Universidad Zaragoza

1542

Trabajo Fin de Máster

Inicialización SLAM Visual-Inercial: una formulación lineal general

Visual-Inertial SLAM Initialization: A general linear formulation

Autor

Javier Domínguez Conti

Director

Javier Civera Sancho

Escuela de Ingeniería y Arquitectura (EINA)

2019-2020

Resumen

Los sistemas de Localización y Mapeado Simultáneo (SLAM, del inglés *Simultaneous Localization and Mapping*) son una de las tecnologías más relevantes para aplicaciones robóticas y de realidad virtual y aumentada. El objetivo de estos sistemas es reconstruir un mapa de un determinado entorno a partir de imágenes así como estimar la trayectoria de la cámara que toma dichas imágenes.

La relevancia que han adquirido los sistemas SLAM a lo largo de los últimos años se debe a la importancia de sus numerosas aplicaciones. Estos sistemas son imprescindibles en, por ejemplo, la realidad aumentada, los vehículos autónomos o la navegación de robots en entornos desconocidos.

Los sistemas de SLAM visual-inercial (VI-SLAM) son aquellos que, además de imágenes, utilizan medidas inerciales para reconstruir el entorno. El bajo coste, pequeño tamaño y baja potencia requerida por los sensores utilizados para tomar las medidas han hecho de los sistemas VI-SLAM uno de los más populares dentro de los sistemas SLAM. Sin embargo, a pesar de su popularidad, la inicialización de los sistemas VI-SLAM continúa siendo crítica debido a su falta de robustez y precisión.

En este trabajo se ha abordado el problema mediante el planteamiento de una nueva formulación general para la inicialización de sistemas VI-SLAM. Con la formulación propuesta se han alcanzado mayor robustez y precisión que el actual estado del arte. El trabajo se basa en la utilización de un sensor monocular-inercial, es decir, una cámara monocular y una unidad de medición inercial (IMU, del inglés *Inertial Measurement Unit*).

La formulación presentada ha sido probada en un *dataset* público que dispone de una considerable variedad de condiciones de iluminación, movimiento y escenarios. Además, se han realizado una gran cantidad de experimentos que muestran la importante mejora de robustez y precisión con respecto a los actuales sistemas de inicialización.

Parte del trabajo realizado ha sido recogido en el artículo *"Visual-inertial slam initialization: A general linear formulation and a gravity-observing non-linear optimization."*, que fue publicado y presentado en el IEEE International Symposium for Mixed and Augmented Reality 2018 (ISMAR).

Índice general

Capítulo 1

Introducción

1.1. Visión por computador

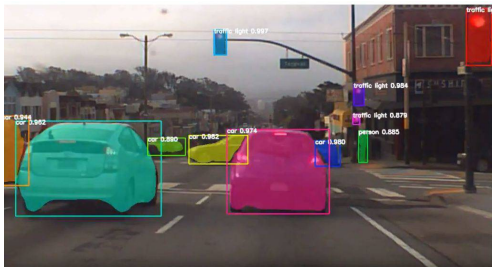
La visión por computador es el conjunto de técnicas para captar, procesar y analizar imágenes con el objetivo de que un ordenador realice una tarea determinada. Según esta definición, el propósito de la visión por computador es dotar a un ordenador de capacidades asociadas al sistema de visión humano.

La visión por computador ha ganado popularidad desde su aparición en los años 60 y, hoy en día, es uno de las disciplinas más activas y con un progreso más rápido, empujado en gran parte por el gran número y relevancia de sus aplicaciones potenciales. El creciente interés por esta disciplina se debe a la importancia de los problemas que puede resolver. Gran parte de estos problemas consisten en la interacción de máquinas con su entorno.

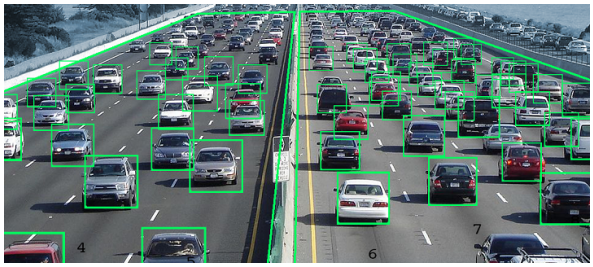
Para ilustrar la relevancia de esta disciplina, la Figura 1.1 recoge solo algunas de sus aplicaciones más comunes. La Figura 1.1a muestra su aplicación en vehículos autónomos donde la visión por computador es imprescindible para la identificación de otros vehículos, semáforos, peatones, etc. Por otro lado, la Figura 1.1b muestra un proceso de identificación de coches en una carretera para el control del tráfico. Otra de sus aplicaciones más extendidas se encuentra en la industria, donde se implementan sistemas para el control de procesos. Por ejemplo, las Figuras 1.1c y 1.1d muestran el reconocimiento de tapones de botella correctamente colocados y la clasificación de diferentes botellas de plástico en un proceso de reciclado, respectivamente.

1.2. SLAM Visual

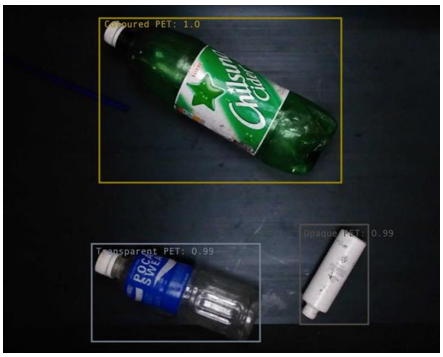
El modelado y reconstrucción de entornos en 3D es otro de los principales intereses de la visión por computador. En la Figura 1.2 pueden verse algunos ejemplos representativos. Las representaciones 3D de espacios permiten analizarlos en detalle y facilitan la navegación por ellos. Por ejemplo, la Figura 1.2a muestra el modelado de un tramo del sistema digestivo humano. Este tipo de representaciones pueden facilitar en gran medida los diagnósticos médicos y las intervenciones. Por otro lado, la Figura 1.2b



(a) Identificación de elementos por parte de un vehículo autónomo. Aplicación de [11].



(b) Identificación de vehículos para el control del tráfico. Aplicación de [26].



(c) Clasificación de botellas en un proceso de reciclado.

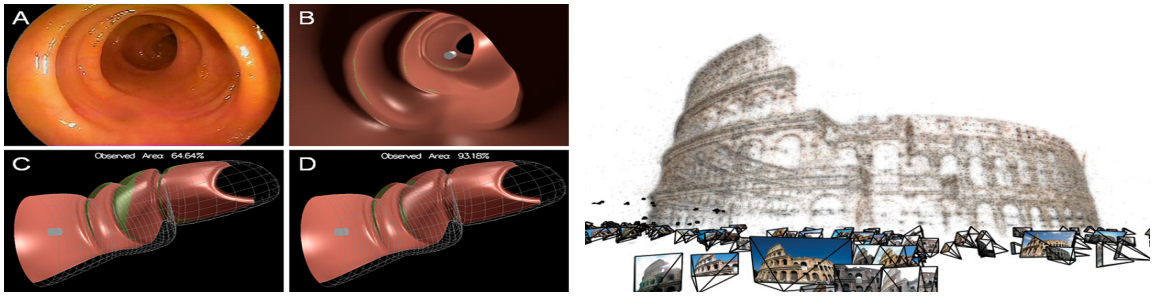
(d) Identificación de tapones de botella en un control de calidad.

Figura 1.1: Ejemplo de aplicaciones de la visión por computador. (a) y (b) muestran aplicaciones en vehículos autónomos y control de tráfico. En (c) y (d) se muestran aplicaciones industriales.

recoge la reconstrucción de un edificio completo. Este otro tipo de reconstrucciones permiten, por ejemplo, la navegación de robots autónomos, el desarrollo de estudios arquitectónicos o la visualización de alteraciones en la escena (realidad aumentada y realidad virtual).

Entre las diferentes técnicas de modelado 3D se pueden destacar las de localización y mapeado simultáneos (SLAM, del inglés *Simultaneous Localization and Mapping*). Los sistemas SLAM permiten reconstruir escenas a medida que una cámara avanza por ellas y, a su vez, conocer la posición de dicha cámara.

La importancia de estos sistemas se debe, por un lado, a la reconstrucción obtenida de la escena y, por otro lado, a que permiten orientarse en un entorno a cualquier



(a) Reconstrucción de un tramo del aparato digestivo humano, [5].

(b) Reconstrucción de un monumento, [1].

Figura 1.2: Ejemplos de reconstrucciones de entornos en medicina, (a), y de exteriores, (b).

dispositivo que incluya una cámara y un computador. Es decir, los sistemas SLAM proporcionan un modelo del mundo y la localización dentro de él a sistemas computacionales sensorizados, lo cual es esencial para tareas que impliquen razonamiento espacial como, por ejemplo, la navegación autónoma.

