightwards, can be seen a section of the shoe where the INS is the *white box* embedded in the shoe heel. In [12] can be found a fully documented implementation of the system. This implementation is the one where this thesis is going to work on.

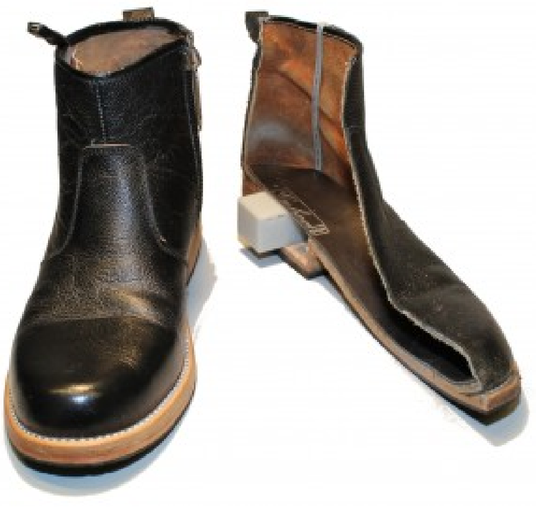


Figure 1.1: OpenShoe implementation.

The main advantage of this system is that it measures movement, while most of the pedestrian navigation systems try to estimate which kind of step the pedestrian has done and its length. This becomes a problem when the movement becomes irregular. However, as the employed implementation measures movement, it does not matter if the pedestrian runs, jumps or walks; it will still track movement and therefore, a position will be provided. The error in the estimation provided by the foot-mounted ZUPT-aided INS can be as low as 0.14% of the total traject, for straight-line trajectories, and in general it will be at least equal or better than equivalent systems. In the same way, important values of this foot-mounted INS are its modularity and its small weight, volume and price faced against the typical sensor-plus-laptop research systems. These features will make feasible to equip a large number of users with foot-mounted INS units. The original work in foot-mounted ZUPT-aided INS is [5], article which provides a good background of the topic.

The goal of a foot-mounted INS is to provide a reliable pedestrian dead-reckoning process. That is, it aims to accurately estimate the position and velocity of a person by tracking its movements from an initial known state. This tracking has a recursive structure. Since the current estimated state depends on the previous one, as the time passes, the estimated state error increases. Therefore, it is required to provide a correction to the position and velocity estimates every short periods of time.

A zero-velocity-updated (ZUPT) aided INS provides this correction while a stationary state occurs. A stationary state takes place while the shoe's heel where the INS is embedded is in contact with the ground. During this time interval, the INS is in a stationary state, and the estimated velocity should be zero. If it is not, the system takes profit of this knowledge to correct the position and velocity estimations. Since these stationary states occur at most every few seconds this correction can be done regularly and hence the error in the estimation never increases too much.

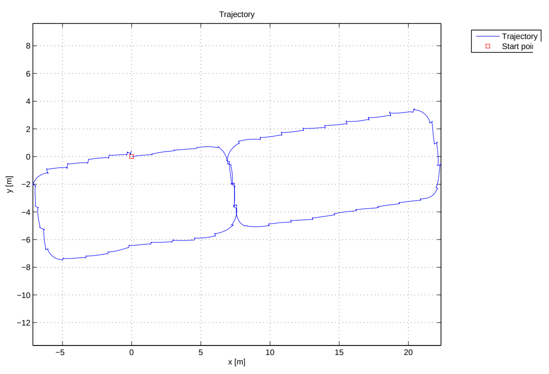
The ZUPT-aided INS tracking process can be divided into three different parts for each instant of time: First the caption of the data by the sensors contained in the inertial measurement unit (IMU); second the detection of the ZUPT intervals when the shoe is stationary; and third the running of a ZUPT-aided

Kalman filter. This ZUPT-aided Kalman filter provides for each instant of time an estimation of the position and the velocity of the pedestrian based on the current IMU measurements and the previous estimated state by combining them using laws of mechanic. Then, this estimation is corrected by a ZUPT if the foot is detected to be in a steady state, since during that time instants the error is known.

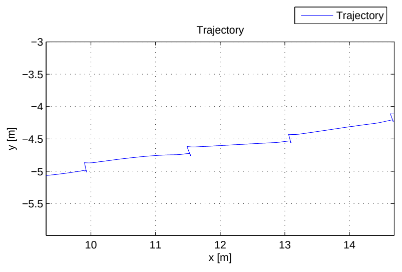
As a real time application, these three steps are done in a continuous way as soon as the measurements are available, allowing the movement's tracking. However, in the Matlab implementation where this thesis will work on these steps are done sequentially one after the previous one is finished.

1.2 Motivation of the thesis

Owing to the nature of the system, discontinuities in the trajectory estimated by a ZUPT-aided INS appear. A typical by the ZUPT-aided INS estimated trajectory is shown in Fig.1.2a. A detailed view of three steps of this trajectory is shown in Fig.1.2b.



(a) Estimated whole trajectory for xy-axis



(b) Detailed view of three steps of the estimated trajectory for xy-axis

Figure 1.2: Estimated trajectory for xy-axis

Analyzing these plots, it is easy to discover a sort of *peaks* that give the estimated trajectory a *rough* appearance. This situation is also found for the z axis, as well as for the velocity estimates. It is caused by the correction performed by the Kalman filter when the information provided by a zero velocity update becomes available. Along a step, the estimation error is increasing. At the end of the step, just before a ZUPT is applied, the error is large. Therefore, the correction applied over the estimated trajectory when the information provided by the ZUPT becomes available is large, appearing these *peaks* that have been shown for every step.

For applications with tight real-time constraints, this behavior is unavoidable, since every estimate corresponds to the best estimate given all the information until that time instant. However, for motion analysis and visualization, these large corrections are undesirable. For many applications, some degree of lag (non-causality) can be tolerated. The information provided by the ZUPTs at the end of a step, causing the sharp correction, can be made available throughout the step. Consequently, the implementation of a step-wise smoothing filter for a ZUPT-aided INS is considered to eliminate the *sharp* corrections and the unsymmetrical covariance over the steps.

To our knowledge, no formal treatment of smoothing for such ZUPT-aided INS has previously been presented, but an extensive literature on the general subject area can be found. Such smoothing filter is judged to be very useful for the further research in the area. Therefore, the aim of this thesis is to implement a specific smoothing filter algorithm for a ZUPT-aided INS.

The thesis has been done in the Signal Processing Laboratory in KTH, with the supervission of John-Olof Nilsson. The examiner is Peter Händel.

1.3 Problem definition

Due to the customary closed-loop complementary filtering used for aided INS, standard smoothing techniques cannot directly be applied. New issues that the general smoothing problem which is found in the literature does not face must be considered. The master thesis work consists of evaluating implementation options and implementing a smoothing algorithm based on this analysis. Further, the smoothing algorithm implementation is to be used to characterize the applied corrections over a typical step relative to the original recursive filter implemntation.

Since the discontinuities appear in a step-wise manner, smoothing of a single step can be considered. Moreover, it is desired to get a system which can be implemented with a near to real-time operation in the future. However, smoothing filters are non-causal techniques of signal processing. By admiting some lag, a segmentation rule is required to be able to create shorter segments of data, which can be identified with an step. These short segments of data can be smoothed as soon as all the components in the segment become available, allowing a near to real-time operation. Therefore, routines for automatic data segmentation must be motivated and designed.

1.4 Thesis overview

This thesis is divided in seven chapters. Through this chapter 1 an overview of the system has been given and the problem that it is aimed to solve has been introduced. During the following chapters, the necessary knowledge to achieve a full understanding of how the already implemented ZUPT-aided INS works is provided. Thus, chapter 2 goes through different aspects of the implementation in depth, by introducing the inertial navigation basics and providing an error model for it. Chapter 3 deals with the Kalman filtering: which are its fundamentals and how it can be used in a ZUPT-aided INS. In order to compensate the error in the estimates, ZUPT aiding has already been shown as a valid error correction method. Here the ZUPT-aiding operation is explained. The next chapters show the work carried out in this thesis. In chapter 4 are reviewed general smoothing algorithms that are used as the basis to deal with the smoothing problem. An analysis of all of the algorithms is performed, leading to the choice of one for its implementation. Chapter 5 deals with the different implementation issues that appear for combining ZUPT-aided INSs with general smoothing algorithms. Particularly, the problem of the data segmentation is considered for the data subsequent processing, as well as it is motivated why an open loop implementation is considered for the correct operation of the system.

Finally, the smoothed ZUPT-aided INS process is shown. Chapter 6 consists of analysis and comparison with the original implementation of different implementations of the proposed smoothing algorithm. Eventually chapter 7 comes to conclusions and shows the future work that can be done in this topic.

1.5 Main references

This thesis is closely related to inertial navigation systems. Due to the broad range of involved technologies, this thesis tries to motivate and take the different results that requires for explanations. For further details and complete demostrations, some guidelines and references are given here to the reader.

The two main sources that we understand can provide a good theorical understanding of the system are [1], for chapters 2 and 3; and [7], for chapter 4. The first provides a complete analysis of INSs, while the second one introduces the main general smoothing algorithms. It is convinient to point out that this thesis does not present the coordinates systems, plane rotation equations, the simplification underlying in the use of quaternions nor the state-space system equations that actually are the basis where the error model is based, the three first ones; and the basis from where to derive the Kalman filter, the last one.

Chapter 2

Inertial navigation system – model and assumptions

Next two chapters present the already implemented ZUPT-aided INS and its operation. Along this chapter is shown the processing of the INS sensors measurements for getting the position, velocity and attitude estimates for one iteration. Afterwards, next chapter shows how the recursive propagation of the estimations is done based on the new measurements and on the previous estimations. Thus, this chapter begins explaining how the INS is and explaining conceptually how it works. Then both, the mechanization equations behind this process and the system error model are introduced. The simplifications that are done by the available ZUPT-aided INS are also motivated along this chapter.

2.1 The IMU – Accelerometers and gyroscopes

There are plenty of navigation applications, which use different types of sensors. Some examples are radar or sonar systems, ... The main difference between them and an inertial navigation system, as the one used in this thesis, is that INSs consist of sensors based on physical laws of motion which do not require the detection of external signals while other kind of navigation systems do require the detection of external signals, such as electromagnetic fields or pressure waves in the previous examples. Therefore, an INS deals much better with the external interference, providing more reliable measurements. Therefore, INSs provides reliable measurements in hard environments where other navigation systems fail, such as indoors environments or dense forests.

An inertial measurement unit (IMU) is an electronic device that measures and reports the body's velocity, orientation and gravitational forces. A typical IMU for a foot-mounted INS, consists of three accelerometers and three gyroscopes. They measure the body's motion state by measuring the change of the states produced by accelerations affecting the body. If the body's initial position and velocity is known, then the body's movement can be tracked. This process is called dead-reckoning.

Placed in different aligned orthogonal axes, each accelerometer provides the measurement of the specific force in one axis. Specific force is defined as the non-gravitational force per unit mass. That is, the difference between the inertial acceleration and the gravitational acceleration of a body. Since the IMU is moving, the frame where the accelerometers are placed is always moving. This *moving* frame is called *body frame* (also *vehicle frame* in many texts).

The idea of the gyroscope is to know how much the body frame differs from an always *fixed* frame, the so called *navigation frame.* This allows to build a rotation matrix from the body to the navigation frame. Then, the specific force measurements are transformed from the body to the navigation (*fixed*) frame. Having the specific force expressed always in the same reference, the navigation frame, mechanical laws of motion can be applied to calculate accurately the position of the body. The body and navigation frames are illustrated in Fig.2.1., where it can be seen that the body and the body frame move together, while the navigation frame is located on a fixed plane tangent to the Earth's surface.

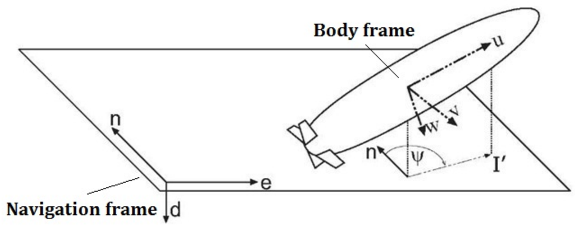


Figure 2.1: Body and navigation frames

Once the coordinate transformation is calculated and the IMU measurements are expressed in the navigation frame, the measurements must be integrated along the time between them in order to calculate the current position and velocity of the body by combining the result of this integration with the previous position and velocity estimates, by applying the equations shown in the next section.

Apart from the position and the velocity, also the body's attitude has to be calculated and stored between iterations. This is required because the body attitude during the previous time instant is combined with the new gyroscopes measurements in every iteration to be able to know the exact attitude of the body. This allows the correct calculation of the rotation matrices required for the coordinate transformation applied to the IMU measurements from the body to the navigation frame. Together with the position and velocity estimates for each axis, the attitude estimate form the from now on called vector of navigation *state*, which will be explained in depth in the next chapter.

Summarizing, first a coordinate transformation process and then an integration process along time allow to know the exact position of the body. One schematic of this process can be found in Fig.2.2. By using the previous state estimations as initial states and repeating this process for the next measurements every iteration, the dead-reckoning process is done. Next chapter will provide a deeper analysis about how the propagation of the estimates between iterations is done.

2.2 INS mechanization equations

In the previous section, the INS operation has been introduced conceptually. This section aims to provide the mathematical expressions and simplifications of this process, with special focus on the INS used in this thesis. That is, from the measurements provided by the IMU, how the data is processed. A simplified version of the navigation equations is used due to the fact that this is a pedestrian dead-reckoning system where distances and speeds are much

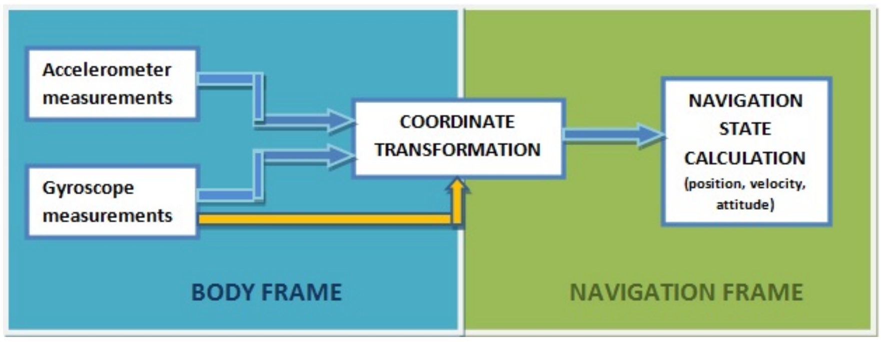

Accelerometer measurements
Gyroscope measurements
COORDINATE TRANSFORMATION
NAVIGATION STATE CALCULATION (position, velocity, attitude)
BODY FRAME
NAVIGATION FRAME

Figure 2.2: Diagram of the process performed in the INS

smaller than for aircraft, ships or land bodies. Moreover, non-high quality sensors are used. Therefore, measurement errors are bigger than the related modelling errors. The full navigation equations compensate for a number of physical effects that have little consequence on the tracking error in our case, such as the centrifugal force due to the rotation of the Earth or the Coriolis force. For further details about the position, velocity and attitude calculations and transformations for general INSs, see [1], chapter 6.2.

In order to explain the mechanization equations logically, the equations will be introduced in the same order that they were used in the process explained in the previous section. Hence, first the attitude equations will be discussed. Once the current attitude of the body is known, a rotation matrix that converts the sensors measurements to the navigation frame can be built. With this measurements in the navigation frame, the current body's velocity and position estimates can be calculated using the corresponding equations.

2.2.1 Attitude equations

Attitude consists of three different angles: roll, pitch and yaw (see Fig.2.3). These angles are used to describe the orientation of a rigid body. The gyroscope contained in the IMU determines the variation of these three angles between two time instants. By using this measurement, and the previous attitude state of the body, a current attitude state of the body can be calculated. From this calculated values, the current rotation matrix $\mathbf{R}_{b2n}$ that converts measurements in the body frame to measurements in the navigation frame can be built.

Owing to some implementation advantages, such as that they are singularity free, more robust and more computationally efficient; quaternions $q = [q_1, q_2, q_3, q_4]$ are used by our INS as the way of representing rotation matrices $\mathbf{R}_{b2n} \in \mathbb{R}^{3x3}$. Theory about quaternions can be found in [2], appendix D; or [1], pages 39-42 and 47-49.

By using quaternions properties, it can be shown (see [3]) that the quaternion derivative is

$$\mathbf{q}' = \mathbf{\Psi q} \quad \text{where} \tag{2.1}$$

$$\mathbf{\Psi} = \frac{1}{2}\begin{bmatrix} 0 & \omega_z & -\omega_y & \omega_x \\ -\omega_z & 0 & \omega_x & \omega_y \\ \omega_y & -\omega_x & 0 & \omega_z \\ -\omega_x & -\omega_y & -\omega_z & 0 \end{bmatrix} \tag{2.2}$$

where ω_x is roll rate in the body frame, ω_y is pitch rate in the body frame and ω_z is yaw rate in the body frame.

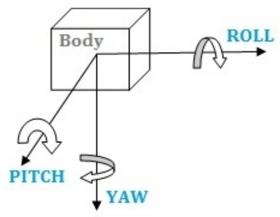


Figure 2.3: Roll, pitch and yaw in the body frame

Since the aim is to know the variation of the attitude between two instants of time t_n and t_{n-1}, and remembering that a derivative is calculated for a local time instant, the derivative shown in equation (2.1) must be integrated between t_n and t_{n-1} in order to *accumulate* the variations between the two time instants. It can be shown (see [1], pages 47-49) that the solution to this integral is

$$\mathbf{q}(t_n) = e^{\mathbf{\Omega}}\mathbf{q}(t_{n-1}) \quad \text{where} \tag{2.3}$$

$$\mathbf{\Omega} = \frac{1}{2}\begin{bmatrix} 0 & Y & -P & R \\ -Y & 0 & R & P \\ P & -R & 0 & Y \\ -R & -P & -Y & 0 \end{bmatrix} \tag{2.4}$$

$$R = \int_{t_{n-1}}^{t_n} \omega_x(t)dt, \quad P = \int_{t_{n-1}}^{t_n} \omega_y(t)dt, \quad Y = \int_{t_{n-1}}^{t_n} \omega_z(t)dt \tag{2.5}$$

As the frequency of the thesis system is high and the variables discrete, the integral's result of R, P and Y can be approximated by

$$\mathbf{q_n} = e^{\mathbf{\Omega}}\mathbf{q_{n-1}} \tag{2.6}$$

$$R = \omega_x(n)T_s, \quad P = \omega_y(n)T_s, \quad Y = \omega_z(n)T_s \tag{2.7}$$

where T_s is the sampling period of the system.