La Figura 1.3 muestra un ejemplo de reconstrucción obtenida mediante un sistema SLAM. En gris se representa la reconstrucción del espacio de interés y en rojo la trayectoria de la cámara que lo ha recorrido. En este caso, el dispositivo que ha recorrido la escena puede navegar libremente sin colisionar con el edificio. Además, la reconstrucción puede utilizarse con los mismos propósitos que la mostrada en la Figura 1.2b.

Tal y como se ha mencionado, los sistemas SLAM solo necesitan de una cámara y un computador por lo que pueden ser implementados en multitud de dispositivos como, por ejemplo, robots, teléfonos móviles, vehículos autónomos o vehículos aéreos no tripulados (UAV, del inglés *Unmanned Aerial Vehicle*).

Estos sistemas pueden ser monoculares si utilizan una única cámara para tomar las imágenes o estéreo si utilizan dos. Los sistemas estéreo simulan la visión binocular humana y, mediante triangulación de los puntos observados por las dos cámaras, se pueden conocer las dimensiones del mapa y trayectoria reconstruidos. Sin embargo, estas dimensiones son desconocidas en el caso de los sistemas monoculares, es decir, la trayectoria y el mapa se pueden reconstruir a una escala desconocida.

Esta ambigüedad en la escala es uno de los principales problemas de los sistemas SLAM monoculares y, por lo tanto, ha sido uno de los más abordados. Para resolver

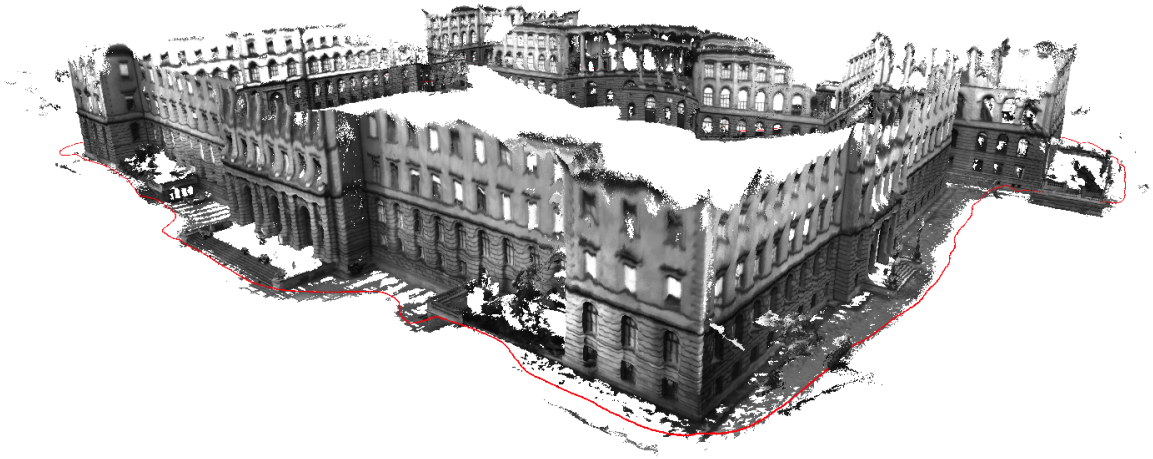


Figura 1.3: Ejemplo de aplicación del sistema SLAM presentado en [27]. Reconstrucción 3D de un entorno (en gris) y estimación de la trayectoria de la cámara (en rojo).

dicha ambigüedad es necesario, además de las imágenes captadas por la cámara, el uso de información adicional. Dicha información puede ser conocida a priori como, por ejemplo, alguna dimensión del entorno o puede obtenerse al incluir la medición de nuevas variables durante el proceso. Una de las técnicas más extendidas consiste en realizar medidas inerciales del movimiento de la cámara dando lugar así a los sistemas VI-SLAM (del inglés *Visual-Inertial* SLAM).

Los sistemas VI-SLAM se implementan en dispositivos que incluyen una cámara monocular y una unidad de medición inercial (IMU, del inglés *Inertial Measurement Unit*). Mediante la información inercial (IMU) se puede calcular la trayectoria del sensor, lo que unido a la información visual (cámara monocular) y el uso de triangulación, permite reconstruir la escena y calcular su escala real.

El uso de sistemas VI-SLAM para resolver la ambigüedad en la escala es una de las soluciones más utilizadas puesto que tanto las cámaras monoculares como las IMUs son sensores económicos, ligeros y de pequeño tamaño. Estas características permiten incluirlos, por ejemplo, en teléfonos móviles y en gafas de realidad aumentada (AR, del inglés *Augmented Reality*) y realidad virtual (VR, del inglés *Virtual Reality*).

Además, la popularidad de los sistemas VI-SLAM también se debe a la complementariedad de los sensores monoculares e inerciales. Por un lado, las medidas visuales tomadas con un sensor exteroceptivo, como es la cámara monocular, evitan

la deriva propioceptiva y, además, proveen información del entorno excelente para su reconstrucción. Por otro lado, las IMUs registran información de su propio movimiento y, por lo tanto, del de la cámara ya que mantienen una posición relativa fija. Además, la alta frecuencia a la que funcionan las IMUs hace que se pueda disponer de información fiable incluso en situaciones con movimientos agresivos.

Prueba de la popularidad de los sistemas VI-SLAM es que grandes compañías como Google o Apple (entre otras) han dedicado gran esfuerzo a desarrollar sistemas como Tango o ARKit, respectivamente, basados en sistemas VI-SLAM. La Figura 1.4 muestra dos ejemplos de realidad aumentada usando cada una de estas librerías.

(a) Sistema *ARKit* (b) Sistema *Tango*

Figura 1.4: Ejemplos de aplicaciones de AR basados en sistemas VI-SLAM. (a) muestra una aplicación para decoración del hogar y (b) muestra un juego de realidad aumentada.

1.3. Inicialización de sistemas SLAM Visual-Inercial

Los sistemas SLAM y VI-SLAM son planteados como problemas de optimización no lineal en los que se calcula la posición de la cámara y de los objetos que la rodean. Esto implica la necesidad de una estimación o "semilla" inicial para poder resolver el problema de optimización. El cálculo de dicha semilla es crítico para la convergencia del problema y se calcula durante el proceso de inicialización. Esto convierte al proceso de inicialización en uno de los más críticos para garantizar el adecuado funcionamiento de estos sistemas.

Los procesos de inicialización de los sistemas VI-SLAM requieren de una serie de imágenes de la escena así como de las medidas inerciales tomadas por la IMU y

permiten estimar el movimiento inicial de la cámara y la escala de la escena a partir de la localización de una serie de puntos del entorno, Figura 1.5.

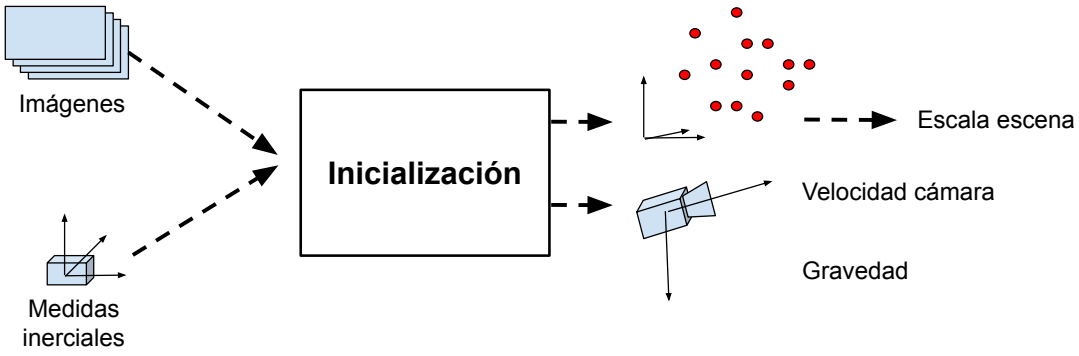


Figura 1.5: Esquema global de las entradas y salidas del sistema de inicialización. Se trata de una formulación que utiliza imágenes y medidas inerciales (velocidad angular y aceleración lineal) y devuelve la escala de la escena, la velocidad inicial de la cámara y el vector de la gravedad.

Por un lado, los sensores inerciales proporcionan medidas indirectas de la velocidad angular y de la aceleración lineal (incluida la correspondiente a la gravedad). Estas medidas están afectadas de un sesgo o *bias* aditivo así como de un ruido de alta frecuencia. En general, el tratamiento de los sensores inerciales no es sencillo debido a la necesidad de integrar sus medidas para estimar la posición y orientación del dispositivo, la gravedad y los *bias*.

Por otro lado, las imágenes son utilizadas para la extracción de características visuales de la escena (también llamadas puntos de interés o *features*). La Figura 1.6 muestra el emparejamiento de características entre dos imágenes de una misma escena.

Si bien los avances en técnicas SLAM durante las últimas dos décadas han permitido alcanzar una robustez y precisión adecuadas para muchas aplicaciones [14, 17, 20], los sistemas VI-SLAM no han alcanzado todavía los niveles de fiabilidad requeridos en muchos casos debido, en gran medida, a los problemas y limitaciones que presenta la etapa de inicialización [2, 4, 8, 29].