Expanding the term $e^{\mathbf{\Omega}}$ in (2.3) by a power series and truncating it yields to the expression that refreshes the quaternion between two consecutive iterations:

$$\mathbf{q_n} = \left[\cos\left(\frac{\|\vartheta\|}{2}\right)\mathbf{I} + \frac{2}{\|\vartheta\|}\sin\left(\frac{\|\vartheta\|}{2}\right)\mathbf{\Omega}\right]\mathbf{q_{n-1}} \tag{2.8}$$

where $\vartheta = (R, P, Y)^T$.

Thus, the corrected rotation matrix for the iteration n is determined by building (2.4) using the approximation in (2.7), and afterwards computing (2.8).

By identification between the rotation matrix elements and the updated quaternion components, the values of the roll ϕ, pitch θ, and yaw ψ in the navigation frame can be determined by using the next equations:

$$\phi = \arctan\left(\frac{R_{b2n}(3,2)}{R_{b2n}(3,3)}\right) \tag{2.9}$$

$$\theta = \arctan\left(\frac{R_{b2n}(3,1)}{\sqrt{R_{b2n}^2(3,2) + R_{b2n}^2(3,3)}}\right) \tag{2.10}$$

$$\psi = \arctan\left(\frac{R_{b2n}(2,1)}{R_{b2n}(1,1)}\right) \tag{2.11}$$

where $\mathbf{R}_{b2n}$ is the 3x3 rotation matrix from the body to the navigation frame, rebuilt from the refreshed quaternions values.

2.2.2 Velocity and position equations

Once the current attitude of the body is obtained, and therefore the current rotation matrix $\mathbf{R}_{b2n}$ is known, the current body's velocity and position values can be determined. In a first step, the corresponding rotation matrix is generated from the already updated quaternion $\mathbf{q}_n$ and applied to the measured body frame specific force vector $\delta\mathbf{f}^b$. This gives the navigation frame specific force vector $\delta\mathbf{f}^n$

$$\delta\mathbf{f}^n = \mathbf{R}_{b2n}\delta\mathbf{f}^b \tag{2.12}$$

Then the gravity acceleration has to be removed from the navigation axes where it affects to the specific force $\delta\mathbf{f}^n$ value, obtaining $\mathbf{a}_n$, acceleration in the navigation frame.

$$\mathbf{a}_n = \delta\mathbf{f}^n - \mathbf{g} = \delta\mathbf{f}^n - (0, 0, 9.80665)^T \tag{2.13}$$

where standard gravity $\mathbf{g}$ is defined as 9.80665 m/s^2 in yaw's direction of the navigation frame.

Finally, the acceleration $\mathbf{a}_n$ is integrated over t_n and t_{n-1} to get the position and velocity estimates. Since the frequency is high and the variables discrete, the acceleration $\mathbf{a}_n$ can be considered constant between two time samples. Thus, the basic equations of motion can be applied

$$\mathbf{p}_n = \mathbf{p}_{n-1} + \mathbf{v}_{n-1}T_s + \frac{1}{2}\,\mathbf{a}_n T_s^2 \tag{2.14}$$

$$\mathbf{v}_n = \mathbf{v}_{n-1} + \mathbf{a}_n T_s \tag{2.15}$$

where $\mathbf{p}_n$ is the position estimate at instant n, $\mathbf{v}_n$ is the velocity estimate at instant n and T_s the integration interval, that as said in the system used is the inverse of the frequency of the system's measurements. $\mathbf{p}_{n-1}$ and $\mathbf{v}_{n-1}$ are known from the previous iteration estimate, $\mathbf{p}_0$ and $\mathbf{v}_0$ initial states.

With this, the estimation of the velocity and position are obtained for each instant from the current measurements and the previous estimate.

A summary of all the mechanization equations used by the INS can be found in table 2.1.

INS mechanization equations

Get new quaternions ($\equiv$ get $\mathbf{R}_{b2n}$)	$R = \omega_x(n)T_s,\ P = w_y(n)T_s,\ Y = w_z(n)T_s$ Build $\mathbf{\Omega}$ $\mathbf{q_n} = \left[\cos\left(\frac{\lVert\vartheta\rVert}{2}\right)\mathbf{I} + \frac{2}{\lVert\vartheta\rVert}\sin\left(\frac{\lVert\vartheta\rVert}{2}\right)\mathbf{\Omega}\right]\mathbf{q_{n-1}}$
Attitude equations	Build rotation matrix $\mathbf{R}_{b2n}$ from $\mathbf{q}_n$ $\phi = \arctan\left(\frac{R_{b2n}(3,2)}{R_{b2n}(3,3)}\right)$ $\theta = \arctan\left(\frac{R_{b2n}(3,1)}{\sqrt{R_{b2n}^2(3,2)+R_{b2n}^2(3,3)}}\right)$ $\psi = \arctan\left(\frac{R_{b2n}(2,1)}{R_{b2n}(1,1)}\right)$
Velocity and position equations	$\delta\mathbf{f}^n = \mathbf{R}_{b2n}\delta\mathbf{f}^b$ $\mathbf{a}_n = \delta\mathbf{f}^n - \mathbf{g}$ $\mathbf{p}_n = \mathbf{p}_{n-1} + \mathbf{v}_{n-1}T_s + \frac{1}{2}\,\mathbf{a}_n T_s^2$ $\mathbf{v}_n = \mathbf{v}_{n-1} + \mathbf{a}_n T_s$

Table 2.1: Mechanization equations

2.3 Error model

In an INS, the error can come from a set of different sources. Four are the main ones:

Instrumentation errors: The measured data and the real one differ because of imperfections of the sensors (i.e. bias, random noise...)

Computational errors: The error is caused because of the digital processing (i.e. quantization, overflow errors...)

Alignment errors: The sensors and the platform cannot be perfectly aligned, producing an error. Furthermore, the two previous error can result in errors in the coordinate transformation process.

Environment errors: The environment cannot be perfectly modelled. Hence, it can cause errors. For instance, it is not possible to exactly predict the magnitude and direction of the effective gravity vector.

Besides, the different models and approaches followed also can introduce additional error. Therefore, one goal of an INS is to develop a system that achieves its aims in a cost-effective way. Furthermore, the initial values and the recursive propagated navigation states will never exactly be the same that the real navigation state. As the current state is calculated by considering the previous one, the error will be propagated along time, and with feedback from the own system. Therefore, it is desirable that the system is self-correcting, in the sense that the system can estimate and correct calibration, alignment and navigation-state errors. This is the main motivation for aiding sensors, as well as the reason why next chapter will present Kalman filtering. This section aims to introduce the error model that this filter will use.

Determination of linear equations that describe the error dynamics is achieved by linearization of the INS mechanization equations that have been explained in the previous section.

The linearized error equations can be written as

$$\delta\mathbf{x}' = \mathbf{F}\delta\mathbf{x} + \mathbf{G}\epsilon \tag{2.16}$$

where $\delta\mathbf{x}$ the error. In detail

$$\begin{bmatrix} \delta\mathbf{p}' \\ \delta\mathbf{v}' \\ \delta\rho' \end{bmatrix} = \begin{bmatrix} \mathbf{F}_{pp} & \mathbf{F}_{pv} & \mathbf{F}_{p\rho} \\ \mathbf{F}_{vp} & \mathbf{F}_{vv} & \mathbf{F}_{v\rho} \\ \mathbf{F}_{\rho p} & \mathbf{F}_{\rho v} & \mathbf{F}_{\rho\rho} \end{bmatrix} \begin{bmatrix} \delta\mathbf{p} \\ \delta\mathbf{v} \\ \delta\rho \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ -\hat{\mathbf{R}}_{b2n} & \mathbf{0} \\ \mathbf{0} & \hat{\mathbf{R}}_{b2n} \end{bmatrix} \begin{bmatrix} \delta\mathbf{f}^b \\ \delta\omega^b \end{bmatrix} \tag{2.17}$$

where $\delta\mathbf{p} = \mathbf{p} - \hat{\mathbf{p}}$, $\delta\mathbf{v} = \mathbf{v} - \hat{\mathbf{v}}$ and $\delta\rho$ are the position, velocity and attitude error quantities; the matrices $\mathbf{F}_{ij}$ subcomponents of the $\mathbf{F}$ matrix are $\mathbb{R}^{3x3}$ matrices about to be introduced; and $\delta\mathbf{f}^b = \Delta\mathbf{f}^b - \Delta\hat{\mathbf{f}}^b$ and $\delta\omega^b = \Delta\omega^b - \Delta\hat{\omega}^b$, specific force and angular rate measurements error from the IMU in the body frame. Elements in (2.16) can be identified by inspection in this equation.

The matrices $\mathbf{F}_{ij}$ subcomponents of the $\mathbf{F}$ matrix show how error in the different state components are propagated between them (i.e., the instantaneous variation of the position error $\delta\mathbf{p}'$ depends on the error in the position, velocity and attitude estimates by the terms in $\mathbf{F}_{pv}, \mathbf{F}_{pv}$ and $\mathbf{F}_{p\rho}$ by $\delta\mathbf{p}' = \mathbf{F}_{pp}\delta\mathbf{p} + \mathbf{F}_{pv}\delta\mathbf{v} + \mathbf{F}_{p\rho}\delta\rho$)

In most of the cases, each matrix $\mathbf{F}_{ij}$ is obtained by identifying and truncating terms in the linearization of the Taylor series expansion of the position, velocity and attitude mechanization equation around the solution provided by the by the system estimated value. A good derivation of them is provided in [2], section 11.4. It can also be found in [1], section 6.4. For a complete understanding of it, it is important to have read before the previously referred mechanization equations derivation in [1], section 6.2.

As with the mechanization equations, the estimated error must be calculated between two time instants t_n and t_{n-1}. Hence, as it was done in the previous section, by integrating between these two time instants the error can be calculated. Since the frequency of the thesis system is high and the variables discrete, the approximated result of the integration can be obtained by multiplying by the sampling period T_s. Therefore, the error between two iterations is propagated in this thesis system as

$$\delta\mathbf{x}_n = \mathbf{F}_n \delta\mathbf{x}_{n-1} + \mathbf{G}_n \epsilon \tag{2.18}$$

where

$$\mathbf{F}_n = \left[\begin{array}{ccc|ccc|ccc} 1 & 0 & 0 & T_s & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & T_s & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & T_s & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 & 0 & 0 & 0 & f_D T_s & -f_E T_s \\ 0 & 0 & 0 & 0 & 1 & 0 & -f_D T_s & 0 & f_N T_s \\ 0 & 0 & 0 & 0 & 0 & 1 & f_E T_s & -f_N T_s & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right] \tag{2.19}$$

$$\mathbf{G}_n = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ -\hat{\mathbf{R}}_{b2n} T_s & \mathbf{0} \\ \mathbf{0} & \hat{\mathbf{R}}_{b2n} T_s \end{bmatrix} \tag{2.20}$$

In the $\mathbf{F}_n$ matrix can be seen that the position error depends on the previous position error as well as the velocity error accumulated along time, as well as the velocity error depends on the previous velocity error besides the attitude error accumulated along time. Therefore, error in attitude estimation is propagated also to the position estimation, leading to a continuous increase of the error throughout iterations. However, the error in the attitude estimate is highly independent from the other two, and just depends on the estimate of the rotation matrix via the $\mathbf{G}_n$ equation.

Other foot-mounted ZUPT-aided INS implementations use a more complex error model, closer to the ones explained in the references. However, experimentally has been seen that the difference is nearly negligible and therefore this simplified approach followed in the thesis system is valid.

Chapter 3

Kalman filtering

Up to now the INS and its operation has been introduced, giving an estimate of the pedestrian state for a system's iteration. This chapter deals with the propagation of the estimates among iterations necessary for proper dead reckoning, completing the explanation of the ZUPT-aided INSs. Nevertheless, the error in the estimations is increasingly accumulated among iterations, because the current estimate depends on the previous estimated state. To solve this problem, estimation and correction methods based on Kalman filtering are implemented. Therefore, Kalman filtering is introduced in first place in this chapter. Furthermore, the foot-mounted implementation is a ZUPT-aided INS. This means that the Kalman error correction is done while a stationary state occurs, since the error committed in the states can be estimated during that instant. ZUPT-aiding is also introduced in this chapter.

3.1 Kalman filtering

The state of a discrete-time system is the smallest set of numbers known at instant n that, together with the knowledge of the system input for $l \geq n$, are enough to determine the system response for all $l > n$. As it was mentioned in the previous chapter, in a ZUPT-aided INS these are nine numbers determining position, velocity and attitude value. Throughout that chapter, it was also shown how to determine the current navigation state by using the available measurements from the IMU. However, error is increasingly accumulated which is not desired. Therefore a natural question that comes up is: is there an optimal mean of estimating the state and at the same time correcting the estimations by using the available data? The answer is provided by the Kalman filtering algorithm.

This algorithm is what is going to be derived in this section. The measurements processing in an INS can be related with a Markov chain process, which is a mathematical random process characterized as memoryless in the sense that the next state depends only on the current state and the current available data. This allows starting the Kalman filtering derivation from a state-space model, where minimum mean-square error (MMSE) problem is considered. This MMSE problem is going to be solved in parts, and eventually the Kalman filter expressions are calculated. For a complete mathematical derivation, see [1] sections 3.2., 4.2. and 4.3.

A state-space model is a mathematical representation of a physical system as a set of input u (accelerometers and gyroscopes measurements), output y (velocity) and system state x variables that are related by first-order differential equations. The equations of the discrete-time state space model are

$$\mathbf{x}(n+1) = \mathbf{F}\mathbf{x}(n) + \mathbf{G}\mathbf{u}(n) \tag{3.1}$$

$$\mathbf{y}(n) = \mathbf{H}\mathbf{x}(n) \tag{3.2}$$

where $\mathbf{F}$ is the state transition matrix, $\mathbf{G}$ is the process noise gain matrix (both of them introduced in section 2.3) and $\mathbf{H}$ is the measurement observation matrix. This model is useful in the Kalman recursion because, just with the data of the $n-1$, it allows to calculate the state in the discrete time n, by taking under consideration that it is a Markovian model.

Two additional results must be considered before starting with the Kalman filter derivation, as parts of the step by step MMSE problem we intend to solve. First of them is the Weighted Least Squares (WLS) solution. Given a set of noisy measurements $(\tilde{\mathbf{y}}_1, ..., \tilde{\mathbf{y}}_n)$, this problem looks for the optimal estimate of the unknown state vector $\hat{\mathbf{x}}_n$, where $\tilde{\mathbf{y}}_n = \mathbf{H}_n\mathbf{x}_n + \mathbf{n}_n$ and $\mathbf{n}_n$ noise.

A cost function is defined. Its mathematical expression can be motivated by the wish of minimizing some norm of the error between the measurements $\tilde{\mathbf{Y}}$ and the estimated measurements $\hat{\mathbf{Y}} = \mathbf{H}\hat{\mathbf{x}}$. Thus, the objective is to find an estimate $\hat{\mathbf{x}}$ that minimizes this cost function:

$$J_{WLS} = \frac{1}{2}\,(\tilde{\mathbf{Y}} - \mathbf{H}\hat{\mathbf{x}})^T\mathbf{W}(\tilde{\mathbf{Y}} - \mathbf{H}\hat{\mathbf{x}}) \tag{3.3}$$

The cost function is a weighted two norm where $\mathbf{W} \in \mathbb{R}^{nxn}$ is a positive definite matrix, $\tilde{\mathbf{Y}} = [\tilde{\mathbf{y}}_1, ..., \tilde{\mathbf{y}}_n]^T$ and $\mathbf{H} = [\mathbf{H}_1^T, ..., \mathbf{H}_n^T]^T$.

The fact that some measurements are more accurate than others is considered by electing $\mathbf{W} = \mathbf{R}^{-1}$, where $\mathbf{R}$ is the noise covariance matrix $E[\mathbf{n}_n\mathbf{n}_n^T]$. Operating, the solution to this WLS problem is

$$\hat{\mathbf{x}} = \mathbf{P}\mathbf{H}\mathbf{R}^{-1}\tilde{\mathbf{Y}} \tag{3.4}$$

$$\mathbf{P} = (\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H})^{-1} \tag{3.5}$$

where $\mathbf{P}$ is the states error covariance matrix $E[(\mathbf{x}_n - \hat{\mathbf{x}}_n)(\mathbf{x}_n - \hat{\mathbf{x}}_n)^T] = var(\hat{\mathbf{x}}_n)$. Thus, under this conditions, $\hat{\mathbf{x}}$ is the optimal state estimate.

The second result that needs to be considered is the solution to the Recursive Least Squares (RLS) problem. Starting from the WLS solution, where an optimal estimate of the state $\hat{\mathbf{x}}_n$ has been obtained given the current state noisy input measurements $(\tilde{\mathbf{y}}_1, ..., \tilde{\mathbf{y}}_n)$; now a new measurement $\tilde{\mathbf{y}}_{n+1}$ becomes available. This RLS problem aims to calculate the best estimate of $\mathbf{x}_{n+1}$ from the already calculated best estimate $\hat{\mathbf{x}}_n$ and the new measurement $\tilde{\mathbf{y}}_{n+1}$. Therefore it is aimed to develop a recursive estimation equation of the form

$$\hat{\mathbf{x}}_{n+1} = \hat{\mathbf{x}}_n + \mathbf{K}(\tilde{\mathbf{y}}_{n+1} - \hat{\mathbf{y}}_{n+1}) \tag{3.6}$$

where $\hat{\mathbf{y}}_{n+1} = \mathbf{H}_{n+1}\hat{\mathbf{x}}_n$ and $\mathbf{K}$ a vector gain to be determined.

The solution for this RLS problem, which uses the solution for the WLS (equations (3.4) and (3.5)) in the derivation process, is

$$\mathbf{P}_{n+1}^{-1} = \mathbf{P}_n^{-1} + (\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H})^{-1} \tag{3.7}$$

$$\mathbf{K} = \mathbf{P}_{n+1}\mathbf{H}^T\mathbf{R}^{-1} \tag{3.8}$$

$$\hat{\mathbf{x}}_{n+1} = \hat{\mathbf{x}}_n + \mathbf{K}(\tilde{\mathbf{y}}_{n+1} - \mathbf{H}\hat{\mathbf{x}}_n) \tag{3.9}$$

With these previous results, the Kalman filter derivation can be considered. This is the last extension to the least-squares estimation problem considered until now. The objective is to optimally estimate the state of a dynamic system described by a linear ordinary difference equation, which is represented by the state-space model (equations (3.1) and (3.2)):

$$\mathbf{x}(n) = \mathbf{F}(n-1)\mathbf{x}(n-1) + \mathbf{G}(n-1)\mathbf{u}(n-1) \tag{3.10}$$

where only can be measured a noisy linear combination of the system states

$$\tilde{\mathbf{y}}(n) = \mathbf{H}(n)\mathbf{x}(n) + \mathbf{n}(n) \tag{3.11}$$

It is assumed that

$$\mathbf{Q}(n) = E[\mathbf{u}(n)\mathbf{u}^T(n)] \qquad \mathbf{R}(n) = E[\mathbf{n}(n)\mathbf{n}^T(n)] \tag{3.12}$$

are known. It is also assumed that from the previous state are available $\hat{\mathbf{x}}(n-1)$ and $\mathbf{P}(n-1)$. It is aimed to produce the optimal estimate of $\hat{\mathbf{x}}(n)$ that has been corrected for the measurement $\tilde{\mathbf{y}}(n)$. The solution is achieved in two steps.

First, let's consider that the objective is to get the best estimation of $\mathbf{x}(n)$ without having available the measurement $\tilde{\mathbf{y}}$ at the instant n. Denoting this estimate $\hat{\mathbf{x}}^-(n)$, taking the expectation in both sides of equation (3.10) leads to

$$\begin{aligned} E[\mathbf{x}(n)] &= \mathbf{F}(n-1)E[\mathbf{x}(n-1)] = \mathbf{F}(n-1)\hat{\mathbf{x}}(n-1) \\ \hat{\mathbf{x}}^-(n) &= \mathbf{F}(n-1)\hat{\mathbf{x}}(n-1) \end{aligned} \tag{3.13}$$

With the variance that can be proved it is (see [1], section 3.4.4.)

$$\begin{aligned} \mathbf{P}^-(n) &= E[\delta\mathbf{x}^-(n)(\delta\mathbf{x}^-(n))^T] = \\ &= \mathbf{F}(n-1)\mathbf{P}(n-1)\mathbf{F}^T(n-1) + \mathbf{G}\mathbf{Q}(n-1)\mathbf{G}^T \end{aligned} \tag{3.14}$$

Therefore, before incorporating the new measurement $\tilde{\mathbf{y}}(n)$ two sets of information about the value of $\mathbf{x}(n)$ are available: equations (3.13) which provides the best estimate of $\mathbf{x}(n)$ obtained from the previous best estimate $\hat{\mathbf{x}}(n-1)$; and (3.11), where a new measurement becomes available and is incorporated to the system. This situation is exactly the starting point of the already introduced RLS problem. Hence the RLS solution can be applied here and the best estimate $\hat{\mathbf{x}}(n)$ corrected by the measurement $\tilde{\mathbf{y}}(n)$ becomes

$$\mathbf{P}^{-1}(n) = (\mathbf{P}^-)^{-1}(n) + (\mathbf{H}^T(n)\mathbf{R}^{-1}(n)\mathbf{H}(n))^{-1} \tag{3.15}$$

$$\mathbf{K} = \mathbf{P}(n)\mathbf{H}^T(n)\mathbf{R}^{-1}(n) \tag{3.16}$$

$$\hat{\mathbf{x}}(n) = \hat{\mathbf{x}}_n^- + \mathbf{K}(\tilde{\mathbf{y}}(n) - \mathbf{H}(n)\hat{\mathbf{x}}^-(n)) \tag{3.17}$$

With all these equations, the problem has already been solved. It can be seen that a Kalman iteration consists of two steps, a time update and a measurement update. First, the time update is given by equations (3.13) and (3.14). It provides the best estimate $\hat{\mathbf{x}}^-(n)$ of $\mathbf{x}(n)$ by considering just the previous best estimate $\hat{\mathbf{x}}(n-1)$. Afterwards the measurement correction is performed by equations (3.15), (3.16) and (3.17). In the measurement correction a new measurement $\tilde{\mathbf{y}}(n)$ has become available and the optimal estimate $\hat{\mathbf{x}}^-(n)$ can be corrected incorporating this new information, giving $\hat{\mathbf{x}}(n)$. By repeating this process recursively, the states of the system for each time instant n are optimally estimated. It is clear now that for the purpose of estimate the position of a body and correct it by using the IMU measurements which motivated this section, the Kalman filter provides an optimal solution to the problem. Nevertheless, the foot-mounted INS which this thesis deals with presents some specific features when facing the Kalman filtering that must be introduced: this is the ZUPT-aiding, which is dealt with in the next section.

Kalman equations can be adjusted in many ways for their optimal computation. Our ZUPT-aided INS uses the ones in table 3.1, which will be explained in the next section. Note that the measurement error correction specifics of this thesis system are about to be developed in the next section.

Discrete Kalman filtering equations for closed-loop ZUPT-aided INS

Initialization	$\hat{\mathbf{x}}(0) = E[\mathbf{x}(0)]$ $\mathbf{P}(0) = var(\mathbf{x}(0))$
Loop: for n = 1 to end of data	
Time update	Mechanization equations: equations in table 2.1. (this is the INS equivalent to $\hat{\mathbf{x}}^{-}(n) = \mathbf{F}(n-1)\hat{\mathbf{x}}(n-1)$) $\mathbf{P}^{-}(n) = \mathbf{F}(n)\mathbf{P}(n-1)\mathbf{F}^{T}(n) + \mathbf{GQG}^{T}$
Measurement correction (if ZUPT)	$\mathbf{K}(n) = \mathbf{P}^{-}(n)\mathbf{H}^{T}(\mathbf{HP}^{-}(n)\mathbf{H}^{T} + \mathbf{R})^{-1}$ Prediction error: $\delta\mathbf{x} = \mathbf{K}(n)(-\mathbf{v})$ (this is the INS equivalent to $\mathbf{K}(n)(\tilde{\mathbf{y}}(n) - \hat{\mathbf{y}}(n))$) Compensate internal states: equations (3.18), (3.19) (this is the INS equivalent to $\hat{\mathbf{x}}(n) = \hat{\mathbf{x}}^{-}(n) + \delta\mathbf{x}(n)$) $\mathbf{P}(n) = \mathbf{P}^{-}(n)(\mathbf{I} - \mathbf{K}(n)\mathbf{H})$

Table 3.1: Kalman filter for closed-loop ZUPT-aided INS

3.2 ZUPT-aided INS

As it has been motivated hitherto, the estimated state error increases along the time. Applications for different purposes in gait analysis and pedestrian tracking require high quality motion information. For giving a more precise idea, without any kind of correction, the estimated velocity error would increase linearly with time, and the estimated position error would increase cubically. This must be corrected somehow. When a zero velocity (ZV) situation takes place, information for reseting the velocity error is obtained (and thereby, correct the position and attitude estimate as well): this is why this is called *zero velocity update-aided (ZUPT-aided) INS*.

For normal walking situations a ZV state occurs during the stance phase of a step, that is, the instants while the shoe's heel where the INS is embedded is in contact with the ground. Fig.3.1 illustrates this. There, the INS is in the shoe sole (point 'A') instead of the heel. During a short portion of time ΔT, 'A' is not moving relative to the ground and the velocity can be considered zero.Hence, estimate correction information is available every short time periods, which has made this method a popular choice for pedestrian tracking. In this section, two questions are answered: How the zero-velocity detection is done? And, how the correction of the estimated state is done?

For a better understanding of this problem, the second question is going to be answered in first place. During the time update in the Kalman filtering process,

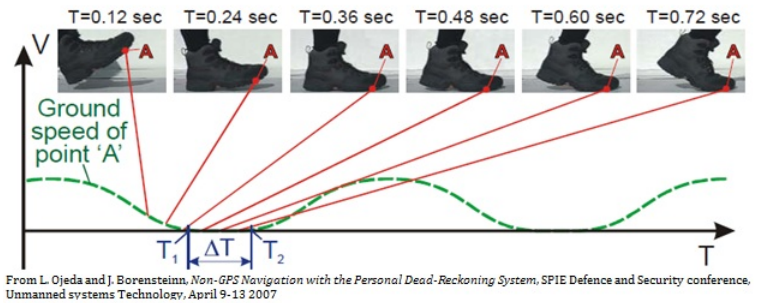

T=0.12 sec
T=0.24 sec
T=0.36 sec
T=0.48 sec
T=0.60 sec
T=0.72 sec
A
V
Ground speed of point 'A'
T1
ΔT
T2
T
From L. Ojeda and J. Borensteinn, Non-GPS Navigation with the Personal Dead-Reckoning System, SPIE Defence and Security conference, Unmanned systems Technology, April 9-13 2007

Figure 3.1: Key phases in a step. During ΔT the point A is stationary.

a new estimation of the body state is done. In the case of a ZUPT-aided INS, equation (3.13) is replaced as it can be seen in table 3.1. The estimated state $\hat{\mathbf{x}}^{-}(n)$ is obtained by combining the previous estimated state $\hat{\mathbf{x}}(n-1)$ with the measurements of the sensors in the IMU by using the equations in table 2.1, as we explained in the previous chapter. As it was stated, one of the main sources of error is the error produced by the alignment of the sensors and their uncorrect estimation of the rotation matrix from the body to the navigation frame. Therefore, the error in the estimated state increases along time. For correcting this data is needed a measurement reference which does not use the rotation matrix, which corresponds to the measurement $\tilde{\mathbf{y}}(n)$ and which provides the additional information that allows the measurement correction. This measurement reference in a ZUPT-aided INS is the stationary phase of a step.

It is intuitive to realize that during the stance phase of a step, the velocity of the foot is 0 m/s. Therefore, this is the other reference it was desired. Hence, when the stance phase of the step is detected, the current estimate of the velocity *should* be 0 m/s. If it is not, then the velocity error is the difference between the current velocity estimate and zero. By a simple substraction, this error can be corrected. Nevertheless, it has to be propagated to the position and attitude estimates. How can this be done?

An estimate of the prediction error (see table 3.1) for each component of the state vector $\delta\mathbf{x} = \mathbf{K}(n)(\tilde{\mathbf{y}}(n) - \hat{\mathbf{y}}(n))$ must be obtained. As it has been justified, the measurement of the system error is $\tilde{\mathbf{y}}(n) - \hat{\mathbf{y}}(n) = \mathbf{0} - \mathbf{v}$, which is a $\mathbb{R}^3$ vector. A way of propagating this correction to the position and covariance should be found. This is done by the Kalman gain matrix $\mathbf{K}(n)$, which is a $\mathbb{R}^{9x3}$ matrix.

Then, for position and velocity the compensated internal state is directly calculated as

$$\hat{\mathbf{x}}(n) = \hat{\mathbf{x}}^{-}(n) + \delta\mathbf{x} \tag{3.18}$$

However, to compensate the attitude error, different operations must be done. The quaternion, which represents a rotation matrix, must be corrected and from this correction, the new attitude of the system must be calculated. From the quaternion,$\hat{\mathbf{R}}_{b2n}(n)$ is built. The correction is

$$\hat{\mathbf{R}}_{b2n}(n) = (\mathbf{I}_3 - \boldsymbol{\Delta})\hat{\mathbf{R}}^{-}_{b2n}(n) \tag{3.19}$$

where

$$\boldsymbol{\Delta} = \begin{bmatrix} 0 & -\delta\mathbf{x}_{yaw} & \delta\mathbf{x}_{pitch} \\ \delta\mathbf{x}_{yaw} & 0 & -\delta\mathbf{x}_{roll} \\ -\delta\mathbf{x}_{pitch} & \delta\mathbf{x}_{roll} & 0 \end{bmatrix} \tag{3.20}$$

After this correction, the attitude components must be calculated and stored again by using equations (2.9), (2.10) and (2.11).

To end this discussion, it must be noticed that what it was called *measurement correction* while the Kalman filtering was introduced can only be done in a foot-mounted INS *when* there is a ZUPT. Otherwise, the real velocity of the foot is not known and therefore no valid correction can be done. It is straight forward to realize that the measurement update equations of the Kalman filter cannot be applied while there is not a ZV state. Therefore, along a step the error increases with the time and as soon as the ZUPT is detected, a big correction takes place, as it can be seen in Fig.3.2. This is the origin of the undesired situation that motivated this thesis.

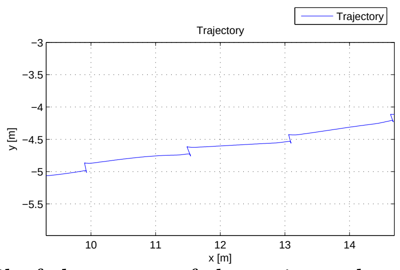


Figure 3.2: Detail of three step of the estimated trajectory for xy-axis

The second question it was posed is how the ZV detection is done. As we have seen, detecting properly when the stationary state is reached is important, since the correction to the measurement is then done. ZV detection can be a chapter on its own. An article discussing how it is done and different algorithms performance is [14]. The system used in this thesis can be configured to use four different ZV detection algorithms.The generalized likelihood rate tests (GLRT) algorithm is here succintly introduced, since it is the one which has been used in this thesis simulations and the other three are specific and simplified cases of this one. A full explanation of the GLRT criteria can be found in [6].

The GLRT method takes profit of which are the values provided by the accelerometers and gyroscopes that form the IMU. ZV detection is done by applying thresholds to the measured acceleration magnitude, gyroscope magnitude and local acceleration variance. When these three conditions are fulfilled at the same time instant, a ZUPT is decided for time n. The value of the thresholds is decided by the values that the accelerometers and gyroscopes should provide when there is a ZV state. Therefore, a good adjustement of the thresholds is necessary for the correct behaviour of the ZV state detector.

Chapter 4

Smoothing algorithms

So far the operation of ZUPT-aided INSs has been fully described. Nevertheless, the aim of this thesis is to implement a specific smoothing algorithm for this type of INSs that avoids the appearance of sharp corrections in the estimated states of the system. To our knowledge, there is not information available about such algorithms. Thus, a specific smoothing algorithm is about to be designed. This chapter introduces different general smoothing algorithms. Then, the main characteristics and implementation features of the algorithms are reviewed. Finally, decision of which method is more suitable for the implementation of a smoothed ZUPT-aided INS is taken. In the next chapter, the chosen general algorithm will be adapted for the specific features of a ZUPT-aided INS in order to get the sought smoothed ZUPT-aided INS.

4.1 Smoothing basics

The goal of a smoothing estimation process (also known as noncausal estimation process) is to determine the estimated state vector $\hat{\mathbf{x}}_{n|N},\ n < N$ such as its error variance is minimized by providing information from the future. This is the so called *smoothing problem.*

There are different approaches to solve a smoothing problem. The most known and widely used is 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 $n \in \{0, 1, ..., N\}$. The *fixed-point* approach keeps n fixed while N increases; and the *fixed-lag* approach makes n vary at the same time that N does, so $\forall n,\ n + L = N$, $L \in \mathbb{N}$ and fixed.

Under the scope of this thesis only methods belonging to the fixed-interval smoothing problem are going to be analyzed. This is done because it is considered that the structure of the signal fits with this method. Each step has a different different duration of N_i samples, with the ZUPT in the last samples. Hence, if each step is divided in segments of N_i samples, for each divided segment the fixed-interval problem can be solved. How to properly split these segments is a problem that is about to be discussed in the next chapter. If the information given by the ZUPT in the end of a step causing the sharp corrections is provided along each step calculating $\hat{\mathbf{x}}_{n|N}$ for every $n \in \{0, 1, ..., N\}$ via a smoothing filter, it is expected that the sharp corrections in the end of the step do not take place, achieving the smoothing effect sought by this thesis.

4.2 The Bryson-Frazier (BF) formula

The Bryson-Frazier (BF) formula exploits the standard state-space model shown in (3.10) and (3.11). It consists of a ”two-pass” smoothing algorithm, one pass forward and another backwards. On the forward pass, the estimated states $\hat{\mathbf{x}}(n|n)$ are computed such that on the backwards pass, where the BF formula is used, the smoothed states $\hat{\mathbf{x}}(n|N)$ are obtained.

With conditions $\forall n \in [0,\ ...,\ N]\ \exists\ \mathbf{F}_n^{-1},\ \exists\ \mathbf{P}_n^{-1}$, and $E[\mathbf{u}_n \mathbf{n}_l^T] = 0$, the BF formula is

$$
\begin{aligned}
&for\ n = N-1,\ ...,\ 0\\
&\qquad \mathbf{A}(n) = \mathbf{F}(n)[\mathbf{I} - \mathbf{K}(n)\ \mathbf{H}^T(n)]\\
&\qquad \mathbf{r}(n-1) = \mathbf{A}^T(n)\mathbf{r}(n) + \mathbf{H}^T(n)\mathbf{Q}^{-1}(n)\mathbf{e}(n)\\
&\qquad \mathbf{R}(n-1) = \mathbf{A}^T(n)\mathbf{R}(n)\mathbf{A}(n) + \mathbf{H}^T(n)\mathbf{Q}^{-1}(n)\mathbf{H}(n)\\
&\qquad \hat{\mathbf{x}}(n|N) = \hat{\mathbf{x}}(n|n-1) + \mathbf{P}(n|n-1)\mathbf{r}(n-1)\\
&\qquad \mathbf{P}(n|N) = \mathbf{P}(n|n-1) - \mathbf{P}(n|n-1)\mathbf{R}(n-1)\mathbf{P}(n|n-1)
\end{aligned}
\tag{4.1}
$$

with initial conditions $\mathbf{r}(N) = 0$ and $\mathbf{R}(N) = \mathbf{0}$; with the innovation $\mathbf{e}(n) = \tilde{\mathbf{y}}(n) - \hat{\mathbf{y}}(n) = \tilde{\mathbf{y}}(n) - \mathbf{H}(n)\hat{\mathbf{x}}(n|n-1)$; and with $\mathbf{A}(n)$, $\mathbf{r}(n-1)$ and $\mathbf{R}(n-1)$ understood as auxiliar variables to simplify notation.