El actual estado del arte de las inicializaciones de sistemas VI-SLAM requiere características visuales que sean localizadas y emparejadas en todas las imágenes utilizadas para la inicialización. Es decir, es necesario que los puntos de interés sean

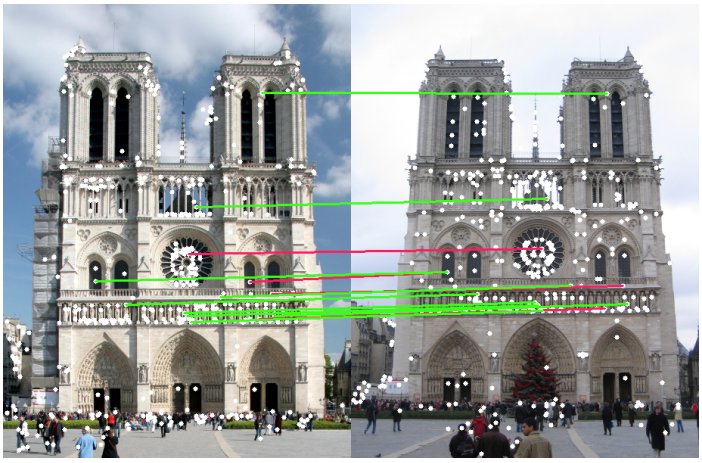
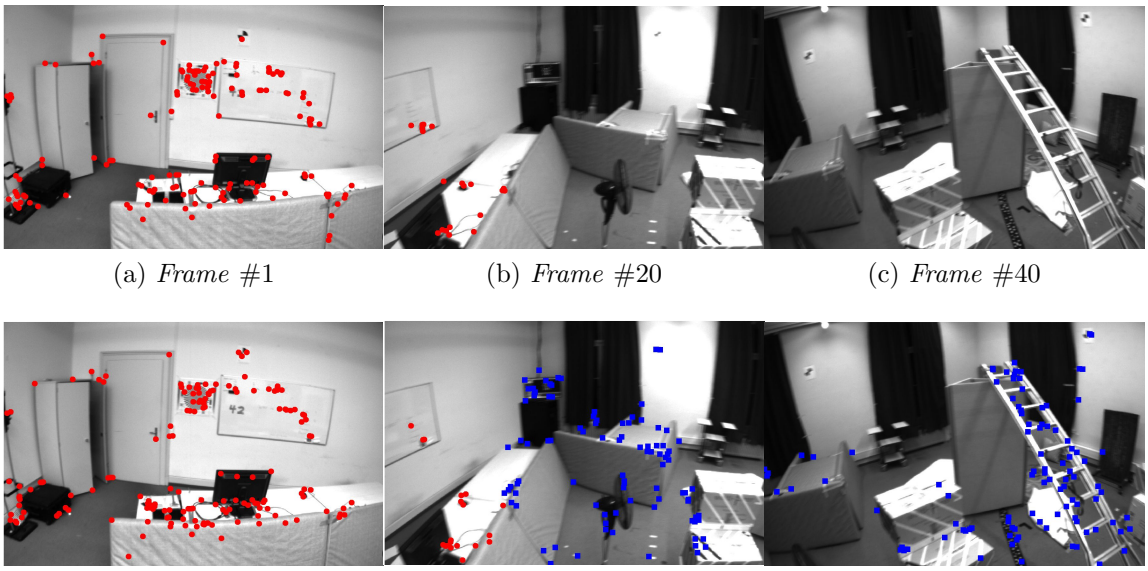


Figura 1.6: Emparejamiento de características visuales entre dos imágenes de una misma escena. Cada línea une la localización de un mismo punto de interés (*feature*) en los dos *frames*. En rojo se muestran emparejamientos espurios. Ejemplo tomado de [31].

comunes a todas las imágenes, lo que supone una limitación muy seria. Las Figuras 1.7a-1.7c muestran las características utilizadas por [13] en una secuencia del *dataset* utilizado en este trabajo, [3]. Nótese que entre los *frames* 1 y 20 hay muy pocas características comunes a ambas imágenes (en rojo) y que entre los *frames* 1 y 40 no queda ninguna característica común. En el caso mostrado es tal el movimiento de la cámara que no existen puntos de la escena comunes a todas las imágenes y, por lo tanto, la inicialización falla.

En este trabajo se propone una nueva formulación lineal general para la inicialización de sistemas VI-SLAM. La formulación propuesta elimina la restricción impuesta por la necesidad de puntos comunes a todos los *frames* utilizados en la inicialización. Es decir, no es necesario que las características sean comunes a todas las imágenes sino que pueden ser emparejadas entre cualquier conjunto de *frames*. A modo de ejemplo, las Figuras 1.7d-1.7f muestran las características utilizadas por la formulación propuesta en el ejemplo anterior. Obsérvese que en los *frames* 20 y 40 se añaden nuevas características (en azul) que no son comunes a toda la secuencia. Al contrario que las formulaciones que componen el estado del arte, la propuesta consigue inicializar correctamente.

Con la formulación introducida en este trabajo se mejora en gran medida la robustez y precisión del actual estado del arte. Dicha formulación se construye a partir de la desarrollada por [13] y que a su vez se basa en el trabajo desarrollado por [19].



(a) *Frame* #1 (b) *Frame* #20 (c) *Frame* #40

(d) *Frame* #1 (e) *Frame* #20 (f) *Frame* #40

Figura 1.7: *Frames* de una secuencia extraída del *dataset* utilizado para la realización del trabajo. (a)-(c): inicialización según [13] donde es imposible conseguir buenos resultados a partir del *frame* #20 por falta de *features* comunes. (d)-(f): inicialización propuesta donde se observa la adición de nuevos *features* a partir del *frame* #20 (en azul), lo que permite inicializar el sistema.

La formulación propuesta no solo aumenta la tasa de éxito en la inicialización de un sistema visual-inercial sino que también mejora la precisión puesto que cuenta con un mayor número de *features* así como con una mejor distribución de los mismos en las imágenes.

Puede concluirse que el objetivo de este trabajo es formular una nueva inicialización que suponga una importante mejora de todo el sistema VI-SLAM en cuanto a robustez y precisión.

Para alcanzar este objetivo es necesario, en primer lugar, un estudio detallado de las inicializaciones actuales que permita identificar posibles mejoras. Posteriormente, se diseña y desarrolla la nueva formulación general para la inicialización. Una vez desarrollada la nueva inicialización, se plantean y realizan experimentos para evaluar su funcionamiento y cuantificar las mejoras conseguidas. Para ello, se ha realizado una evaluación exhaustiva de la formulación propuesta en un *dataset* público [3] y se

han comparado los resultados con los obtenidos con la formulación de [13].

El resto de la memoria se estructura como sigue. El capítulo 2 detalla los trabajos relacionados. El capítulo 3 presenta y detalla la formulación propuesta para la inicialización así como una serie de modelos y definiciones utilizados en su desarrollo. A continuación, el capítulo 4 recoge los experimentos realizados. Por último el capítulo 5 muestra las conclusiones del trabajo realizado y discute el trabajo futuro.

Capítulo 2

Trabajo relacionado

A pesar de la popularidad de los sistemas VI-SLAM, tradicionalmente su inicialización no ha sido casi nunca abordada en los trabajos y estudios que componen el estado del arte. Por ejemplo, algunos trabajos e investigaciones importantes como [7, 16, 23] ni siquiera mencionan la etapa de inicialización. Otros, como [12, 15, 22] hacen referencia a ella sin detenerse en su análisis o evaluación. La mayor parte de estos trabajos asumen sistemas ya inicializados correctamente.

Los escasos trabajos que han reflejado la etapa de inicialización simplemente mencionan una serie de asunciones o técnicas heurísticas para dar con un estado inicial según el cual sus sistemas convergen adecuadamente. Si bien estas técnicas muestran un buen comportamiento, no demuestran su validez para escenarios más allá de los experimentos que proponen estos trabajos ni comparan su funcionamiento frente a otras técnicas o formulaciones.

Ha sido únicamente en los trabajos más recientes donde se ha abordado de manera más concisa la etapa de inicialización. En estos últimos se proponen técnicas concretas para la inicialización y se reconoce que se trata de una etapa crítica que todavía presenta problemas (p. ej. [33, 25]).

Los trabajos que abordan el problema de la inicialización pueden ser clasificados en dos grandes grupos en función de cómo enfocan el problema y su solución: aquellos con un enfoque global y aquellos con un enfoque progresivo. A continuación se resumen el funcionamiento y diferencias de ambos tipos de solución.

2.1. Inicialización progresiva

Algunos trabajos, como [21, 24], realizan la estimación visual-inercial del estado del sistema mediante una técnica compuesta por sucesivas etapas. Es decir, la idea es la división de la inicialización en varios subproblemas.