In the considered ZUPT-aided INS, $\hat{\mathbf{x}}(n|n-1)$ and $\mathbf{P}(n|n-1)$ correspond to the time update of the forward Kalman filter $\hat{\mathbf{x}}^-(n)$ and $\mathbf{P}^-(n)$, whereas $\hat{\mathbf{x}}(n|n)$ and $\mathbf{P}(n|n)$ correspond to the zero-velocity update which, in the case that a ZV state is not detected, are identical to the time update $\hat{\mathbf{x}}(n|n-1) = \hat{\mathbf{x}}(n|n)$. However, some special treatment of the data in the forward pass is required to achieve the conditions of a *fixed-interval* smoothing problem. This processing is discussed in the next chapter.

4.2.1 Bryson-Frazier formula derivation

In this section, it is shown where the BF formulas come from. For a complete derivation of this formula, see [7], sections 10.1. and 10.2.

First is obtained a general smoothing formula which takes profit of the available information from the past and the future time instants. By operating in this formula different solutions for the smoothing problem can be derived, among them the BF formula.

The basic formula for estimation given the assumed uncorrelated innovations process $\mathbf{e}_n = \tilde{\mathbf{y}}_n - \hat{\mathbf{y}}_n$ can be obtained (see [7], section 4.2.) by considering that the error between the real value and the estimated value from the innovation should be orthogonal to the innovation. That is, $E[(\mathbf{x}_n - \hat{\mathbf{x}}_{n|N})\mathbf{e}_n^T] = 0$. The basic estimation formula is:

$$\hat{\mathbf{x}}_n = \sum_{l=0}^{n} \langle \mathbf{x}_n, \mathbf{e}_l \rangle \mathbf{R}_{e,l}^{-1} \mathbf{e}_l \tag{4.2}$$

This estimation formula expressed for a fixed-interval smoothing problem where measurements from $n = 0, ..., N$ are known, is

$$\hat{\mathbf{x}}_{n|N} = \sum_{l=0}^{N} \langle \mathbf{x}_n, \mathbf{e}_l \rangle \mathbf{R}_{e,l}^{-1} \mathbf{e}_l = \hat{\mathbf{x}}_n + \sum_{l=n}^{N} \langle \mathbf{x}_n, \mathbf{e}_l \rangle \mathbf{R}_{e,l}^{-1} \mathbf{e}_l \tag{4.3}$$

with $\mathbf{R}_{e,l} = E[\mathbf{e}_l\mathbf{e}_l^T] = \mathbf{R}_l + \mathbf{H}_l\mathbf{P}_l\mathbf{H}_l^T$; and denoting $\langle \mathbf{x}_n, \mathbf{e}_l \rangle \equiv E[\mathbf{x}_n\mathbf{e}_l^T]$. Along this chapter this notation is going to be used for easier visualization.

The previous equation is the sought general smoothing formula. From this equation, it is clear that the estimated $\hat{\mathbf{x}}_{n|N}$ is obtained by addition of the optimal forward estimation until instant n, $\hat{\mathbf{x}}_n$ (obtained by Kalman filtering); plus a backwards estimation from N, $N-1$, ..., n that considers how much innovations in future time instants influence in the current estimate.

What it is looked for is a recursion for the optimal estimation of this backwards value. In order to get it, the summation in (4.3) must be expressed in a recursive way that does not require to calculate the whole summation for each n. The only term in the summation that depends on different time instants n and l is $\langle \mathbf{x}_n, \mathbf{e}_l \rangle$. Therefore, this term is the one which has to be calculated recursively. Considering that

$$\langle \mathbf{x}_n, \mathbf{e}_l \rangle = \langle \mathbf{x}_n, \tilde{\mathbf{y}}_l - \hat{\mathbf{y}}_l \rangle = \langle \mathbf{x}_n, \mathbf{x}_l \rangle \mathbf{H}_l^T = \mathbf{P}_{nl}\mathbf{H}_l^T \tag{4.4}$$

where $\tilde{\mathbf{y}}_l = \mathbf{H}_l\mathbf{x}_l + \mathbf{n}_l$ has been used, (4.3) can be written as

$$\hat{\mathbf{x}}_{n|N} = \hat{\mathbf{x}}_n + \sum_{l=n}^{N} \mathbf{P}_{nl}\mathbf{H}_l^T\mathbf{R}_{e,l}^{-1}\mathbf{e}_l \tag{4.5}$$

From this equation, it can be shown that $\mathbf{P}_{nl} = \mathbf{P}_n \prod_{l=n}^{N-1}(\mathbf{F}_l - \mathbf{K}_l\mathbf{H}_l)$, allowing to find the looked recursion by expressing $\mathbf{P}_{nl}$ as

$$\mathbf{P}_{nl} = \mathbf{P}_n(\mathbf{F}_n - \mathbf{K}_n\mathbf{H}_n) \prod_{l=n+1}^{N-1} (\mathbf{F}_l - \mathbf{K}_l\mathbf{H}_l) \tag{4.6}$$

where the product sequence term is known from the previous iteration and therefore getting the searched recursion. Developing the terms in the equation (4.6) give the BF recursion general

$$\hat{\mathbf{x}}_{n|N} = \hat{\mathbf{x}}_n + \mathbf{P}_n\mathbf{r}_{n|N} \tag{4.7}$$

where $\mathbf{r}_{n|N}$ is the smoothing effect calculated from posterior samples, which can be recursively propagated backwards and which its mathematical expression can be found in (4.1). Identifying, this equation has the same terms that the $\hat{\mathbf{x}}_{n|N}$ equation had in (4.1) and allows to calculate the n^{th} iteration coefficients from the previous one $n+1$. An equivalent process can be done for getting $\mathbf{P}_{n|N}$.

4.3 The Rauch-Tung-Striebel (RTS) formula

As the BF algorithm, the Rauch-Tung-Striebel (RTS) formula also takes advantage of the state-space structure. If $\mathbf{x}_n$, $\hat{\mathbf{x}}_n$ and $\mathbf{r}_{n|N}$ follow recursive equations that allow the calculation of $\hat{\mathbf{x}}_{n|N}$, it seems logical that $\hat{\mathbf{x}}_{n|N}$ also can follow directly some kind of recursion. This is the idea that lies under the RTS recursions: to get a recursive formula that allows calculating $\hat{\mathbf{x}}_{n|N}$ from $\hat{\mathbf{x}}_{n+1|N}$.

As well as with the BF smoothing algorithm, the RTS algorithm also consists of a "two-pass" smoothing algorithm. It is again in the backwards pass where the RTS formula is applied. The forward pass will be dealt in the next chapter.

There are several RTS recursions forms. In the next lines, it can be found the so called *original RTS formula*, which compared with the other RTS models presents the implementation benefit that it just requires for every iteration the inversion of the $\mathbf{P}_{n+1|n}$ matrix. The other methods require for every iteration both, the inversion of the $\mathbf{P}_{n+1|n}$ and the $\mathbf{F}_n$ matrices which is, of course, computationally more expensive.

With conditions $\forall n \in [0,\ ...,\ N]\ \exists\ \mathbf{F}_n^{-1}$, $E[\mathbf{u}_n \mathbf{n}_l^T] = 0$ and $\mathbf{P}_n > 0$, the RTS formula is

$$
\begin{aligned}
&for\ n = N-1,\ ...,\ 0\\
&\qquad \mathbf{A}(n) = \mathbf{P}(n|n)\mathbf{F}^T\mathbf{P}^{-1}(n+1|n)\\
&\qquad \hat{\mathbf{x}}(n|N) = \hat{\mathbf{x}}(n|n) + \mathbf{A}(n)[\hat{\mathbf{x}}(n+1|N) - \hat{\mathbf{x}}(n+1|n)]\\
&\qquad \mathbf{P}(n|N) = \mathbf{P}(n|n) + \mathbf{A}(n)[\mathbf{P}(n+1|N) - \mathbf{P}(n+1|n)]\mathbf{A}^T(n)
\end{aligned}
\tag{4.8}
$$

with the initial conditions $\hat{\mathbf{x}}(N|N), \mathbf{P}(N|N)$ provided by the forward Kalman filter; and where for clarity $\hat{\mathbf{x}}(n)$ has been written $\hat{\mathbf{x}}(n|n)$

Note again that in the ZUPT-aided INS, $\hat{\mathbf{x}}(n|n-1)$ and $\mathbf{P}(n|n-1)$ correspond to the time update of the forward Kalman filter $\hat{\mathbf{x}}^-(n)$ and $\mathbf{P}^-(n)$, whereas $\hat{\mathbf{x}}(n|n)$ and $\mathbf{P}(n|n-1)$ correspond to the zero-velocity update which, in the case that a ZV state is not detected, are identical to the time update.

4.3.1 Rauch-Tung-Striebel formula derivation

As it has been done for the BF formula in section 4.2.1, an overall idea of where this formula comes from is going to be provided. For a complete derivation, see [7] sections 10.2. and 10.3.

It is useful to remark that the aim of the Rauch-Tung-Striebel formula was, as stated in the previous section, to get a recursion which gives $\hat{\mathbf{x}}_{n|N}$ dependent on the $\hat{\mathbf{x}}_{n+1|N}$ term, which is known from the previous iteration.

Starting from (4.5) and (4.6), and taking profit of the iterative structure they have, those equations can be written for the time instant $n+1$ as

$$
\begin{aligned}
&\mathbf{P}_n\mathbf{F}_n^T\mathbf{P}_{n+1|n}^{-1}\hat{\mathbf{x}}_{n+1|N} =\\
&= \hat{\mathbf{x}}_{n|N} + (\mathbf{P}_n\mathbf{F}_n^T\mathbf{P}_{n+1|n}^{-1}\mathbf{F}_n - \mathbf{I})\hat{\mathbf{x}}_{n|n-1} + (\mathbf{P}_n\mathbf{F}_n^T\mathbf{P}_{n+1|n}^{-1}\mathbf{F}_n - \mathbf{I})\mathbf{P}_n\mathbf{H}_n^T\mathbf{R}_{e,n}^{-1}\mathbf{e}_n
\end{aligned}
\tag{4.9}
$$

where $\mathbf{e}_n$ is the innovation. Denoting $\mathbf{P}_n\mathbf{F}_n^T\mathbf{P}_{n+1|n}^{-1}$ as $\mathbf{A}_n$, and grouping terms:

$$
\hat{\mathbf{x}}_{n|N} = \mathbf{A}_n\hat{\mathbf{x}}_{n+1|N} + (\mathbf{I} - \mathbf{A}_n\mathbf{F}_n)(\hat{\mathbf{x}}_{n|n-1} + \mathbf{P}_n\mathbf{H}_n^T\mathbf{R}_{e,n}^{-1}\mathbf{e}_n) \tag{4.10}
$$

and by noting that $\hat{\mathbf{x}}_n = \hat{\mathbf{x}}_{n|n-1} + \mathbf{P}_n\mathbf{H}_n^T\mathbf{R}_{e,n}^{-1}\mathbf{e}_n$ and $\hat{\mathbf{x}}_{n+1|n} = \mathbf{F}_n\hat{\mathbf{x}}_n$ the previous expression can be transformed as follows

$$
\begin{aligned}
\hat{\mathbf{x}}_{n|N} &= \mathbf{A}_n\hat{\mathbf{x}}_{n+1|N} + (\mathbf{I} - \mathbf{A}_n\mathbf{F}_n)\hat{\mathbf{x}}_n =\\
&= \hat{\mathbf{x}}_n + \mathbf{A}_n(\hat{\mathbf{x}}_{n+1|N} - \hat{\mathbf{x}}_{n+1|n})
\end{aligned}
\tag{4.11}
$$

which is the RTS formula that has been presented in (4.8), where $\hat{\mathbf{x}}_{n|N}$ is expressed in terms of $\hat{\mathbf{x}}_{n+1|N}$. An equivalent process can be done for the derivation of the RTS $\mathbf{P}_{n|N}$ formula.

4.4 Two filter formulas

Both the BF and the RTS smoothing formulas were based on expressing $\hat{\mathbf{x}}_{n|N}$ in terms of a sum with both a forwards innovations term and a recursion for the backwards estimation. However, this is not the only approach that can be followed. Since the time interval which is processed is fixed, direction of the time is not important, because all the observations $\tilde{\mathbf{y}}_0, ..., \tilde{\mathbf{y}}_N$ obtained from the INS are available and are independent. Therefore, it should be possible to process data backwards (from $\tilde{\mathbf{y}}_N$ to $\tilde{\mathbf{y}}_0$). This is the basic idea of the two filter formulas: by properly combining data processing forwards and backwards, different *two-filter formulas* can be obtained. Through this section an overview of different existing sets of two-filter formulas is given.

Two general options can be considered when combining data:

- To combine the forwards *predicted* estimator $\hat{\mathbf{x}}_{n|n-1}$ with the backwards *filtered* estimator $\hat{\mathbf{x}}^b_{n|n}$, which is the l.l.m.s.e. of $\mathbf{x}_n$ given the data measurements $\tilde{\mathbf{y}}$ from $\tilde{\mathbf{y}}_n,\ \tilde{\mathbf{y}}_{n+1},\ ...,\ \tilde{\mathbf{y}}_N$.
- To combine the forwards *filtered* estimator $\hat{\mathbf{x}}_n (= \hat{\mathbf{x}}_{n|n})$ with the backwards *predicted* estimator $\hat{\mathbf{x}}^b_n (= \hat{\mathbf{x}}^b_{n|n+1})$, which is the l.l.m.s.e. of $\mathbf{x}_n$ given the data measurements $\tilde{\mathbf{y}}_{n+1},\ \tilde{\mathbf{y}}_{n+2},\ ...,\ \tilde{\mathbf{y}}_N$.

It should be noticed that now the backwards estimation is calculated by using the measurements $\tilde{\mathbf{y}}$, not the innovation $\mathbf{e}$. This shows clearly the just explained two-filter smoothing basic idea. For the sake of simplicity, the second option is going to be followed in the further expressions of this section of the thesis. Formulas concerning the first option can be found in [7], section 10.4.

4.4.1 General two filter formula

With conditions $\langle \tilde{\mathbf{y}}_n, \mathbf{n}_l \rangle = 0$ and $\exists\ \mathbf{F}_n^{-1}$, the general two filter formula is

$$\begin{aligned} \hat{\mathbf{x}}_{n|N} &= \mathbf{P}_{n|N}\Big(\mathbf{P}_n^{-1}\hat{\mathbf{x}}_n + (\mathbf{P}^b_n)^{-1}\hat{\mathbf{x}}^b_n\Big) \\ \mathbf{P}_{n|N} &= \Big(\mathbf{P}_n^{-1} + (\mathbf{P}^b_n)^{-1} - \mathbf{\Pi}_n^{-1}\Big)^{-1} \end{aligned} \tag{4.12}$$

where $\mathbf{\Pi}_n = \langle \mathbf{x}_n, \mathbf{x}_n \rangle$

To prove it, we begin by defining a state-space model for the two-filter formulas, which is based on the standard state-space model (3.10) and (3.11):

$$\mathbf{x}_{n+1} = \mathbf{F}_n\mathbf{x}_n + \mathbf{G}_n\mathbf{u}_n \qquad \mathbf{y}_n = \mathbf{H}_n\mathbf{x}_n + \mathbf{n}_n \qquad \textit{forwards} \tag{4.13}$$

$$\mathbf{x}_n = \mathbf{F}^b_{n+1}\mathbf{x}_{n+1} + \mathbf{u}^b_{n+1} \qquad \mathbf{y}_n = \mathbf{H}_n\mathbf{x}_n + \mathbf{n}^b_n \qquad \textit{backwards} \tag{4.14}$$

The idea, using this state space model as a start point, is to calculate the forwards filtered estimator $\hat{\mathbf{x}}_n$ by getting a recursion that allows to calculate it. In the same way, it is aimed to calculate the backwards predicted estimator $\hat{\mathbf{x}}^b_n$ by getting another recursive formula. As it is shown in [7], section 10.4.3., both recursive formulas can be determined so an estimate of both $\hat{\mathbf{x}}_n$ and $\hat{\mathbf{x}}^b_n$ is obtained.

On the other side, let's assume that variables $\{\mathbf{n}_n, \mathbf{n}^b_n, \mathbf{x}_n\}$ in the state space model are mutually uncorrelated zero-mean random variables. Then, the estimations from past and future can be combined as follows

$$\mathbf{P}^{-1}\hat{\mathbf{x}} = (\mathbf{P}^{forw})^{-1}\hat{\mathbf{x}}^{forw} + (\mathbf{P}^b)^{-1}\hat{\mathbf{x}}^b \tag{4.15}$$

with $\mathbf{P}^{-1} = (\mathbf{P}^{forw})^{-1} + (\mathbf{P}^{b})^{-1} - \mathbf{R}^{-1}$

Thus, by applying (4.15), both estimated values $\hat{\mathbf{x}}_n$ and $\hat{\mathbf{x}}_n^b$ can be combined giving the general two-filter formula equation (4.12). The full derivation can be found in [7] section 10.4.3.

4.4.2 Fraser-Potter and Mayne formulas

With conditions $\langle \mathbf{u}_n, \mathbf{n}_l \rangle = 0$, $\exists\ \mathbf{F}_n^{-1}$ and $\mathbf{P}_0 > 0$, and with boundary condition $\mathbf{P}_{N+1}^b = \infty \cdot \mathbf{I}$, the Fraser-Potter formula is

$$\begin{aligned} \hat{\mathbf{x}}_{n|N} &= \mathbf{P}_{n|N}\left(\mathbf{P}_n^{-1}\hat{\mathbf{x}}_n + (\mathbf{P}_n^b)^{-1}\hat{\mathbf{x}}_n^b\right) \\ \mathbf{P}_{n|N} &= \left(\mathbf{P}_n^{-1} + (\mathbf{P}_n^b)^{-1}\right)^{-1} \end{aligned} \tag{4.16}$$

which essentially consists of the general two-filter formula by choosing $\mathbf{P}_n = \infty \cdot \mathbf{I}$, so as the terms in $\mathbf{P}_n^{-1}$ can be neglected and the final expression, simplified. This is possible due to the additional degree of freedom that it is obtained from the backwards computation, solving the problem of computing $\langle \mathbf{x}_n, \mathbf{x}_n \rangle$, that should not easily be known. However, this assumption requires an additional discussion involving Markovian states models that can be found in [15] and show that this formula is impractical.