En líneas generales, primero estiman la rotación y la dirección de traslación de la cámara únicamente con el uso de medidas visuales (imágenes). A continuación, se estima el *bias* del giróscopo (error sistemático) a partir de la rotación calculada y de la integración de las medidas del giróscopo. Por último se calculan la gravedad,

la escala de la escena, la velocidad inicial del dispositivo y el *bias* del acelerómetro mediante los resultados anteriores y las medidas del acelerómetro. Este proceso queda ilustrado en la Figura 2.1.

Este tipo de inicialización cuenta con las mismas limitaciones que la inicialización de los sistemas SLAM monoculares puesto que dependen enteramente del primer paso, es decir, del uso de medidas visuales para obtener la rotación y la dirección del dispositivo. Por este motivo estas inicializaciones pueden tener problemas de robustez y precisión además de requerir tiempos relativamente altos para ser llevadas a cabo. Por ejemplo, [21] muestra tiempos de hasta 10 segundos.

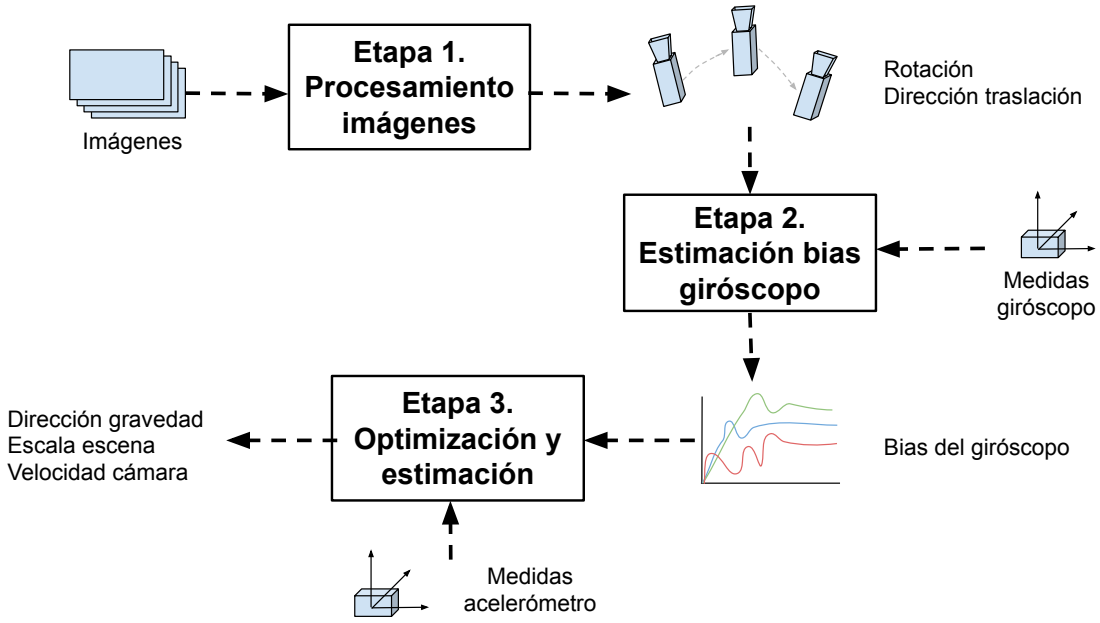


Figura 2.1: Proceso aproximado de una inicialización progresiva. Se han propuesto diferentes inicializaciones de este tipo ([21, 24]) aunque el esquema general es común.

2.2. Inicialización global (*closed-form*)

Estas formulaciones proponen una solución global o solución de forma cerrada (comúnmente *closed-form* en inglés) para determinar el estado inicial mediante el uso de medidas tanto visuales como inerciales. Se trata de un enfoque global de la inicialización sin dividirla en subproblemas. La formulación propuesta en este trabajo se enmarca en esta categoría, al igual que los algoritmos en los que se basa, [13, 18, 19].

La resolución del problema desde este enfoque global se ha tratado siempre mediante un sistema lineal de ecuaciones donde las incógnitas componen el estado inicial (gravedad, velocidad inicial, escala de la escena y *bias*) y las variables son obtenidas del procesamiento de las imágenes (emparejamiento de *features*) y medidas de la IMU, Figura 2.2. En el capítulo 3.4 se detalla el caso concreto de este trabajo.

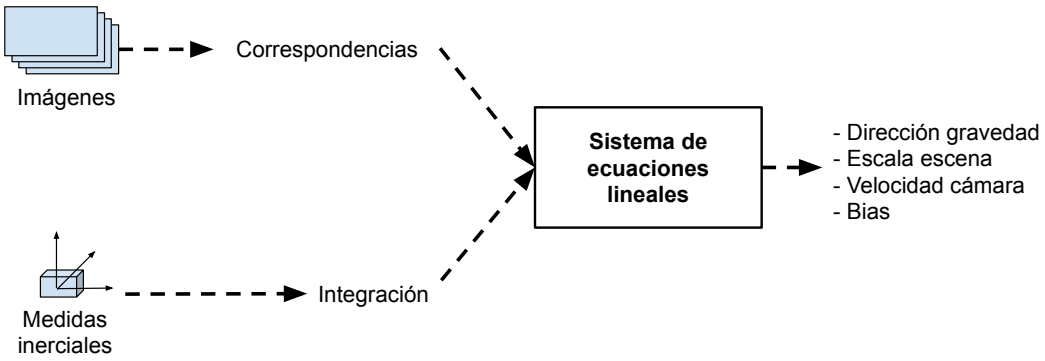


Figura 2.2: Proceso aproximado de una inicialización global (*closed-form*).

En general, estas inicializaciones muestran mejores resultados tal y como refleja [13], donde se necesitan tiempos de unos 2 segundos para inicializar el sistema. Sin embargo, estas formulaciones también cuentan con limitaciones. Quizás la más notable es el hecho de que precisan de *features* comunes a todos los *frames* durante la inicialización. Como ya ha sido mencionado, esto hace que ante movimientos o escenas complicadas, la inicialización falle y sea necesario comenzar de nuevo el proceso.

La formulación propuesta supera esta limitación ya que permite al sistema incluir nuevos *features* a medida que avanza la inicialización. De esta manera, se puede inicializar con movimientos relativamente bruscos, giros amplios, escenas con escaso número de *features* y en casos con mala iluminación.

Otro de los problemas asociados a esta solución se debe al cálculo del *bias* del giróscopo. Como se explica más adelante, el *bias* es calculado minimizando el residuo de una función de coste y, tal y como refleja [13], la naturaleza de dicha función puede causar la divergencia del *bias* calculado. Además, los trabajos propuestos hasta ahora parten de un valor inicial del *bias* en lugar de estimarlo desde el comienzo. En el apartado 3.6 se detalla cómo se ha abordado este problema.

Capítulo 3

Inicialización de estado

El objetivo de la inicialización es estimar el movimiento inicial del dispositivo así como la localización de una serie de puntos para poder realizar una posterior optimización no lineal. El resultado de dicha optimización es la localización (trayectoria) de la cámara, el mapa de la escena y el estado del dispositivo (velocidad, *bias* y vector gravedad).

En este capítulo se detalla la formulación propuesta para la inicialización así como una serie de conceptos previos necesarios para el desarrollo de la formulación.

3.1. Modelo de la cámara

En esta sección se presenta el modelo de cámara utilizado para el desarrollo de la formulación propuesta. Se ha utilizado el modelo de cámara oscura o cámara *pinhole* puesto que se trata de un modelo simple y aplicable a diferentes tipos de cámara.

La Figura 3.1 muestra la proyección de un punto del espacio 3D, $\mathbf{X}$, en la imagen 2D, $\mathbf{x}$, según el modelo *pinhole*. El centro óptico de la cámara, $\mathbf{c}$, se considera como origen del sistema de coordenadas y el plano sobre el que se forma la imagen se sitúa ortogonal al eje z y a una distancia positiva f del centro óptico. La distancia f es la distancia focal de la cámara. El punto donde el eje z corta al plano de la imagen se denomina punto principal, $\mathbf{p}$, y suele ser muy próximo al centro de la imagen (pueden existir pequeñas desviaciones en cuyo caso hay que modelar).

En realidad, el plano de la imagen se encuentra detrás del centro óptico de la cámara ($z = -f$). Sin embargo, como norma general y para mayor claridad, el modelo *pinhole* lo asume delante ($z = f$) teniendo en cuenta las consiguientes modificaciones en las expresiones.

Según el modelo *pinhole*, la proyección en la imagen, $\mathbf{x}$, de un punto del espacio, $\mathbf{X}$, coincide con la intersección con el plano imagen del rayo que une el centro óptico con el punto $\mathbf{X}$, Figura 3.1. Por equivalencia de triángulos se puede deducir que un punto del espacio 3D con coordenadas $\mathbf{X} = (X, Y, Z)$ queda proyectado sobre el plano imagen en el punto $\mathbf{x} = (fX/Z, fY/Z, f)$. Si se omite la tercera coordenada del punto proyectado, se obtienen las coordenadas 2D del punto en la imagen, $(fX/Z, fY/Z)$.

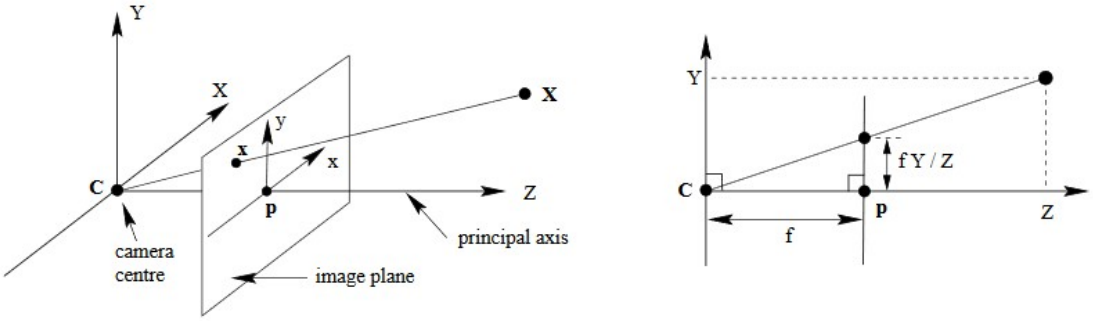


Figura 3.1: Modelo de cámara *pinhole*. Figura extraída de [10].

Es muy común el uso de coordenadas homogéneas para expresar de forma matricial la proyección descrita entre el espacio 3D y la imagen tal y como sigue:

$$\begin{pmatrix} fX \\ fY \\ Z \end{pmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{1}$$

$$\mathbf{x} = \mathtt{P}\mathbf{X}$$

En caso de que el centro de la imagen no coincida con el punto principal, $\mathbf{p}$, la expresión anterior queda ligeramente modificada.

$$\begin{pmatrix} fX + Zpx \\ fY + Zpy \\ Z \end{pmatrix} = \begin{bmatrix} f & 0 & px & 0 \\ 0 & f & py & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{2}$$

Donde px y py es el *offset* del punto principal, $\mathbf{p}$, con respecto al centro de la imagen. De esta forma, las nuevas coordenadas en la imagen toman la forma $(fX/Z + px, fY/Z + py)$

La matriz $\mathtt{P}$ se denomina matriz de proyección y suele expresarse de la siguiente forma:

$$\mathbf{x} = \mathtt{P}\mathbf{X} = \mathtt{K}[\mathtt{I}|\mathtt{0}]\mathbf{X} \tag{3}$$

Donde la matriz K de dimensiones 3x3 se denomina matriz de calibración y es la que contiene la información intrínseca a la cámara.

$$\mathtt{K} = \begin{bmatrix} f & 0 & px \\ 0 & f & py \\ 0 & 0 & 1 \end{bmatrix} \tag{4}$$

Para el proceso de inicialización, se debe tener en cuenta que la cámara se mueve y observa los puntos del entorno desde diferentes posiciones. Por ello, es necesario definir cómo se van a manejar las traslaciones y rotaciones de la cámara.

Los puntos del espacio 3D se expresan respecto a un sistema de referencia global, **O**. Dicho sistema de referencia y el de las sucesivas cámaras que observan los puntos se relacionan mediante una rotación y una traslación (Figura 3.2). La rotación se expresa mediante una matriz de rotación 3x3, R, y la traslación mediante un vector 3x1, t. El origen del sistema de referencia de cualquier cámara coincide con su centro óptico, **c**. En la Figura 3.2 se muestra la relación entre el sistema de referencia global, **O**, y el de una cámara cualquiera, j.

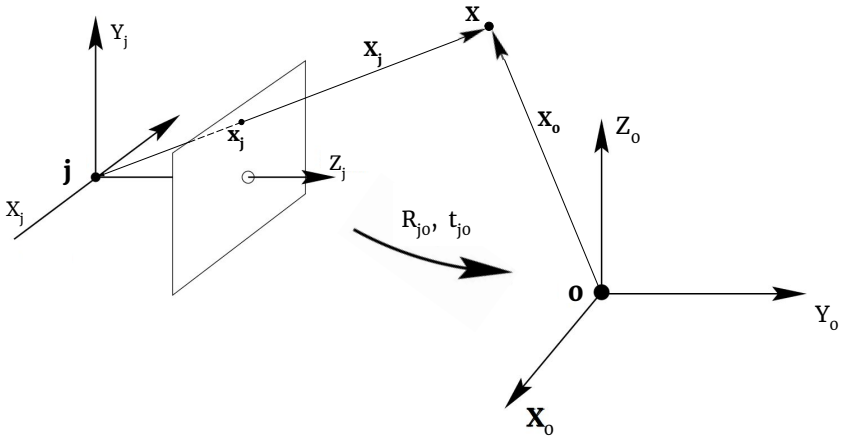


Figura 3.2: Transformación entre los sistemas de coordenadas **O** y j mediante los parámetros extrínsecos a la cámara. Adaptación de [10].

Para expresar puntos del espacio 3D en el sistema de referencia de la cámara se emplea la siguiente transformación:

$$\mathbf{X_j} = \begin{bmatrix} \mathtt{R_{j0}} & \mathtt{t_{j0}} \\ 0 & 1 \end{bmatrix} \mathbf{X_0} \tag{5}$$

$$\mathbf{X_j} = \mathtt{T_{j0}}\mathbf{X_0}$$

Donde $\mathbf{X_j}$ y $\mathbf{X_0}$ son las coordenadas homogéneas del punto en el sistema de referencia de la cámara j y global, respectivamente, y $\mathtt{T_{j0}}$ la matriz de transformación homogénea. En el sistema de referencia global, $\mathbf{O}$, la posición del centro óptico de la cámara j, $\mathbf{c_j}$, viene dado por:

$$\mathbf{c_j} = -\mathtt{R}_{\mathtt{j0}}^{\mathtt{T}}\mathtt{t_{j0}} \tag{6}$$

Teniendo en cuenta la transformación entre ambos sistemas de referencia, la Expresión 3 se ve modificada como sigue:

$$\mathbf{x_j} = \mathtt{K_j}[\mathtt{I}|\mathtt{0}]\mathtt{T_{j0}}\mathbf{X_0} \tag{7}$$

Donde $\mathbf{x_j}$ es la proyección del punto $\mathbf{X_0}$ en la imagen. La rotación y traslación se conocen como los parámetros extrínsecos de la cámara y junto con los intrínsecos (matriz de calibración) forman la matriz de proyección más general de la cámara.

$$\mathtt{P_j} = \mathtt{K_j}[\mathtt{I}|\mathtt{0}]\mathtt{T_{j0}}$$

3.2. Modelo de la IMU

La IMU incorpora un acelerómetro y un giróscopo con los que puede medir la aceleración lineal y la velocidad angular en los tres ejes. A continuación se detalla el modelo de IMU que se ha asumido para desarrollar la formulación.

El modelo de la IMU establece los términos de los que se componen las medidas tomadas por el sensor. El modelo adoptado incluye dos tipos de errores en las medidas, [32]: un error aleatorio que varía de forma rápida y un error sistemático (*bias*) que varía lentamente. Tanto la velocidad angular como la aceleración lineal en los tres ejes se ven afectadas por los mismos errores. Así, las medidas del sensor se pueden expresar como:

$$\tilde{\boldsymbol{\omega}}(t) = \boldsymbol{\omega}(t) + \mathbf{b}_g + \boldsymbol{\eta}_g(t) \tag{8}$$

$$\tilde{\boldsymbol{a}}(t) = \boldsymbol{a}(t) + \mathbf{b}_a - \mathbf{g}(t) + \boldsymbol{\eta}_a(t) \tag{9}$$

Donde:

- $\tilde{\boldsymbol{\omega}}(t)$ y $\tilde{\boldsymbol{a}}(t)$ son las medidas de la velocidad angular y aceleración lineal, respectivamente, que toma la IMU.

- $\boldsymbol{\omega}(t)$ y $\boldsymbol{a}(t)$ son la velocidad angular y aceleración lineal reales de la IMU, respectivamente.
- $\mathbf{b}_g$ y $\mathbf{b}_a$ es el *bias* que afecta al giróscopo y al acelerómetro, respectivamente. Nótese que durante el intervalo de tiempo que se realiza la inicialización el *bias* puede considerarse constante ya que su variación es muy lenta.
- $\boldsymbol{\eta}_g(t)$ y $\boldsymbol{\eta}_a(t)$ es el error aleatorio que afecta al giróscopo y al acelerómetro, respectivamente. Este error aleatorio se puede entender como un ruido blanco gaussiano modelado de manera heurística mediante una normal de media cero y desviación típica σ_g (giróscopo) o σ_a (acelerómetro).
- $\mathbf{g}(t)$ es el vector de la gravedad. Téngase en cuenta que el vector de la gravedad depende del tiempo ya que la cámara puede rotar y con ella su sistema de referencia local al que está referida la gravedad.