Due to the fact that the previous formula was impractical, a different set of equations was proposed: the Mayne formula. Here, in order to avoid the infinities, $(\mathbf{P}_n^b)^{-1}$ and $(\mathbf{P}_n^b)^{-1}\hat{\mathbf{x}}_i^b$ are also propagated iteration by iteration.

With conditions $\exists\ \mathbf{F}_n^{-1}$, $\exists\ \mathbf{P}_n^{-1}$, $\langle \mathbf{u}_n, \mathbf{n}_l \rangle = 0$ and $\mathbf{P}_0 > 0$, the Mayne formula is

$$\begin{aligned} \hat{\mathbf{x}}_{n|N} &= \mathbf{P}_{n|N}(\mathbf{P}_n^{-1}\hat{\mathbf{x}}_n + \mathbf{z}_n^b) \\ \mathbf{z}_n^b &= \mathbf{F}_n^T(\mathbf{I} + \mathbf{L}_{n+1}^b \mathbf{G}_n \mathbf{Q}_n \mathbf{G}_n^T)^{-1}\mathbf{z}_{n+1}^b + \mathbf{H}_n^T \mathbf{R}_n^{-1}\mathbf{y}_n \\ \mathbf{P}_{n|N}^{-1} &= \mathbf{P}_n^{-1} + (\mathbf{L}_n^b)^{-1} \\ \mathbf{L}_n^b &= \mathbf{F}_n^T(\mathbf{I} + \mathbf{L}_{n+1}^b \mathbf{G}_n \mathbf{Q}_n \mathbf{G}_n^T)^{-1}\mathbf{L}_{n+1}^b \mathbf{F}_n + \mathbf{H}_n^T \mathbf{R}_n^{-1}\mathbf{H}_n \end{aligned} \tag{4.17}$$

with boundary condition $\mathbf{L}_{N+1}^b$, and where it can be identified $\mathbf{z}_n^b = (\mathbf{P}_n^b)^{-1}$ and $\mathbf{L}_n^b = (\mathbf{P}_n^b)^{-1}\hat{\mathbf{x}}_i^b$

4.5 Other smoothing estimators

Some other approaches can be followed in behalf of solving the smoothing problem. For example, the assumption (already made for the Kalman filter derivation) that $\mathbf{R}_n > 0$ can be used for smoothing estimation, leading to the so called Hamiltonian equations, from which the BF and the RTS recursions can be obtained (and vice versa, which provides an easy way of demonstrating the Hamiltonian equations)

Assuming that $\exists\ \mathbf{R}_n^{-1}$, and $\langle \mathbf{u}_n, \mathbf{n}_l \rangle = 0$ and $\mathbf{R}_n > 0$, the Hamiltonian equations are

$$\begin{bmatrix} \hat{\mathbf{x}}_{n+1|N} \\ \mathbf{r}_{n|N} \end{bmatrix} = \begin{bmatrix} \mathbf{F}_n & \mathbf{G}_n \mathbf{Q}_n \mathbf{G}_n^T \\ -\mathbf{H}_n \mathbf{R}_n^{-1} \mathbf{H}_n^T & \mathbf{F}_n \end{bmatrix} \begin{bmatrix} \hat{\mathbf{x}}_{n|N} \\ \mathbf{r}_{n+1|N} \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \mathbf{H}_n^T \mathbf{R}_n^{-1} \end{bmatrix} \mathbf{y}_n \tag{4.18}$$

where the boundary conditions are $\hat{\mathbf{x}}_{0|N} = \mathbf{P}_{0|N}\mathbf{r}_{n|N}$, $\mathbf{r}_{N+1|N} = 0$. As these conditions are one for $n = 0$ and other for $n = N + 1$, problems for finding a recursion arise. Methods to getting it are, for example, obtaining the BF and RTS formulas from here, or using the so called *sweep methods*, consisting of triangularize the matrix expression achieving a solution. For further details, see [8]

There are other strategies for solving the smoothing problem, but most of them are based on the already presented BF, RTS and two-filter formulas, presenting slightly different modifications, also dependant on the application. For example, it can be hesitated to increase the stability of the system by looking for methods that avoid the matrix inversions required. In the end, they are different kinds of modifications on the same equations that have been shown in this chapter.

4.6 Algorithms comparison and choice

Throughout the last sections, the most relevant smoothing algorithms have been reviewed in order to decide which one is going to be used for the implementation of a smoothed ZUPT-aided INS. Some criteria for the election of an smoothing algorithm must be fixed and analysed.

Moreover, not an extensive and well documented work has been performed in smoothing filtering for ZUPT-aided INS, leading to a lack of sources where basing the smoothing algorithm's choice. Since the implementation of a step-wise smoothing of a ZUPT-aided INS have some implementation specifics that must be considered, as it will be shown in the next chapter, on the following essentially one criteria is to be followed: the simplicity of the algorithm. This will allow to focus on the specific problems of the implementation of a smoothing filter for a ZUPT-aided INS.

As the system is to be runned in Matlab, memory requirements are not considered a constraint for the time being. In the same way, real-time issues are not fully considered because the system is runned when all the sensor measurements have been taken and therefore are available. Nevertheless, it should be noticed that this thesis aims to implement a *step-wise* smoothing ZUPT-aided INS. In a short period of time the smoothing algorithm about to be proposed will be implemented for near real-time applications. Therefore, this issue is not completely neglected on the following.

Considering all these aspects, the smoothing algorithm choice can be done. Due to the implementation problems that the two-filter formulas have, they are discarded. By checking the algorithms left, (4.1) and (4.8); it is clear that the RTS algorithm presents a simpler implementation. Further, the BF recursions require the inversion of both, the $\mathbf{P}_n$ and the $\mathbf{F}_n$ matrices, whereas the RTS recursions require only the inversion of the $\mathbf{P}_n$ matrix. Beyond this, the RTS algorithm does not require the innovation vector $\mathbf{e}_n$, whereas BF does. This vector is not directly calculated in the available version of the system.

Therefore, the implementation of the RTS method is straight-forward and presents some advantages respect to the BF. Therefore, the RTS is the algorithm chosen for the smoothing filter implementation for a ZUPT-aided INS that is the aim of this thesis. Even though it is not a constraint in this thesis, from equation 4.8 it must be noticed that matrices $\mathbf{P}(n|n), \mathbf{P}(n|n-1)$ and vectors $\hat{\mathbf{x}}(n|n), \hat{\mathbf{x}}(n|n-1)$ must be stored in memory in the pass forwards since the pass backwards requires them, increasing considerably the memory requirements. Besides, computational complexity is the same in the pass backwards than in the pass forwards.

Chapter 5

Smoothed ZUPT-aided INS

Up to now the ZUPT-aided INSs on one side, and smoothing filters algorithms on the other have been introduced. They must be combined for obtaining the sought smoothed ZUPT-aided INS. Nevertheless, combining both parts implies considering some issues that must be solved. This chapter deals with these issues, where they appear as well as which is the solution proposed for the implementation in this thesis. First point introduces why is required an open loop implementation for the ZUPT-aided INS and explains it. The second section of this chapter deals with the data segmentation for creating a fixed-interval smoothing problem. Since the steps are not uniformly spaced, a varying-lag rule for the segmentation is required and therefore introduced.

5.1 Open-loop implementation

Relating the RTS formula (4.8) with the previously derived Kalman filter recursion (table 3.1) is not a straight forward process because the estimated states $\hat{\mathbf{x}}_n$ cannot be directly linked to the RTS formula components. This is due to the fact that the implemented Kalman filter is a closed-loop version of the filter. This means that every iteration the filter provides an estimate of the whole state $\hat{\mathbf{x}}_n$ via what we called compensation of the internal states, equations (3.18), (3.19). Direct smoothing over $\hat{\mathbf{x}}_n$ is not possible since the available covariance matrix is the *error* covariance matrix $\mathbf{P}$, not the states covariance matrix.

For direct identification with the terms in the RTS formula, opening the Kalman filter loop is required. This allows to propagate on one side an estimate of the state $\hat{\mathbf{x}}^-$ using the measurements taken by the IMU and the mechanization equations (table 2.1). On the other side the via Kalman filtering estimated error $\delta\hat{\mathbf{x}}$ is propagated. The estimated error $\delta\hat{\mathbf{x}}$ contains the ZUPT corrections to be smoothed. Since the error covariance matrix $\mathbf{P}$ is available, smoothing can be done over $\delta\hat{\mathbf{x}}$. Thus, Kalman filtering equations are modified to the ones in table 5.1, where an open-loop implementation of the ZUPT-aided INS is found.

The time update follows the standard Kalman propagation equation

$$\delta\hat{\mathbf{x}}^-(n) = \mathbf{F}(n)\delta\hat{\mathbf{x}}(n-1) \tag{5.1}$$

where the estimated error depends on the previous iteration estimated error.

In the same way, the estimated state $\hat{\mathbf{x}}_n^-$ is propagated during a time update by following the same mechanization equations that have been used so far (see table

2.1). However, owing to the open-loop nature of the implementation, during the measurement update no feedback is provided to $\hat{\mathbf{x}}_n^-$.

Instead of doing the feedback over $\hat{\mathbf{x}}_n^-$, during the measurement update the estimated error $\delta\hat{\mathbf{x}}^-(n)$ is corrected. During a ZUPT the *error* is the difference between the estimated velocity and zero, and that is the correction that was applied in the closed-loop formulas. This correction is the amount that the *estimated* error $\delta\hat{\mathbf{x}}^-(n)$ should decrease during the ZUPT. Thus, the measurement update equation for the open-loop becomes:

$$\delta\hat{\mathbf{x}} = \delta\hat{\mathbf{x}}^-(n) + \mathbf{K}(n)(-\mathbf{v}) \tag{5.2}$$

Therefore, an estimated state $\hat{\mathbf{x}}_n^-$ is propagated without being corrected, as well as an estimate of the error $\delta\hat{\mathbf{x}}_n$. After data segmentation by applying the criteria about to be explained in the next section, the smoothing filter can be applied to the segment of estimated error states $\delta\hat{\mathbf{x}}$. This segment is the one which has the drastic corrections and that can be directly identified with the terms in the RTS formula (4.8). After smoothing the estimated error $\delta\hat{\mathbf{x}}$, compensation of the internal states (equations (3.18), (3.19)) is done for each sample in the segment and the final estimated state $\hat{\mathbf{x}}_n$ is calculated for each sample. However, before introducing the complete equations for a smoothed ZUPT-aided INS, next section will introduce the data segmentation issue.

Discrete Kalman filtering equations for open-loop ZUPT-aided INS

Initialization	$\hat{\mathbf{x}}(0) = E[\mathbf{x}(0)]$; $\delta\hat{\mathbf{x}}(0) = 0$; $\mathbf{P}(0) = var(\mathbf{x}(0))$
Loop: for n = 1 to end of data	
Time update	Mechaniz. equations for $\hat{\mathbf{x}}^-(n)$: equations in table 2.1. $\delta\hat{\mathbf{x}}^-(n) = \mathbf{F}(n)\delta\hat{\mathbf{x}}(n-1)$ $\mathbf{P}^-(n) = \mathbf{F}(n)\mathbf{P}(n-1)\mathbf{F}^T(n) + \mathbf{GQG}^T$
Measurement correction (if ZUPT)	$\mathbf{K}(n) = \mathbf{P}^-(n)\mathbf{H}^T(\mathbf{HP}^-(n)\mathbf{H}^T + \mathbf{R})^{-1}$ Prediction error: $\delta\hat{\mathbf{x}} = \delta\hat{\mathbf{x}}^-(n) + \mathbf{K}(n)(-\mathbf{v})$ (this is the INS equivalent to $\mathbf{K}(n)(\tilde{\mathbf{y}}(n) - \hat{\mathbf{y}}(n))$ $\mathbf{P}(n) = \mathbf{P}^-(n)(\mathbf{I} - \mathbf{K}(n)\mathbf{H})$
After loop	For each sample: compensate internal states: equations (3.18), (3.19) (this is the INS equivalent to $\hat{\mathbf{x}}(n) = \hat{\mathbf{x}}^-(n) + \delta\mathbf{x}(n)$)

Table 5.1: Kalman filter for open-loop ZUPT-aided INS

5.2 Data segmentation

In the previous chapter we stated that a *fixed-interval* smoothing problem is to be solved. This problem takes a fixed segment of measurements $\{\tilde{\mathbf{y}}_0, \tilde{\mathbf{y}}_1, ..., \tilde{\mathbf{y}}_N\}$ and calculates $\hat{\mathbf{x}}_{n|N}$ for each sample in the segement. This way of focusing the smoothing problem was judged to fit very well with the purpose of the system, since every segment can be associated with a step of the person and the information provided by the ZUPT, giving the sharp corrections, can be made available along the step. Additionally, dividing the data in short duration segments, as it is a step, will allow the implementation of a smoothed ZUPT-aided INS in near real-time.

Accordingly, for the achievement of the fixed-interval smoothing problem conditions data must be segmented. Nevertheless, some problems arise. It is easy to notice that two different steps do not last for the same number of samples. Therefore, some kind of segmentation rule which can consider the different duration of each step is required for proper data segmentation. The next paragraphs motivate the one decided for this thesis.

One intuitive thought is to take profit of the detected ZV intervals and divide the step with them, because there is one stance phase of the foot per step. However, this is not possible, at least not directly. As it was introduced in section 3.2, ZV states detection is done by applying thresholds to the measured acceleration magnitude, gyroscope magnitude and/or local acceleration variance. When the measurements are in between all these thresholds (logic AND), a ZV state is decided. Even if some methods to avoid the wrong decission of a ZV state are applied, as calculate the average of neighbouring samples, experimentally is seen that during a stance phase of a step several non-ZV segments are decided when they should not. This is illustrated in the ZUPT vector in Fig. 5.1, where around the time instant 1.4 sec., there is a wrong decision of no-ZUPT. The higher the velocity of the pedestrian, the more often this occurs. Therefore, using directly the ZV decission as a method of data segmentation is not possible, since a step cannot be directly identified by the sequence *"no ZV state detected - ZV state detected = 1 step"*.

However, some aspects of the ZV detection idea can be exploited for getting the segmentation rule that is used in this thesis, where covariance and timing thresholds are used. First, let's analyze the velocity error covariance evolution along a step. From Fig. 5.1, the velocity error covariance of a step decreases drastically as soon as a ZUPT occurs. When the foot reaches a steady state, correction by taking external measurements is done and therefore the dependencies among the different variables decrease and are kept into a minimum while the ZUPTs take place. As soon as the ZUPTs end, the covariance monotically increases again.

From the analyzed evolution of the velocity covariance segment analyzed in the previous paragraph, it is decided to fix a covariance threshold t whose crossing will decide the segmentation. Consequently, a natural question comes up: where and how fixing the evaluation of this velocity covariance threshold t?

Three different segmentation points are considered. These are marked in Fig.5.2.

- A, point where the velocity covariance is decreasing drastically because of the ZUPT.
- B, point where the first discontinuity in the ZUPT vector is detected for the stationary phase of the step.
- C, point where the stationary phase of the step ends.

Unfortunately, none of this points presents adequate characteristics for being the point where deciding the segmentation:

- Segmentation in A, where the bigger corrections are being done to the error covariance and states value. Since the system has not converged yet, the proper smoothing of the covariances and the states is not possible because the information provided by the ZUPT is still not fully available.
- Segmentation in B, where the first ZV-state detection in the step ends, implies the same problem than A if the covariance has not converged yet, as well as a hard characterization of the system since this point is randomly

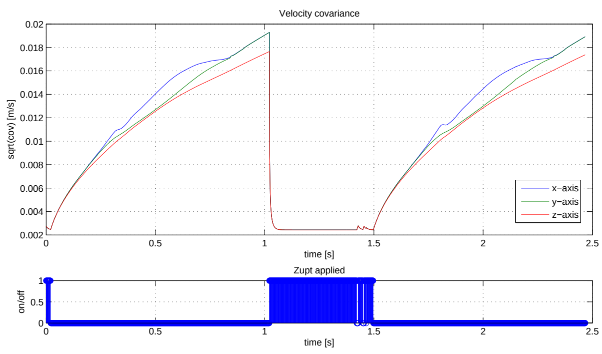


Figure 5.1: Velocity error covariance for two steps

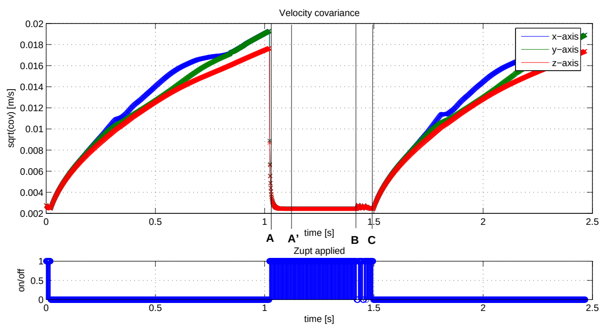


Figure 5.2: Velocity error covariance during a ZUPT. Each sample is marked.

placed in the ZUPT segment (even if this event use to happen when the stationary phase is approaching to its end).

- Due to the randomness in the appearance of discontinuities in the ZUPT vector, segmentation in C presents the problem that is impossible to be done without admitting some degree of non-causality.

Despite these issues, from situation A a valid segmentation rule is obtained by deciding the segmentation a constant time afterwards, in A'. In A the system the velocity error covariance has not converged yet. This means that the information from a ZUPT has not become fully available. If the segmentation is decided a constant time afterwards in A' the velocity error covariance has almost completely converged to a minimum and the information provided by the ZUPT is fully available. Thus, the smoothing is correctly done. For determining the distance between A' and A, a time threshold k must be decided.

On the other hand, the crossing of the threshold t must be detected when the covariance is decreasing. Otherwise, when the velocity covariance is increasing, a decission of segmentation can be done. Additional detection problems do not need to be considered, since the character of the covariance function is monotically decreasing during a ZUPT, and monotically increasing otherwise.

Thus, the proposed segmentation rule has two steps: first, the detection of a crossing instant of a velocity covariance threshold t. The value of t must be big enough to avoid that this threshold is crossed also when a ZUPT after a non-ZV state detection in a stationary segment is done. k samples (seconds) afterwards the t threshold crossing is detected, segmentation is done and a step is correctly obtained and the data in that segment can be properly smoothed. A summary of the segmentation rule can be found in Fig. 5.3., where the different parts of the graph have been exagerated to highlight the segmentation rule operation.