Uno de los aspectos más importantes a tener en cuenta cuando se trabaja con sensores de diferente naturaleza, es la manera de integrarlos para trabajar con ellos de manera conjunta. En el caso de la cámara y la IMU, una de las dificultades más importantes surge de la diferencia entre sus tiempos de muestreo. Por ejemplo, en el *dataset* empleado en este trabajo, [3], se utiliza una cámara que toma imágenes a una frecuencia de 20 Hz y una IMU que registra medidas a 200 Hz, Figura 3.3.

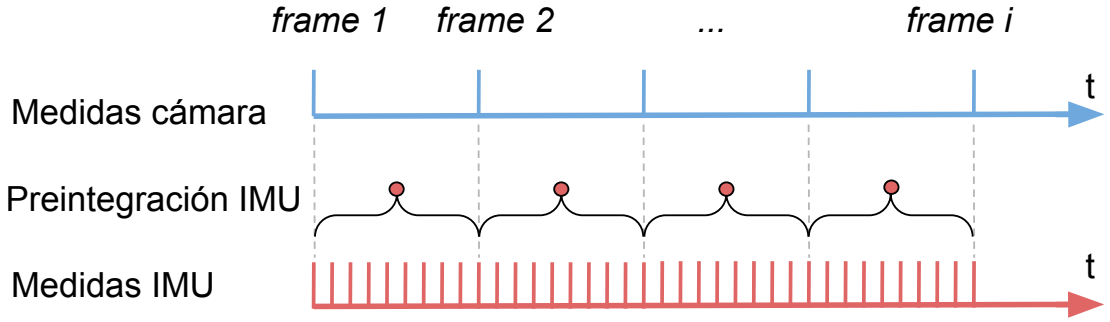


Figura 3.3: Relación temporal en la toma de datos de la cámara y de la IMU. Se supone en todo momento que las capturas de imágenes y de medidas inerciales están perfectamente sincronizadas. La cámara y la IMU utilizadas para los experimentos tienen unas frecuencias de 20Hz y 200Hz, respectivamente.

Para facilitar el trabajo con medidas tomadas a diferentes frecuencias y disminuir la cantidad de datos a manejar, se realiza una preintegración de los datos de la IMU

comprendidos entre dos *frames* consecutivos. La preintegración permite obtener un dato de la IMU asociado al intervalo entre *frames* tal y como muestra la Figura 3.3. Las expresiones utilizadas para la preintegración se desarrollan en la Sección 3.5 junto con la formulación propuesta.

3.3. Generalidades de la inicialización propuesta

El algoritmo propuesto permite calcular un vector de estado compuesto por la velocidad inicial de la cámara, la gravedad, el *bias* del giróscopo y la distancia entre la cámara y una serie de *features*. Con esta información puede obtenerse la trayectoria de la cámara durante la inicialización así como una nube de puntos y escala de la escena. Para ello, la información de partida (*input*) son los $\mathcal{M}$ *frames* grabados por la cámara así como las medidas inerciales correspondientes captadas por la IMU.

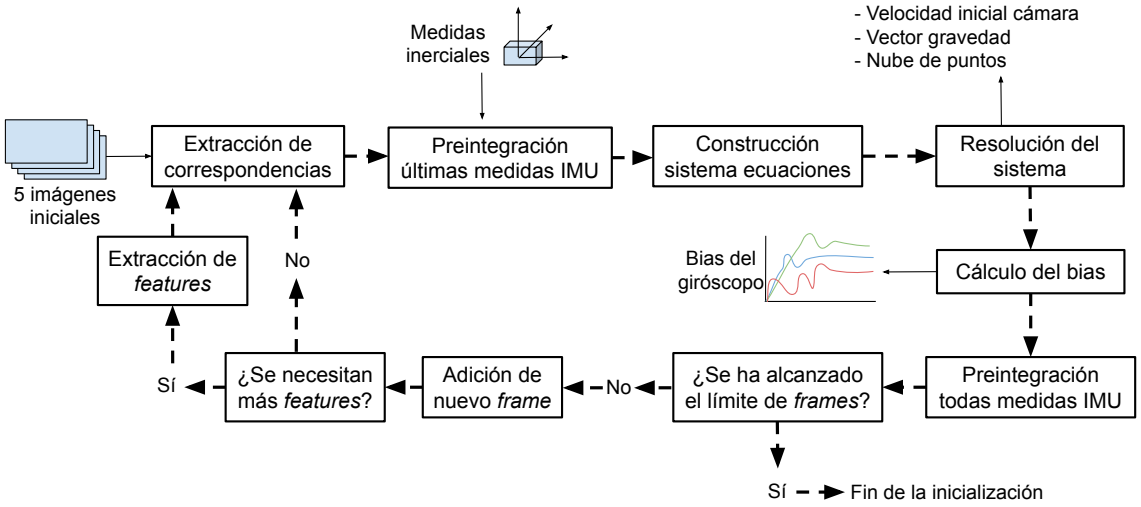


Figura 3.4: Diagrama de bloques del algoritmo de la inicialización propuesta.

Antes de desarrollar en detalle cada uno de los procesos en los que se divide la inicialización de estado, la Figura 3.4 resume el funcionamiento general del algoritmo desarrollado. En primer lugar se procesan 5 imágenes y se obtienen correspondencias entre ellas. Una vez obtenidas las correspondencias, se preintegran los datos de la IMU correspondientes al intervalo de tiempo entre las últimas imágenes procesadas. Con las correspondencias y la preintegración, se construye y resuelve el sistema de ecuaciones lineales cuya solución es el vector de estado. Tras haber resuelto el sistema, se calcula el *bias* del giróscopo mediante la minimización de una función de coste.

Habiendo calculado el *bias*, este se utiliza para preintegrar todos los datos de la IMU, desde la primera imagen hasta la última que haya sido incluida hasta el momento. Una vez hecho esto, se procesa una nueva imagen obteniéndose correspondencias comunes entre ésta y las anteriores y repitiendo todo el proceso hasta alcanzar el límite de *frames* para la inicialización. Cada vez que se procesa una nueva imagen se comprueba si es necesario extraer nuevos *features* y así obtener correspondencias con las imágenes siguientes.

A continuación se detallan las etapas en las que se divide la inicialización de estado en el mismo orden en el que se llevan a cabo: procesamiento de las imágenes, inicialización y cálculo del *bias*.

3.4. Procesamiento de las imágenes

El primer paso es proyectar los puntos observados del espacio 3D (*features*) en las imágenes para obtener sus coordenadas en píxeles según los principios expuestos en la Sección 3.1. Para obtener las coordenadas en píxeles de todos los *features* en todas las imágenes se utilizan dos algoritmos.

En primer lugar, se utiliza el algoritmo propuesto por [28] para detectar los *features* y obtener sus coordenadas solo en la primera imagen. El detector utilizado extrae puntos esquina, es decir, los *features* extraídos se localizan en píxeles con variaciones de gradiente en, al menos, dos direcciones ortogonales.

En segundo lugar, se calculan las coordenadas de los *features* en el resto de imágenes (correspondencias). Cuando la cámara se mueve, las coordenadas en píxeles de los puntos 3D se modifican (Figura 3.5). Para encontrar la nueva localización del punto en el siguiente *frame* se aplica el algoritmo Kanade-Lucas-Tomasi (KLT) [30]. Este algoritmo de seguimiento de *features* calcula el posible desplazamiento de cada punto con respecto al *frame* anterior a partir de los gradientes en el entorno local de cada *feature* en la imagen. La Figura 3.5 muestra el emparejamiento de características visuales (*features*) en dos *frames* extraídos de los experimentos realizados.

El algoritmo utilizado para la extracción de *features*, [28], fue desarrollado para que el desplazamiento de dichos *features* en la imagen fuese fácilmente calculable por el algoritmo de seguimiento propuesto por [30]. Es por esto que ambos algoritmos presentan muy buen comportamiento cuando se implementan conjuntamente.

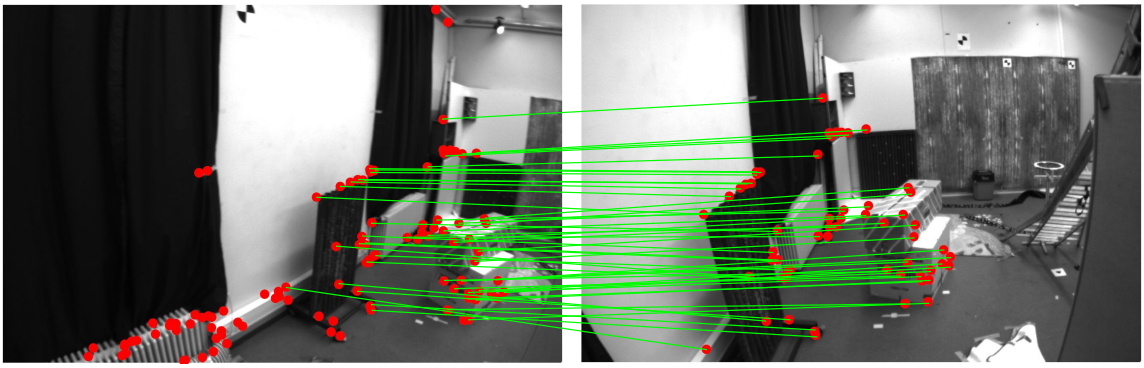


Figura 3.5: Aplicación de [28] para obtener características (*features*) en la primera imagen de la secuencia (izquierda) y de [30] para encontrar su localización en las imágenes sucesivas (derecha). Las líneas verdes unen la localización de cada *feature* en ambos *frames*.

Así pues, la información a extraer de las imágenes necesaria para la inicialización son las coordenadas en píxeles de todas las correspondencias en cada una de las imágenes.

Conforme la cámara se mueve por la escena, se van perdiendo correspondencias y aparecen zonas sin *features* (Figura 1.7). Para mantener un número suficiente de correspondencias en todas la imágenes, se vuelven a extraer *features* cuando el número de correspondencias cae por debajo de un determinado umbral o se crean grandes áreas de la imagen sin correspondencias.

Los umbrales que marcan la extracción de nuevos *features* son totalmente modificables sin afectar notablemente los resultados. En el caso de los experimentos presentados en el Capítulo 4, se ha fijado un mínimo de 30 correspondencias en secuencias clasificadas como fáciles y medias y de 10 en secuencias difíciles. Cuando las correspondencias caen por debajo de dichos límites, se vuelven a extraer nuevos *features*. Un proceso similar se sigue en cuanto a las regiones de los *frames* que quedan sin correspondencias. En secuencias fáciles y medias, se extraen *features* cada vez que se tiene más de un tercio de la imagen sin correspondencias mientras que en el caso de las secuencias difíciles se extraen cada vez que se tiene medio *frame* sin correspondencias.

También se han establecido límites superiores de forma que nunca se extraen más de 200 *features*. Hay que tener en cuenta que en la práctica no se encuentran correspondencias para todos los *features* extraídos.

Por otro lado, no se tienen en cuenta todos los *frames* captados por la cámara sino que se utiliza uno de cada dos por razones de eficiencia. En secuencias en las que el desplazamiento entre imágenes no es excesivo (fáciles y medias en el *dataset* utilizado) podría utilizarse uno de cada tres o cada cuatro *frames*.

Durante la realización del trabajo no se han observado *features* espurios (de ahora en adelante, *outliers*), por lo que no se ha incluido ningún algoritmo para su detección.

3.5. Inicialización

La inicialización propuesta supone una generalización de la formulación mostrada en [19] puesto que en este trabajo se elimina la necesidad de un único *frame* de referencia respecto del cual encontrar correspondencias. De esta forma, se permite la utilización de correspondencias entre cualquier par de *frames* $\mathcal{F}_j$ y $\mathcal{F}_k$. En esta sección se establecen unas definiciones y se presentan una serie de ecuaciones que dan lugar a la formulación general lineal propuesta.

Para cada *feature* i ($1 \leqslant i \leqslant N_{jk}$) que sea visto en dos *frames* cualesquiera j y k ($1 \leqslant j < k \leqslant M$) y asumiendo que las medidas inerciales se toman de manera sincronizada con los *frames* según la Figura 3.3, se cumple la siguiente ecuación que sirve como punto de partida para desarrollar la formulación propuesta y que es detallada en las siguientes páginas:

$$-\mathbf{g}_j \frac{t_{jk}^2}{2} - \mathbf{v}_j t_{jk} + \lambda_j^i \boldsymbol{\mu}_j^i - \mathtt{R}_{jk} \lambda_k^i \boldsymbol{\mu}_k^i = \mathbf{s}_{jk} \tag{10}$$

Donde:

- $\mathcal{C}_j$ y $\mathcal{C}_k$ son las posiciones de la cámara cuando se toman los *frames* $\mathcal{F}_j$ y $\mathcal{F}_k$, en tiempos t_j y t_k
- N_{jk} es el número de correspondencias entre las imágenes $\mathcal{F}_j$ y $\mathcal{F}_k$
- M es el número total de *frames* utilizados durante la inicialización
- $t_{jk} = t_k - t_j$ es el intervalo de tiempo entre los instantes t_j y t_k
- $\boldsymbol{\mu}_j^i$ y $\boldsymbol{\mu}_k^i$ son los vectores unitarios cuya dirección es desde el centro óptico de la cámara en los instantes j^o y k^o, respectivamente, hacia el *feature* 3D $\mathbf{X}_i$ en los respectivos sistemas de referencia locales de la cámara

- λ_j^i y λ_k^i son las distancias al punto $\mathbf{X}_i$ desde el centro óptico de la cámara en los instantes j^o y k^o respectivamente
- $\mathbf{v}_j$ es la velocidad de la cámara en el instante j^o en su sistema de referencia local
- $\mathbf{g}_j$ es el vector de la gravedad en el sistema de referencia local de la cámara j^a, que puede ser transformado al sistema de referencia local de la 1^a cámara según $\mathbf{g}_j = \mathtt{R}_{j1}\mathbf{g}_1$
- $\mathtt{R}_{jk}$ es la matriz de rotación relativa entre las cámaras $\mathcal{C}_j$ y $\mathcal{C}_k$
- $\mathbf{s}_{jk}$ es la preintegración de los datos del acelerómetro desde t_j hasta t_k

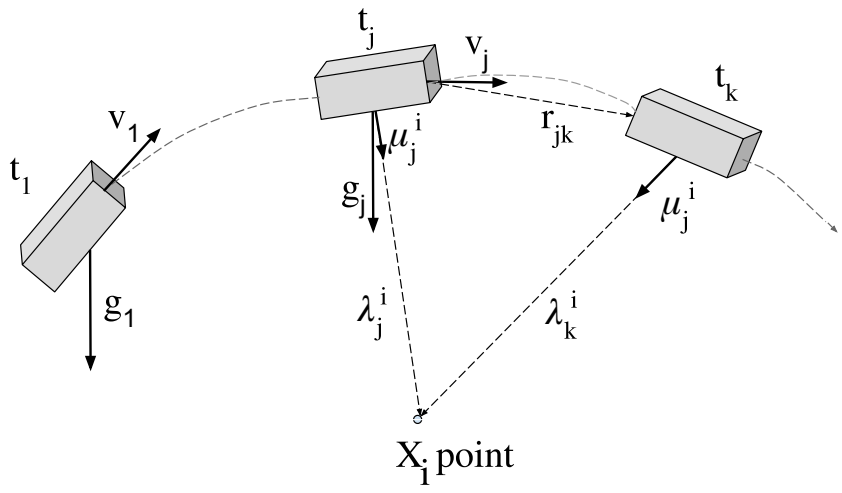


Figura 3.6: Esquema de la notación más relevante empleada en el trabajo.

En la Figura 3.6 se puede ver un esquema del problema que incluye las variables más importantes para la inicialización. Con la ayuda de dicha figura se puede realizar la siguiente interpretación geométrica de la Ecuación 10: $\lambda_j^i \boldsymbol{\mu}_j^i$ es el vector que une el centro óptico de la cámara $\mathcal{C}_j$ con el *feature* $\boldsymbol{X}_i$ en el sistema de referencia local de la cámara $\mathcal{C}_j$; $\mathtt{R}_{jk}\lambda_k^i \boldsymbol{\mu}_k^i$ es el vector que une el centro óptico de la cámara $\mathcal{C}_k$ con el *feaure* $\boldsymbol{X}_i$ en el sistema de referencia local de la cámara $\mathcal{C}_j$.

Restando ambos vectores tal y como se hace en la Ecuación 10, se puede obtener la traslación de la cámara, $\mathbf{r}_{jk}$, entre los instantes j y k.

$$\mathbf{r}_{jk} = \lambda_j^i \boldsymbol{\mu}_j^i - \mathtt{R}_{jk} \lambda_k^i \boldsymbol{\mu}_k^i \tag{11}$$

Tal y como propone [19], la traslación $\mathbf{r}_{jk}$ también puede ser obtenida mediante la siguiente ecuación cinemática:

$$\mathbf{r}_{jk} = \mathbf{v}_j t_{jk} + \int_{t_j}^{t_k} (t_k - \tau) \mathtt{R}_{j\tau} \mathbf{a}_\tau d\tau \tag{12}$$

Donde $\mathbf{a}_\tau$ es la aceleración lineal de la cámara en su referencia local en el tiempo τ y $\mathtt{R}_{j\tau}$ es la rotación relativa de la cámara entre los instantes t_j y τ.