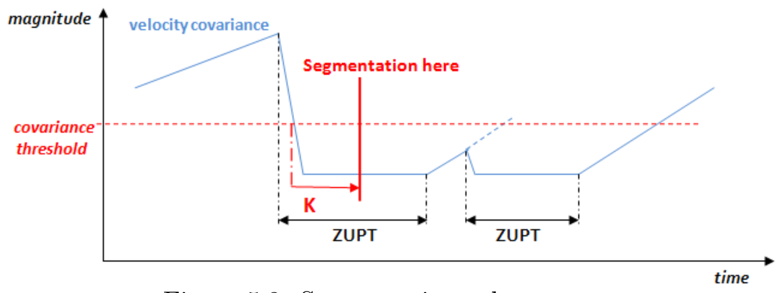


Figure 5.3: Segmentation rule summary

Last, the evaluation of this segmentation rule must be done at some point during the Kalman filter iteration. Since the current value of the covariance is required to be known, this evaluation is decided to be placed in the end of the loop, for every iteration. After segmentation, smoothing is applied over the estimated error vector $\delta\hat{\mathbf{x}}$. Then, the smoothed estimated error must be combined with the states estimated by the mechanization equations, $\hat{\mathbf{x}}^-$, getting the smoothed estimated state vector $\hat{\mathbf{x}}$. On the other side, the loop goes on processing the rest of the trajectory. Hence, the proposed algorithm combines an open-loop Kalman filtering, with a posterior smoothing and state compensation, which closes the loop. Therefore, the method here proposed can be considered a mixed open-closed loop smoothing algorithm. The method equations are shown in table 5.2, in the next section.

5.2.1 Threshold values choice

Fixing values for t and k are necessary. Thereby, let's consider typical values of the system for the test trajectories recorded from a pedestrian walking at 3.5 km/h, and frequency of the system $F_s = 820$ Hz. Under these circumstances,

- the average length of the non-stationary phase in a step is ∼800 samples (0.976 sec.).
- the average length of a stationary phase without wrong non-ZV detections is ∼400 samples (0.488 sec.).
- the mean of the maximum of the sum of velocity covariances value in our simulations (the value before the ZUPTs starts decreasing the covariance value) is $9.58 \cdot 10^{-4}$.
- the mean of the maximum of the sum of the velocity covariances for the short non-ZV states decided during the stationary phase of the step is $2.23 \cdot 10^{-5}$
- the peak value of the sum of the velocity covariances for the short non-ZV states decided during the stationary phase of the step is $4.16 \cdot 10^{-5}$.
- the minimum value of the sum of velocity covariances is $1.78 \cdot 10^{-5}$.

First, a value for t is decided. It has to be small enough so as the covariance has almost converged when detected (avoiding the problem stated for point A), but big enough to not be affected by the increase of the covariance value when a non-ZV state is detected during the stance phase of the step, see Fig. 5.3. (otherwise, two steps would be segmented where there is just one). As the time threshold is about to be fixed to allow the proper convergence, the second factor is the considered critical for the choice of t. A value of $t = 1.50 \cdot 10^{-4}$ is chosen, which is several times above the mean value and the maximum peak value detected in the simulations for the sum of velocity covariances for the short non-ZV states detected during the stationary phase of the step. The chosen value of t for covariances sum is for compensating different effects that can happen along in the different axes.

The time constant k is fixed at 30 samples (0.037 sec.), which experimentally is shown to be enough to let the convergence of the system and hence a correct smoothing. Moreover, for allowing the system to work with persons walking faster, which means shorter length of the two phases of the step, the decided k value has enough margin compared with the total duration of the stationary phase of the pedestrian walking at 3.5 km/h; 0.488 seconds.

In order to check the suitability of the chosen values, new data sets are recorded, with a pedestrian walking at an average velocity of 6.5 km/h. Unfortunately, since the system is not wireless recording data at pedestrian's higher velocities is not possible.For these data sets, the typical values are

- the average length of the non-stationary phase in a step is $\sim$600 samples (0.731 sec.).
- the average length of a stationary phase without wrong non-ZV detections is $\sim$150 samples (0.182 sec.), with high typical deviation.
- the mean of the maximum of the sum of velocity covariances value in our simulations (the value before the ZUPTs starts decreasing the covariance value) is $6.84 \cdot 10^{-4}$.
- the mean of the maximum of the sum of the velocity covariances for the short non-ZV states decided during the stationary phase of the step is $3.45 \cdot 10^{-5}$
- the peak value of the sum of the velocity covariances for the short non-ZV states decided during the stationary phase of the step is $0.966 \cdot 10^{-4}$.
- the minimum value of the sum of velocity covariances is $1.78 \cdot 10^{-5}$.

From these values, the values chosen for k and t have margin enough: k is five times higher than the mean of the maximum of the sum of the velocity covariances for the short non-ZV states decided during the stationary phase of the step; and k is also around 4 times smaller than the mean of the maximum of the sum of velocity covariances value. In the same way, the time threshold of $t = 0.037 sec.$ is five times lower than the average length of a stationary phase without wrong non-ZV detections. Even if these value has a high typical deviation, it still provides margin enough. Finally, despite both values suitably work, for faster displacements of the pedestrian, these values should be reconsidered. According to our tests, the value of k can still be diminished without affecting the estimations. In the same way, the ZV state detection can be improved in these situations but choosing other thresholds. Therefore, for higher velocities of the pedestrian, a tunning process of the thresholds for both the ZV decision and the segmentation rule should be considered.

5.3 Smoothed ZUPT-aided INS implementations

This chapter has stated and solved the particular features of the implementation of a smoothed ZUPT-aided INS. Table 5.2 summarizes the algorithm implemented in this thesis as result. It combines the segmentation rule proposed in section 5.2 with the necessary motivated open-loop implementation.

Open-loop smoothed ZUPT-aided INS equations

Initialization	$\hat{\mathbf{x}}^-(0) = E[\mathbf{x}(0)]$; $\delta\hat{\mathbf{x}}(0) = 0$; $\mathbf{P}(0) = var(\mathbf{x}(0))$;
Loop: for n = 1 to end of data	
Time update	Mechaniz. equations for $\hat{\mathbf{x}}^-(n)$: equations in table 2.1. $\delta\hat{\mathbf{x}}^-(n) = \mathbf{F}(n)\delta\hat{\mathbf{x}}(n-1)$ $\mathbf{P}^-(n) = \mathbf{F}(n)\mathbf{P}(n-1)\mathbf{F}^T(n) + \mathbf{GQG}^T$
Measurement correction (if ZUPT)	$\mathbf{K}(n) = \mathbf{P}^-(n)\mathbf{H}^T(\mathbf{HP}^-(n)\mathbf{H}^T + \mathbf{R})^{-1}$ Prediction error: $\delta\hat{\mathbf{x}}(n) = \delta\hat{\mathbf{x}}^-(n) + \mathbf{K}(n)(-\mathbf{v})$ $\mathbf{P}(n) = \mathbf{P}^-(n)(\mathbf{I} - \mathbf{K}(n)\mathbf{H})$
Segmentation rule eval.	Decide: go back to time update/ leave loop
After loop	Smoothing of the $\delta\hat{\mathbf{x}}$ segment: RTS equations (4.8) For each sample in the segment: compensate internal states to get $\hat{\mathbf{x}}(n\|N)$: equations (3.18), (3.19) Go back to loop and go on processing

Table 5.2: Open-loop smoothed ZUPT-aided INS equations

On the other side, for analysis purpose in the next chapter, a closed-loop algorithm is proposed in table 5.3. The closed-loop filter is run normally, and the smoothing is applied over the error vector $\delta\hat{\mathbf{x}}$. Then, the smoothed error $\delta\hat{\mathbf{x}}(n|N)$ is combined with the non-corrected vector $\hat{\mathbf{x}}^-(n)$ to get the final estimated state $\hat{\mathbf{x}}(n|N)$. Admitting that we are committing error in the set up of this method, along a step the committed error is so small that results negligible and therefore, this implementation can be matter of partial analysis.

Closed-loop smoothed ZUPT-aided INS equations

Initialization	$\hat{\mathbf{x}}(0) = E[\mathbf{x}(0)]$; $\mathbf{P}(0) = var(\mathbf{x}(0))$
Loop: for n = 1 to end of data	
Time update	Mechaniz. equations for $\hat{\mathbf{x}}^-(n)$: equations in table 2.1. $\mathbf{P}^-(n) = \mathbf{F}(n)\mathbf{P}(n-1)\mathbf{F}^T(n) + \mathbf{GQG}^T$
Measurement correction (if ZUPT)	$\mathbf{K}(n) = \mathbf{P}^-(n)\mathbf{H}^T(\mathbf{HP}^-(n)\mathbf{H}^T + \mathbf{R})^{-1}$ Prediction error: $\delta\hat{\mathbf{x}}(n) = \mathbf{K}(n)(-\mathbf{v})$ Compensate internal states to get $\hat{\mathbf{x}}(n)$: equations (3.18), (3.19) $\mathbf{P}(n) = \mathbf{P}^-(n)(\mathbf{I} - \mathbf{K}(n)\mathbf{H})$
Segmentation rule eval.	Decide: go back to time update/ leave loop
After loop	Smoothing of the $\delta\hat{\mathbf{x}}$ segment: RTS equations (4.8) Compensate internal states: combine $\delta\hat{\mathbf{x}}(n\|N)$, $\hat{\mathbf{x}}^-(n)$ to get $\hat{\mathbf{x}}(n\|N)$ Go back to loop and go on processing

Table 5.3: Closed-loop smoothed ZUPT-aided INS equations

Chapter 6

Implementation evaluation and analysis

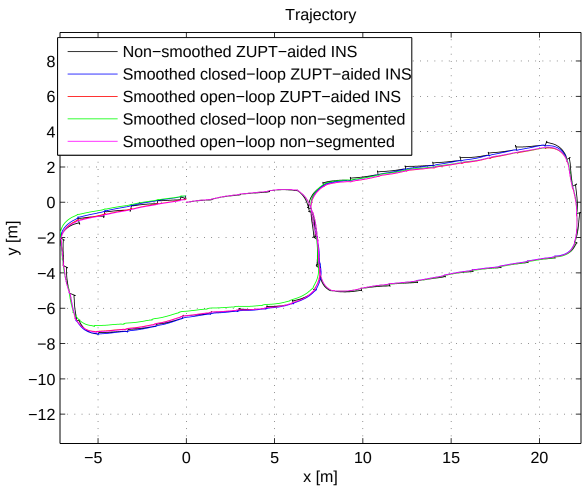
By considering the specific features of a ZUPT-aided INS, a smoothing algorithm for these INSs has been proposed. This chapter analyzes the changes that different implementations of this algorithm for ZUPT-aided INSs produce regarding the performance of the original system. Thus, first section introduces the different available implementations of the system. Next, we go through the estimated states and covariances, analyzing the behavior for each implementation. Eventually we come to analysis conclusions.

6.1 Preliminars

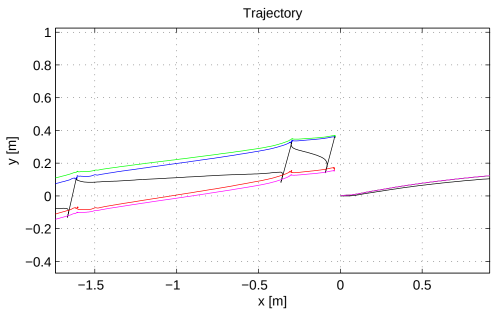
From a ZUPT-aided INS implementation, different smoothing algorithms have been developed following the considerations stated in chapter 5. This chapter evaluates the smoothing effect in the programmed implementations, in order to be able to analyze the differences. Unfortunately, for a comparison with other positioning device no external measurements are available nor can be measured with the available devices in the Signal Processing Lab in KTH. Hence, the main implementations to be evaluated in this chapter are:

- Original ZUPT-aided INS (see table 3.1).
- Smoothed closed-loop ZUPT-aided INS (see table 5.3).
- Smoothed open-loop ZUPT-aided INS (see table 5.2).
- Smoothed non-step-wise (non-segmented) closed-loop ZUPT-aided INS.
- Smoothed non-step-wise (non-segmented) open-loop ZUPT-aided INS.

The evaluation done in this chapter has its focus on the analysis and comparison with the original ZUPT-aided INS, in order to qualitatively and quantitatively (where possible) analyze the changes that have occurred in the system behavior. Then, the different smoothing implementations are compared: A segmented closed-loop implementation is compared with a segmented open-loop implementation. The two remaining non-segmented smoothing implementations do not segment the data, but apply the RTS equation once the whole Kalman filter has worked over the whole set, filtering the estimated error $\delta\hat{\mathbf{x}}$. These *non-segmented* methods would correspond with an off-line processing of the data



(a) Full xy-trajectory



(b) Detailed view of the end and the beginning of the estimated xy-trajectory

Figure 6.1: Estimated xy-trajectories for different implementations

where all the information from the future is available, and can be compared with the step-wise implementation in order to see which difference would appear if all the information from the future is available compared with just the information in a step.

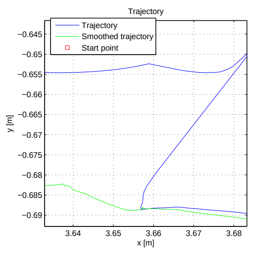
Next section evaluates the effects in the estimated states for these implementations, while the consequent section evaluates the changes in the covariances. All this will give an insight of the changes and improvements performed in the system by the proposed smoothing filter.

6.2 Estimated states analysis

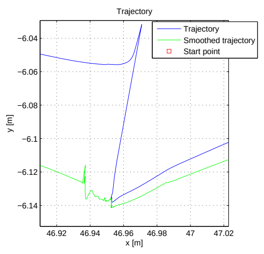
Fig. 6.1. shows the effects of the different smoothing algorithms implemented, faced against the initial implementation. It can be observed that the sought smoothing effect is achieved, but still some aspects must be reviewed. First, each implementation estimates a slightly different trajectory from the others. Furthermore, the smoothing effect is not perfect and small peaks can be appreciated in the estimated trajectory. Hence, throughout each of the next subsections a qualitative analysis of the smoothing effects on the estimated position, velocity and attitude is given, as well as a comparison between the different implementations. In addition, the improvement achieved for the *sharp* effects is quantified.

6.2.1 Estimated position states analysis

From Fig. 6.1a, some aspects in the whole estimated trajectory can be reviewed. First, it shows that the smoothed non-segmented closed-loop implementation of the system commits a large error that invalidates this implementation. The origin of this error is attributed to some flaws in the set up of the closed-loop implementation. Therefore, more analysis is required for this system. The non-segmented implementation is invalidated for further applications. However, the



(a) Segmented closed-loop implementation. 2nd step.



(b) Segmented closed-loop implementation. 34th step.

Figure 6.2: Time effect for smoothed segmented closed-loop implementation

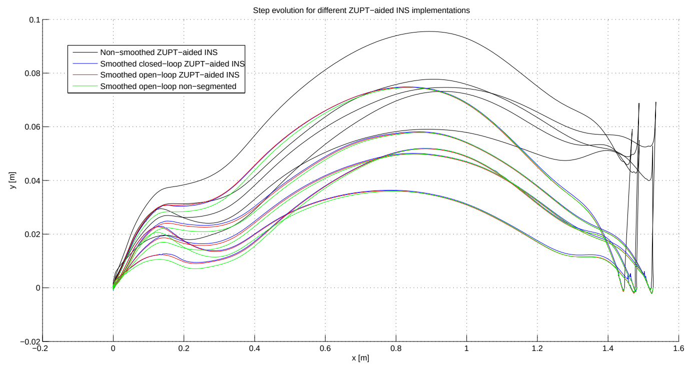
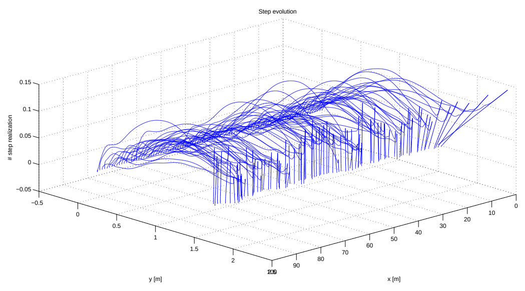


Figure 6.3: 5 realizations of steps for different implementations

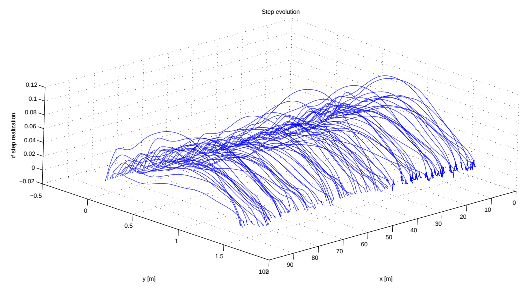
segmented closed-loop implementation seems to provide a valid estimation. Admitting that some error is comitted, the error along a step does not increase so much and can be considered negligible. Therefore, the closed-loop implementation, which presents stability properties that make it desirable, could be used as smoother if the error comitted is assumed. Further analysis than the performed in this thesis is required for this segmented closed-loop implementation. The reason is that the position error covariance increases along time, and this covariance value is required for smoothing. When enough time has passed, it is appreciable he correction that is shown in Fig. 6.2b. In Fig. 6.2 is shown the behavior of the segmented closed-loop estimated trajectory in the sharp transitions, for a straight line test-traject. Fig.6.2a is the second step of the pedestrian, while Fig.6.2b 34th step.

Qualitative analysis of the different implementations can be done. Five aligned realizations of a step of a pedestrian walking along a straight line trajectory are shown overlapped in Fig. 6.3 for the different implementations. Note that the axes are not equal because of illustrative reasons. There is shown that not big differences appear between the different smoothing realizations for the processing of a step.

Although in Fig. 6.3 the smoothing effect is also shown, it is better illustrated in Fig.6.4 , where 3D plots for the original implementation and the segmented open-loop implementation are shown. They are overlapped in Fig.6.5. These figures prove that the large correction at the end of the step for the original implementation has been removed for the smoothed ones. For a ∼50 meters straight line trajectory, the corrections performed increase along time, as the position covariance does. At the end of the original implementation, the typical



(a) Original implementation



(b) Segmented open-loop implementation

Figure 6.4: 3D plot of aligned steps

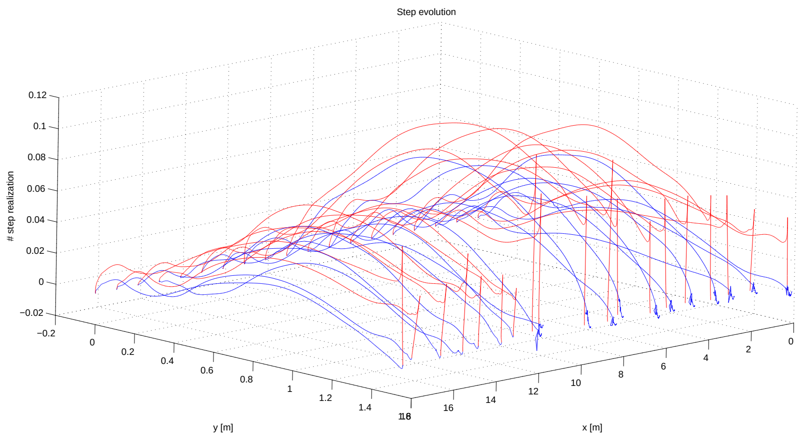


Figure 6.5: Overlapped 3D plot of aligned steps

range of correction in the order of $1 \cdot 10^{-1}$ to $1.5 \cdot 10^{-1}$ m. For the closed-loop implementations, there appear sharp effects due to the smoothing that increase with time. At the beginning of a trajectory these are negligible, but in the end of a 50 meters trajectory this effect is in the order of $5 \cdot 10^{-2}$ meters for the closed-loop implementations. The same occurs for the segmented open-loop, where these effects increase along time and are in the order of 1 to $2 \cdot 10^{-2}$ m. in the end of the 50 meters trajectory, an improvement of an order of magnitude. For a non-segmented open-loop, the order of these corrections is in the order of $1 \cdot 10^{-3}$ in the end of a 50 meters straight line trajectory. Thus, this correction is slightly smaller for the non-segmented that for the segmented implementation. This is caused by the covariance value, which is a little smaller in the non-segmented open-loop that for the segmented because the non-segmented has information of the whole trajectory. Although this error seems to increase along time, for long trajectories it seems to converge as the covariance does. However, further analysis about the time evolution is required. Longer straight line trajectories should be recorded for this.

On the other side, Fig. 6.6 shows that the trajectory next to the *sharp* correction point adopts now a typical *peak* shape around this point. During a ZUPT the body's velocity is nearly zero for a big segment of data, producing a large concentration of estimated positions concentred there. Smoothing that part of the segment is not possible. However, while the shoe is not stationary there is a drift in the estimation, drift that when is corrected leads to the sharp corrections that the original implementation has. Smoothing that part do not

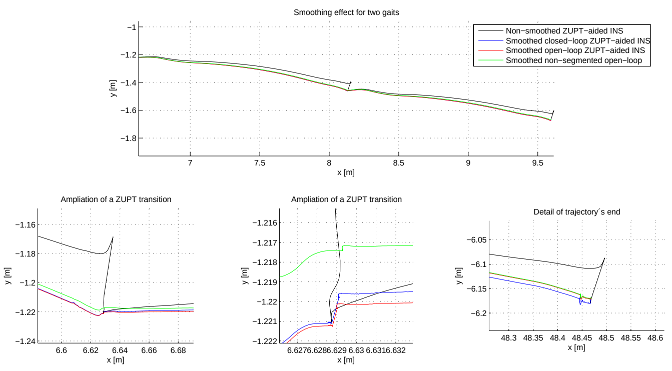


Figure 6.6: Typical realization for two steps. Subfigures show different details.

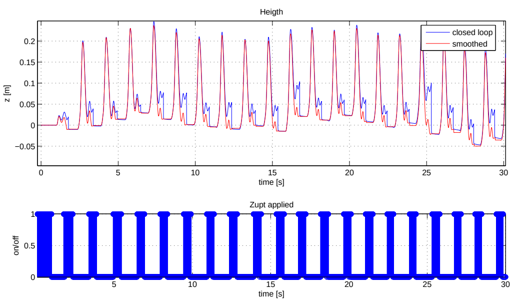


Figure 6.7: Estimated state evolution for z-axis

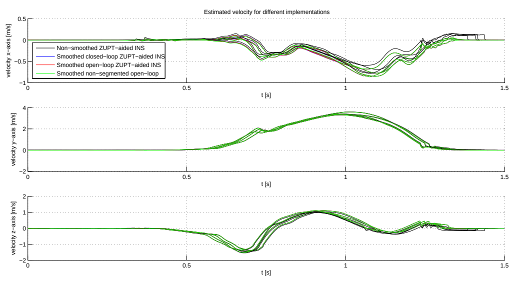
fully compensate this drift, and the estimated trajectory tends towards the direction of the original drift, until the effect of the big concentration of points while the shoe is stationary forces the smoothed trajectory to come back to the original trajectory.

In Fig. 6.6 can be seen as well a detail of the trajectory's end, which is different for each implementation. This occurs due to the way the different estimations are built: the smoothed closed-loop trajectory is built by filtering directly the states, while the other two filter the error and then it is added to a by the navigation equations estimated state. The difference is bigger for the segmented trajectory just due to the fact that these equations accumulate error and a whole smoothing of the trajectory definitely estimates better the error than an estimate of just the error segment by segment.

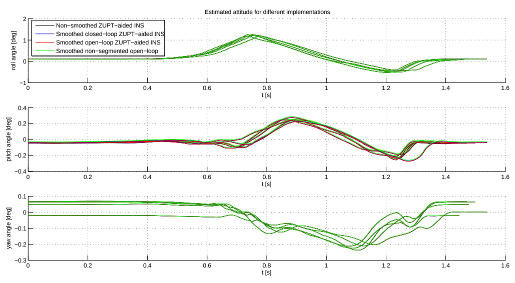
An analysis of the trajectory for the z-axis provides a smoothing of each step, whose values are modified, but the evolution along a whole trajectory of the z-axis has not been fully tested, although its basic performance works properly. Further analysis is required with new data tests. Effect of the smoothing for the z-axis is shown in Fig. 6.7, where it can be observed the smoothing along a step.

6.2.2 Estimated velocity states analysis

From Fig.6.8a it can be seen that not big differences can be appreciated for different implementations for the smoothed velocity estimate. This is due to the fact that the velocity turns zero in every stationary phase, decorrelating the velocity components for every step no matter which implementation is used. On the other side, for this specific realization the smoothing is mainly performed



(a) 5 realizations of estimated velocity for different implementations



(b) 5 realizations of estimated attitude for different implementations

Figure 6.8: 5 realizations for estimated velocity and attitude for different implementations

for the x-axis because the main displacement takes place on this axis. There it can be seen that the drastic correction done for the velocity do not appear in the smoothed implementations.

6.2.3 Estimated attitude states analysis

In Fig.6.8b is shown different realizations of the smoothing filter algorithms applied. There are not big differences among implementations for the attitude states. Even the smoothing effect can hardly be appreciated. This occurs because the attitude components are weakly correlated from the velocity and position estimates, as it can be seen in equations (2.18) (2.19) (2.20).

6.3 Covariances analysis

It must be noticed that the fact of recalculating the estimated states implies modifying the relation among the different states: that is, modifying the covariance matrix of the variables. As it will be graphically shown in next subsections, a general effect produced by the Kalman filter while there is not a ZUPT is that the covariance between the variables increases along time. This is because the filter is doing an estimation by using data from previous iterations without any additional correction. Therefore, when the ZUPT occurs and a correction takes place by using the knowledge of the velocity state of the system, the covariance between variables decreases drastically in the same way that the *sharp* corrections in the trajectory took place.

However, a smoothing problem modifies this character. As we stated in section 4.1., a smoothing problem aims to minimize the error variance of the estimated state vector $\hat{\mathbf{x}}_{n|N}$. Now, the whole data segment is available from the very beginning of the filtering process. That is, the ZUPT information is available at the beginning and therefore the information of the ZUPT can be known for every sample in the segment. Hence, it is logic to expect that now the covariance is minimized.

Through the next subsections, it is shown a qualitative analysis of the error covariance character after the smoothing, by comparing the three systems that we have shown from the beginning. For reasons of clarity in the exposition, the velocity error covariance opens the analysis.

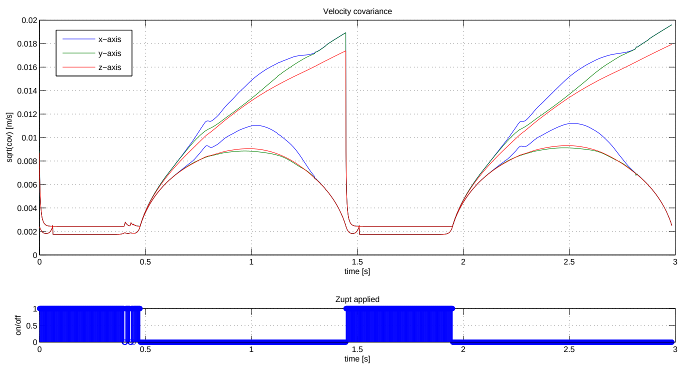


Figure 6.9: Velocity covariance evolution for two steps

6.3.1 Velocity error covariance analysis

In Fig.6.9 the velocity error covariance evolution along time is exemplified for two steps of a typical realization for the original implementation and the smoothed open-loop ZUPT-aided INS. For the original implementation the covariance value monotically increases along time when there is no ZUPT and, when a ZUPT takes place, an external measurement provides additional information that makes the relation between the components decrease drastically and keep on monotically decreasing during the ZUPT until a minimum, which is kept until the end of the ZUPTs.

It is easy to notice that now there is a big difference between the original implementation error covariance and the smoothed. Why is this happening? After segmentation, all data is available from the beginning and therefore the information provided by a ZUPT, which decorrelates the components, is made available along the step. This occurs because the smoothing filter provides the information from the future. All the segments follow the sequence *there is ZUPT - there is no ZUPT - there is ZUPT*. This means that there is ZUPT known at both sides of the segment and both ZUPTs periods are known along a no-ZUPT phase of the step. Logically, during this no-ZUPT part of the step the closer the time instant is to a ZUPT, the more decorrelated the components are since they are closer to the information provided by a ZUPT. This is why the covariance evolution along time has this "semi-circunference" shape in the no-ZUPT parts: since the segment has ZUPT in both extremes, the further we are from any extreme the higher the error covariance is and therefore the middle point of the non-statationary phase is the one with higher error covariance value. This semi-circunference shape shows clearly that the information provided by the ZUPT is available along the whole step. Fig. 6.10 shows this in a more intuitive way. The error covariance monotically increases during a no-ZUPT phase of the step. If we run the system in an anticausal way, then the error covariance increases from the future towards the past. When smoothing is done, information from the future and the past is available at the same time. Therefore, the error covariance adopts the symmetric shape that has been discussed along this paragraph.

Other two aspects can be noted from Fig. 6.9. First, there is a discontinuity in the covariance every time instant that the segment is cut, as it can be seen in $t = 1.5$ sec. The left extreme value corresponds to the first iteration of the smoothing algorithm, which process data backwards, see equation 4.8. Therefore, the smoothed sample has no additional information from the future $\hat{\mathbf{x}}(N|N)$ and the covariance value from the original implementation is the

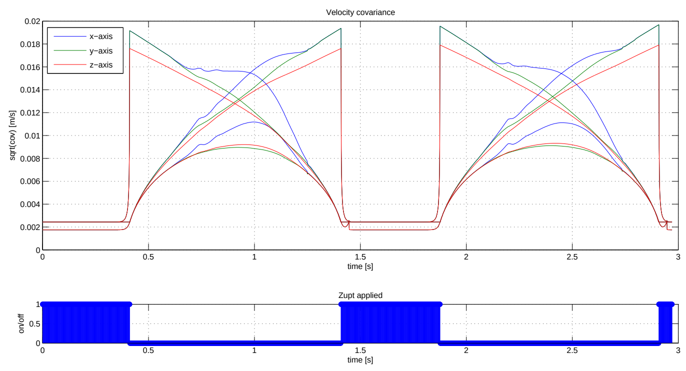


Figure 6.10: Velocity covariance evolution for two steps. Overlapped with anti-causal realization.

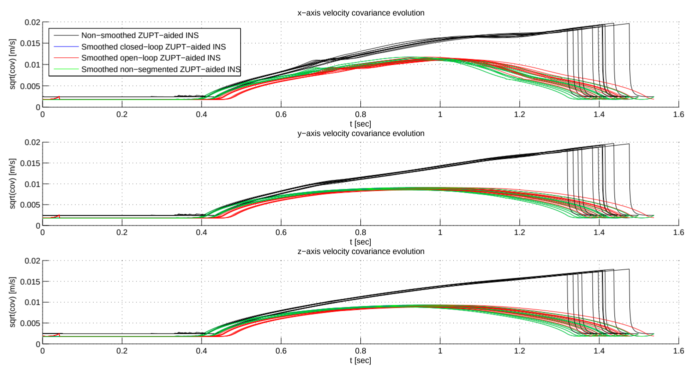
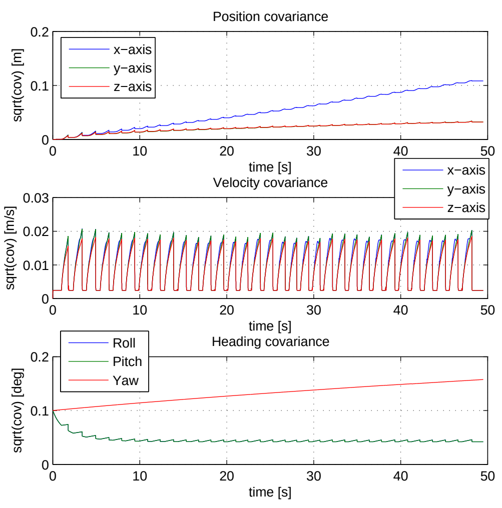


Figure 6.11: Velocity covariance for several implementations

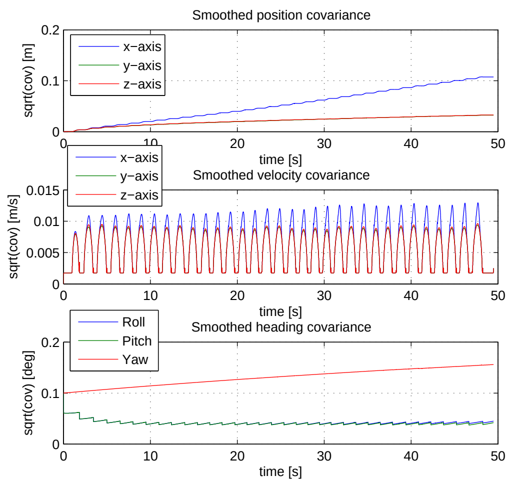
same for the original and the smoothed implementation. As data is processed backwards, information from the future becomes available for the next samples $\hat{\mathbf{x}}(N-1|N), \hat{\mathbf{x}}(N-2|N), ...$ and the covariance decreases until a minimum. Note that if the time threshold used was minor, there would be not enough ZUPT information at both sides of the segment and the information provided by the ZUPT is not enough to provide a correct smoothing.

The second situation to be noted in Fig. 6.9 is the value of the covariance during the ZUPT: it must be noted that it is minor than it was originally: for this implementation, it diminishes from $2.432 \cdot 10^{-3}$ to $1.746 \cdot 10^{-3}$ for each axis. Knowledge of samples from the future explains this decreasement.

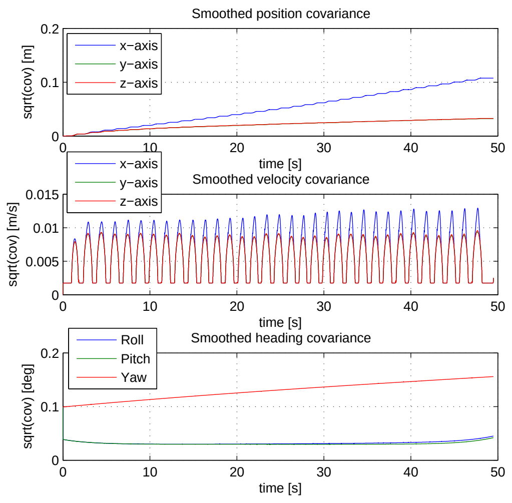
From Fig.6.11., where are shown ten steps for each implementation for each axis, it can be concluded that there are not significative differences between one implementation or other in terms of covariance behavior and therefore the analysis performed for the open-loop can be extended to the other implementations. The minimum covariance values are the same for every implementation. The fact that the open-loop appears slightly delayed is due to the different segmentation points from the closed-loop implementation. Besides, the non-segmented implementation does not present the discontinuitie that is found for the segmented implementation, because there is no segmentation.



(a) Original



(b) Segmented open-loop



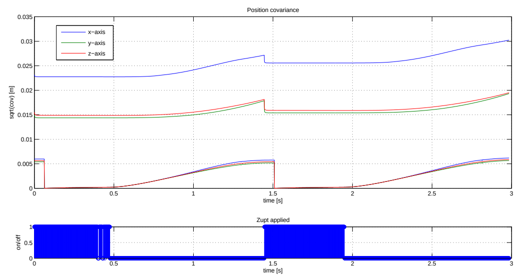
(c) Non-segmented open-loop

Figure 6.12: Position, velocity and heading covariance time evolution

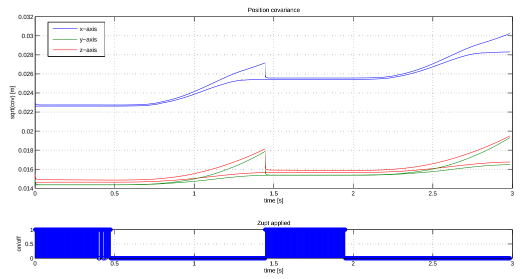
6.3.2 Position error covariance analysis

Fig. 6.12. shows the time evolution for the position, velocity and heading error covariance for a whole realization of the system for the same implementations; and Fig. 6.13 exemplifies the position error covariance evolution along time for two steps of one realization for the original implementation and the segmented and non-segmented smoothed open-loop ZUPT-aided INS.

First, from Fig. 6.12, it must be noticed that the position covariance increases along time for the different implementations. A ZUPT estimates that the velocity should be zero, and from there it corrects the rest of the components. The correction performed over the position covariance is appreciable, but does not fix the position error to zero as it does with the velocity and therefore, in a ZUPT the position components are not almost completely decorrelated as they are the velocity ones. However, as it can be seen in Fig 6.13, the covariance when a ZUPT takes place still decreases drastically.



(a) Position covariance for two steps. Original vs. segmented open-loop



(b) Position covariance for two steps. Original vs. non-segmented open-loop

Figure 6.13: Position covariance for two steps

The position covariance evolution along a step is illustrated by Fig. 6.13. This evolution along a step is different that for the velocity coviariance. For the original implementation, the covariance only increases when there is no ZUPT and, as soon as a ZUPT takes place, it decreases drastically but not back to the original value at the beginning of the segment. Thus, along the time the position error covariance increases. For the non-segmented open-loop implementation, the smoothing effect is appreciated in the fact that now there is no decreasing of the position covariance: since the information for the ZUPT is available along the whole segment, it is logical that the covariance increases when there is no ZUPT but without decreasing at any moment when there is a ZUPT because of a correction. The cause is that the correction information is available for the whole segment. Finally, for the segmented open-loop implementation the position covariance evolves in the same way along a step than for the non-segmented open-loop. However, the covariance values does not increase along time owing to the reinitialization of values done at the beginning of each segment (see table 5.2). As the non-segmented implementation considers all the time values, its covariance value is slightly lower than the position error covariance value for the segmented implementation.

6.3.3 Heading error covariance analysis

In Fig. 6.12 can be seen the time evolution of the heading eerror covariance for different implementations. It can be noted that the original implementation presents some drastic corrections, particularly for the yaw, which do not appear for the non-segmented open-loop implementation, where these corrections have been smoothed. In general the average value of the covariance is slightly lower for the non-segmented open-loop implementation due to the knowledge of samples from the future that provides the smoothing. The segmented open-loop system shows how the covariance is reinitialized at the end of every segment, causing the appearance of the peaks that appear in the graph.

Special mention requires the difference among the roll and pitch covariances, overlapped in the graphs, with the yaw covariance. Why these drastic corrections appear for the yaw covariance but not for the roll and pitch ones? The roll and pitch are observable from the INS measurements. However, the yaw angle is not observable, being reflected in a higher covariance value.

6.4 Analysis conclusions

From last sections, some conclusions can be extracted. First, for a near to real-time implementation, the proposed segmented open-loop smoothing algorithm works properly. For offline processing, the non-segmented open-loop implementation also provides satisfactory results. However, this does not happen for the non-segmented closed-loop implementation. Effects in the estimated states are analyzed and the improvement in the corrections quantified. Further analysis is required for the possibility of implementing a segmented closed-loop ZUPT-aided INS, due to the stability properties a closed-loop implementation has. The available closed-loop implementation provides a practical acceptable result, but assuming that we are committing an error in the set up of the filter.

On the other side, covariance analysis for the different algorithms is also performed. Different effects due to the segmentation and the open-loop implementation are analyzed, as well as the change of behaviour from the original ZUPT-aided INS, that shows that the smoothing is correctly done by showing the effect along a step that has to know all the step information at the moment of processing.

Chapter 7

Conclusions and future work

An analysis of a foot-mounted ZUPT-aided INS on one side; and several general smoothing estimators on the other have lead us to the implementation of a smoothed ZUPT-aided INS. Implementation issues have been considered, and an analysis of different implementations for the smoothing algorithm have been performed. This last chapter comes up to conclusions from all this process, and suggests future work to be done on the topic.

7.1 Conclusions

Along this thesis an step-wise smoothing filter for a ZUPT-aided INS has been proposed. An analysis of the already available system is performed and an analysis of different smoothing algorithms performed, to allow their combination for creating the sought smoothed ZUPT-aided INS. The combining process requires the implementation of an open-loop version of the originally available system to allow the correct linking of the terms available from the previous implementation and the terms in the general smoothing formulas. Moreover, since the steps are irregularly spaced and the measurements (the ZUPTs) appear in clusters, a segmentation rule is proposed for a step-wise data processing. This rule is based on covariance and time thresholds. By considering these two issues, a smoothed open-loop step-wise ZUPT-aided INS version is proposed. Qualitative analysis for this and other smoothing implementations is done, with special emphasis in the proposed smoothing segmented open-loop algorithm. This algorithm is expected to be implemented for near to real-time application.

7.2 Future work

Smoothing for foot-mounted ZUPT-aided INS is considered to be very useful for the further research in pedestrian navigation and tracking. So far, corrections on the by the ZUPT-aided INS estimated trajectory were done by using additional systems, such as cameras worn by the pedestrian, which is not practical. The proposed smoothing algorithm does not require any additional sensor, making it therefore a better solution for inertial navigation.

Under the topic of this thesis, further work is expected to be done. Additional testing of the segmentation rule for faster displacements is required, where a tunning process together with the ZV states detection thresholds must be done. Due to reasons of divergence along time of open-loop algorithms, further analysis for longer time periods should be performed. In the same way, further analysis of the segmented closed-loop proposed algorithm is necessary.

The code used for simulations is to be implemented for a real INS for near real-time applications; as well as the performed work is to be presented to the research community. It is expected that other navigation researchers can be interested in this work for facilitating their job in the area.

Bibliography

[1] Jay A. Farrell, Matthew Barth; *The Global Positioning System & Inertial Navigation*; McGraw Hill Professional, Dec 1998.

[2] Jay A. Farrell; *Aided navigation - GPS with high rate sensors*; McGraw Hill Professional, 1st edition, Apr 2008.

[3] J.L. Farrell; *Integrated Aircraft Navigation*; Academic, New York, 1976.

[4] Carl Fischer, Poorna Talkad Sukumar, Mike Hazas; *Tutorial: implementation of a pedestrian tracker using foot-mounted inertial sensors*; IEEE Pervasive Computing, 09 Jan. 2012. IEEE computer Society Digital Library.

[5] Eric Foxlin; *Pedestrian tracking with shoe-mounted inertial sensors*; IEEE Computer Graphics and Applications, vol. 25, no.6, Nov-Dec 2005.

[6] A. Jiménez, F. Seco, J. Prieto, and J. Guevara; *Indoor Pedestrian Navigation using an INS/EKF framework for Yaw Drift Reduction and a Foot-mounted IMU*; in Proc. of WPNC, 2010.

[7] Thomas Kailath, Ali H. Sayed, Babak Hassibi; *Linear estimation*; Prentice Hall, 1st edition, 2000.

[8] Thomas Kailath, Lennart Ljung; *Two Filter Smoothing Formulae by Diagonalization of the Hamiltonian Equations*; Stanford univ. CA information systems lab., 1982.

[9] Steven M. LaValle; *Planning Algorithms*; Cambridge Univ. Press, 2006.

[10] Anthony Lawrence; *Modern Inertial Technology: Navigation, Guidance, and Control*; Springer, mechanical engineering series, 2nd edition, 1998.

[11] Maciej Niedźwiecki; *Locally Adaptive Cooperative Kalman Smoothing and Its Application to Identification of Nonstationary Stochastic Systems*; IEEE transactions of signal processing, vol. 60, no.1, Jan 2012.

[12] John-Olof Nilsson, Isaac Skog & others; *OpenShoe*, `www.openshoe.org`; 2011.

[13] PooGyeon Park, Thomas Kailath; *New Square-Root Smoothing Algorithms*; IEEE transactions of signal processing, vol. 41, no. 5, May 1996.

[14] Isaac Skog, John-Olof Nilsson, Peter Händel, Jouni Rantakokko; *Zero-Velocity Detection - An Algorithm Evaluation*; IEEE Transactions on Biomedical Engineering, vol. 57, no.11, Nov. 2010.

[15] Joseph E. Wall, Jr., Alan S. Willsky, Nils R. Sandell, Jr.; *On the Fixed-Interval Smoothing Problem*; Gordon and Breach Science Publishers Inc., *Stochastics*, 1981, Vol. 5, pp. 1-41.

Anexo B

Artículo: Smoothing for ZUPT-aided INSs

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 where large corrections to the state estimates are applied. These large corrections complicates motion analysis and can be distracting for visualization. The situation is illustrated in Fig. 1 where multiple tracked steps are plotted aligned beside each other. 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 require some non-causal filtering. Consequently, to eliminate the discontinuities and the unsymmetrical covariance over the steps, the implementation of a smoothing filter for a ZUPT-aided INS is considered. To our knowledge, no formal treatment of smoothing for such systems has previously been presented, even though an extensive literature on the general subject exists.

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, some varying-lag smoothing rule is necessary and therefore introduced. By combining all 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 analyzed and quantified.

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

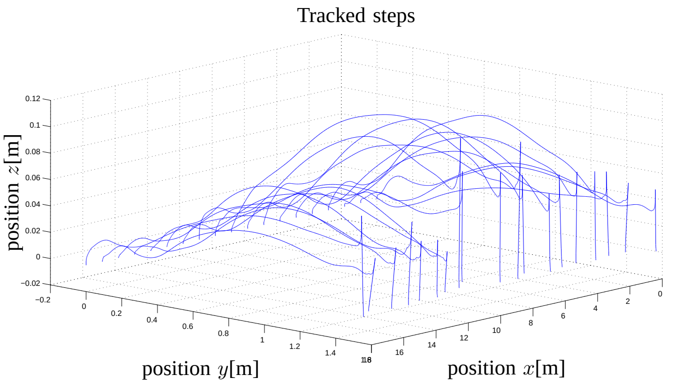


Fig. 1: Steps from a straight-line trajectory as tracked by a ZUPT-aided INS. Large corrections causing apparent discontinuities can be seen at the end of each step. Note the difference in scale between the xy-plane and the z-axis.

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

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

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 are 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 equivalently be 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 interchangeably use the different representations.

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 us to compensate 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 make 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-measurement in form of ZUPTs are used. The idea under laying 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 $\delta\mathbf{x}_n$ are 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 of 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 $T(\{\boldsymbol{\omega}^i, \mathbf{f}^i\}_{W_n}) < \gamma$
        $\mathbf{K}_n = \mathbf{P}_n\mathbf{H}^T(\mathbf{H}\mathbf{P}_n\mathbf{H}^T + \mathbf{R})^{-1}$
        $\delta\hat{\mathbf{x}}_n = \mathbf{K}_n\hat{\mathbf{v}}_n$
        $\mathbf{P}_n \leftarrow \mathbf{P}_n(\mathbf{I} - \mathbf{K}_n\mathbf{H})$                      (8)
        % 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 doing dead-reckoning by a ZUPT-aided INS. However, direct implementation of a smoothing algorithm is not possible. Next section motivates why and introduces the modifications done to the algorithm in order to get 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 very 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 along the whole step via a smoothing algorithm. For many applications, some degree of lag (non-causality) can be tolerated such that smoothing step by step can be done 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 straight-forward relation with the previous customary ZUPT-aided INS algorithm. The RTS formula is

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

where $\hat{\boldsymbol{\chi}}_{n|n}$ is some arbitrary state vector, $\mathbf{\Gamma}_n$ 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 directly be applied to the estimation (8). This is because of the internal compensation done in (8) changing the value of $\delta\hat{\mathbf{x}}_n$ preventing us from directly applying the smoothing to it. This problem can be solved by simply avoiding the internal compensation and instead run (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 $\delta\hat{\mathbf{x}}_{n-1}$

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

However, once the smoothing has been applied, there is nothing that prevents us from doing the internal compensation.

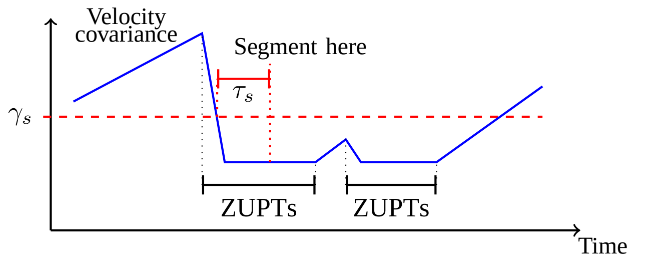


Fig. 2: Velocity error covariance increases along a step and decreases drastically when the ZUPTs are applied. During the steady phase of a step, a no-ZUPT decision can be made. These erroneously decided segments are short, the covariance increases but do not trespass the selected covariance threshold.

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 decission 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 again. Since these erroneously determined no-ZUPT segments are short, the velocity error covariance can not 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 in the point where the velocity error covariance threshold γ_s is trespassed leads to an incorrect behavior of the smoothing algorithm. In 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 done 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 does 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 few covariance values $\mathbf{P}$ could be stored and used to interpolate the rest on the backward pass.

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

```
Initializ.: $\hat{\mathbf{x}}_0 = E[\mathbf{x}_0]$, $\delta\hat{\mathbf{x}}_0 = 0$, $\mathbf{P}_0 = var(\mathbf{x}_0)$,
$c = 0$, $s_{\text{start}} = 1$, $s_{\text{end}} =$"end of data"
Loop while $s_{\text{start}} < s_{\text{end}}$
  % Forward Kalman filter
  Loop: $n = s_{\text{start}}$ to $s_{\text{end}}$
    % 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 $s_{\text{start}}$
      $\mathbf{A}_n = \mathbf{P}_{n|n}\mathbf{F}^T\mathbf{P}^{-1}_{n+1|n}$
    $\delta\hat{\mathbf{x}}_{n|s_{\text{end}}} = \delta\hat{\mathbf{x}}_{n|n} + \mathbf{A}_n(\delta\hat{\mathbf{x}}_{n+1|s_{\text{end}}} - \delta\hat{\mathbf{x}}_{n+1|n})$
    $\mathbf{P}_{n|s_{\text{end}}} = \mathbf{P}_{n|n} + \mathbf{A}_n(\mathbf{P}_{n+1|s_{\text{end}}} - \mathbf{P}_{n+1|n})\mathbf{A}_n^T$
  % Internal state compensation
  Loop: $n = s_{\text{start}}$ to $s_{\text{end}}$
    $\begin{bmatrix} \hat{\mathbf{p}}_n \\ \hat{\mathbf{v}}_n \end{bmatrix} \leftarrow \begin{bmatrix} \hat{\mathbf{p}}_n \\ \hat{\mathbf{v}}_n \end{bmatrix} + \begin{bmatrix} \delta\hat{\mathbf{p}}_{n|s_{\text{end}}} \\ \delta\hat{\mathbf{v}}_{n|s_{\text{end}}} \end{bmatrix}$
    $\hat{\mathbf{R}}_n \leftarrow (\mathbf{I}_3 - \boldsymbol{\Delta}_{n|s_{\text{end}}})(\hat{\mathbf{R}}_n)$
    $\delta\hat{\mathbf{x}}_n \leftarrow \mathbf{0}$
  $s_{\text{start}} = s_{\text{end}} + 1$, $s_{\text{end}} =$ "end of data", $c = 0$
```

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 an off-line processing of the data, where the information provided by all the future ZUPTs is known.

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 the non-segmented implementations are little. 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 we have experienced some sharp corrections in the smoothing. 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.

Fig. 5 shows the smoothing effect over the velocity error covariance. For (8), the error covariance increased 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

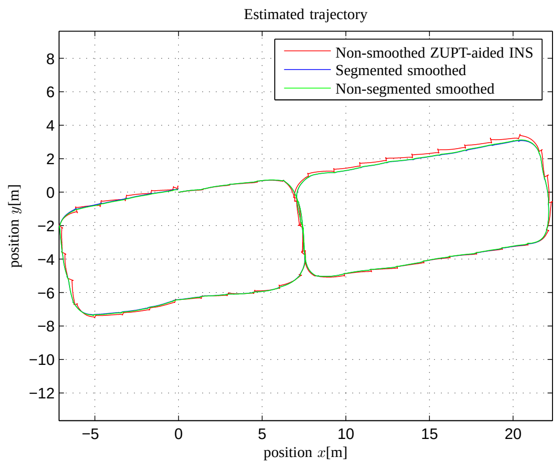


Fig. 3: Effect of smoothing over a trajectory. The large corrections at the end of each step have been smoothed. Note that the segmented and non-segmented smoothed trajectories are essentially overlapping.

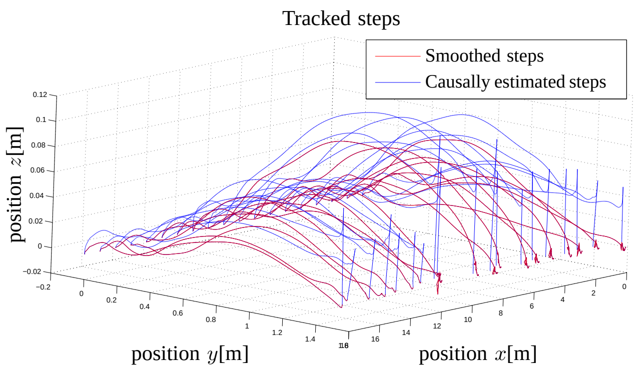


Fig. 4: Effect of smoothing over multiple steps. The large corrections at the end of each step have been smoothed. Note the difference in scale between the xy-plane and the z-axis.

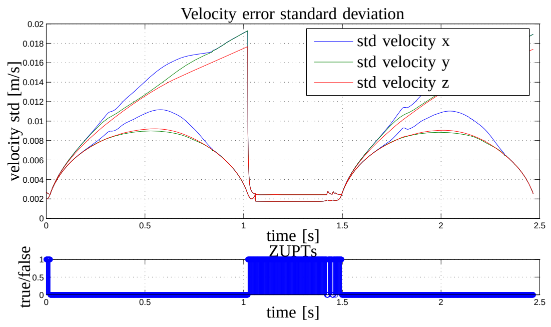


Fig. 5: Typical effect of smoothing over the velocity error covariance throughout two steps.

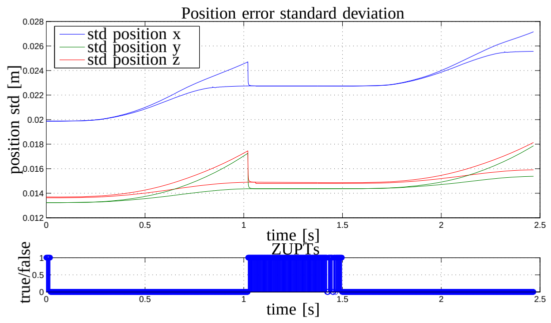


Fig. 6: Typical effect of smoothing over the position error covariance throughout two steps.

the furthest point from any ZUPT, and the covariance looks symmetric 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 along 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 along 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 not 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.

V. Conclusion

In this article we have suggested an smoothing algorithm for ZUPT-aided INSs, which has been shown to eliminates 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. 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 and P. Händel, “A note on the limitations of ZUPTs and the implications on sensor error modeling,” in *Proc. IPIN*, 2012.