Según el modelo del acelerómetro, Expresión 9, la aceleración lineal de la cámara puede reescribirse como:

$$\boldsymbol{a}_\tau = \mathbf{g}_\tau + \tilde{\boldsymbol{a}}_\tau - \mathbf{b}_a - \boldsymbol{\eta}_a \tag{13}$$

A partir de la Expresión 13, la Ecuación 12 puede reescribirse como sigue:

$$\mathbf{r}_{jk} = \mathbf{v}_j t_{jk} + \mathbf{g}_j \frac{t_{jk}^2}{2} + \mathbf{s}_{jk} \tag{14}$$

El término relativo a la preintegración de las medidas del acelerómetro presente en las Ecuaciones 10 y 14 puede ser aproximado como $\mathbf{s}_{jk} = \int_{t_j}^{t_k} (t_k - \tau) \mathtt{R}_{j\tau} (\tilde{\mathbf{a}}_\tau - \mathbf{b}_a) d\tau$. Se trata de la preintegración de las medidas rotadas de la aceleración entre los *frames* j y k. Finalmente, la Ecuación 10 se obtiene al igualar las Ecuaciones 11 y 14.

Por simplicidad, se establece el sistema de referencia local de la posición inicial de la cámara, $\mathcal{C}_1$, como sistema de referencia global, es decir, siguiendo la notación introducida en la Sección 3.1 se cumple $\mathcal{C}_1 = \mathcal{C}_0$. Además, para mayor claridad, se omiten de ahora en adelante los subíndices 1 (*p.ej.*, $\mathtt{R}_j \equiv \mathtt{R}_{1j}$).

A continuación se expone el cálculo del resto de variables presentes en la Ecuación 10 dando lugar finalmente a la formulación propuesta.

En cuanto a la rotación $\mathtt{R}_j$ desde la posición $\mathcal{C}_1$ a la $\mathcal{C}_j$, ésta puede ser obtenida a partir de la preintegración de la velocidad angular, $\boldsymbol{\omega}_\tau$, registrada por la IMU durante el intervalo t_{1j}. Para poder obtener una matriz de rotación 3x3 ($\mathtt{R}_j$) a partir de la velocidad angular 3x1 ($\boldsymbol{\omega}_\tau$), es necesario realizar la siguiente transformación entre espacios:

$$f : \mathbb{R}^3 \rightarrow so(3) \rightarrow SO(3)$$

El grupo de rotación $SO(3)$ es el formado por las matrices de rotación 3D y se define como:

$$SO(3) \doteq \{R \in \mathbb{R}^{3x3} : R^T R = I, det(R) = 1\}$$

El grupo $\mathbf{so}(3)$ es el álgebra de Lie de $SO(3)$ y se trata de un espacio lineal y tangente al espacio $SO(3)$ en la identidad, $\mathbf{I(3)}$ (Figura 3.7). Este espacio es muy útil para trabajar con pequeñas variaciones de la rotación como las producidas entre dos medidas consecutivas de la IMU. El espacio $\mathbf{so}(3)$ está compuesto por todas las matrices antisimétricas 3x3.

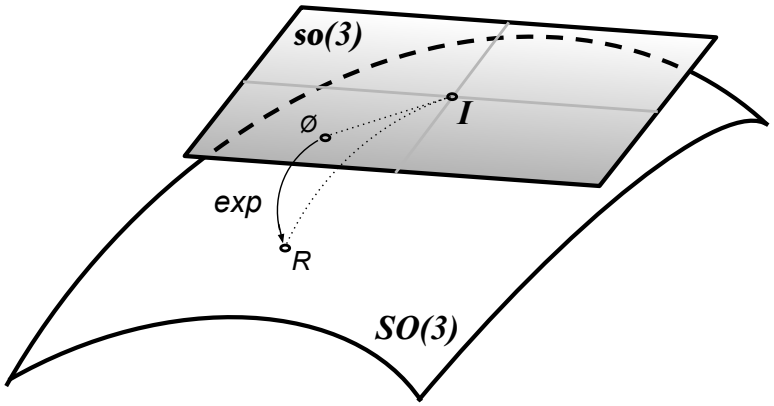


Figura 3.7: Representación del espacio $SO(3)$ y su álgebra de Lie, $so(3)$. La transformación de elementos de $SO(3)$ a $so(3)$ viene dada por la función de mapeo exponencial, *exp*.

El vector de velocidad angular $\boldsymbol{\omega}_\tau \in \mathbb{R}^3$, puede transformarse en una matriz antisimétrica con el operador $(\cdot)^\wedge$:

$$(\cdot)^\wedge : \mathbb{R}^3 \to so(3)$$

$$\boldsymbol{\omega}^\wedge = \begin{pmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{pmatrix}^\wedge = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \tag{15}$$

Y una matriz antisimétrica del grupo $\mathbf{so}(3)$ puede transformarse en una matriz de rotación del grupo $SO(3)$ mediante el mapeo exponencial:

$$\mathtt{E}xp : so(3) \to SO(3)$$

$$exp(\phi^{\wedge}) = \boldsymbol{I} + \frac{sin(\|\phi\|)}{\|\phi\|}\phi^{\wedge} + \frac{1 - cos(\|\phi\|)}{\|\phi\|^2}(\phi^{\wedge})^2 \tag{16}$$

Con las consideraciones anteriores, la rotación $\mathtt{R}_j$ desde la posición $\mathcal{C}_1$ a la $\mathcal{C}_j$ se puede calcular como:

$$\mathtt{R}_j = \mathrm{Exp}\left(\int_{t_1}^{t_j} \boldsymbol{\omega}_\tau d\tau\right) \tag{17}$$

Recordando el modelo del giróscopo, Expresión 8, la velocidad angular $\boldsymbol{\omega}_\tau$ puede reescribirse como:

$$\boldsymbol{\omega}_\tau = \mathbf{b}_g + \boldsymbol{\eta}_g - \tilde{\boldsymbol{\omega}}_\tau \tag{18}$$

Teniendo esto en cuenta, la integral de la Ecuación 17 puede ser expresada por el siguiente producto:

$$\mathtt{R}_j \approx \prod_{n=1}^{j} \mathrm{Exp}\left((\tilde{\boldsymbol{\omega}}_n - \mathbf{b}_g)\,\Delta t_n\right) \tag{19}$$

Por otro lado, la velocidad $\mathbf{v}_j$ de la j^a posición puede ser relacionada con la velocidad de la primera posición, $\mathbf{v}_1$, mediante la integración de la aceleración lineal según:

$$\mathbf{v}_j = \mathbf{v}_1 + \int_{t_1}^{t_j} \mathtt{R}_\tau \mathbf{a}_\tau d\tau \tag{20}$$

Haciendo uso de la Ecuación 9 y asumiendo que la aceleración se mantiene constante durante el intervalo de tiempo Δt_n, la Ecuación 20 puede reescribirse como:

$$\mathbf{v}_j \approx \mathtt{R}_j^{\top}\mathbf{v}_1 + \mathtt{R}_j^{\top}\left[\sum_{n=1}^{j}(\mathtt{R}_n(\tilde{\mathbf{a}}_n - \mathbf{b}_a + \boldsymbol{g}_n))\Delta t_n\right]$$

Al ser $\boldsymbol{g}_n = R_n\boldsymbol{g}_1$, la expresión anterior queda:

$$\mathbf{v}_j \approx \mathtt{R}_j^{\top}\mathbf{v}_1 + \mathtt{R}_j^{\top}\left[\sum_{n=1}^{j}(\mathtt{R}_n(\tilde{\mathbf{a}}_n - \mathbf{b}_a))\Delta t_n\right] + \mathtt{R}_j^{\top} j\boldsymbol{g}_1\Delta t_n$$

$$\mathbf{v}_j \approx \mathtt{R}_j^{\top}(\mathbf{v}_1 + t_{1j}\mathbf{g}_1 + \sum_{n=1}^{j}(\mathtt{R}_n(\tilde{\mathbf{a}}_n - \mathbf{b}_a)\Delta t_n)) \tag{21}$$

Recuérdese de la Sección 3.2 que los *bias* de la aceleración y de la velocidad angular varían ligeramente con el tiempo, [32]. Sin embargo, como sus variaciones son muy pequeñas y la inicialización se realiza típicamente con secuencias de unos pocos segundos, dichos *bias* se consideran constantes en este trabajo.

En las aplicaciones que necesitan de sistemas VI-SLAM, los sensores mantienen una posición relativa fija. Es decir, la IMU y la cámara pueden relacionarse con una transformación $\mathtt{T}_{IC} = (\mathtt{R}_{IC}, \mathbf{p}_{IC})$ ([3]), Sección 3.1. Las medidas inerciales son tomadas por la IMU por lo que es necesario transformar las direcciones del *feature* $\boldsymbol{\mu}_j^i$ y $\boldsymbol{\mu}_k^i$ del sistema de referencia de la cámara al sistema del sensor inercial. Haciendo esto, la Ecuación 10 queda como:

$$-\mathtt{R}_j^\top \mathbf{g}_1 \frac{t_{jk}^2}{2} - \mathbf{v}_j t_{jk} + \mathtt{R}_{IC}\lambda_j^i \boldsymbol{\mu}_j^i - \mathtt{R}_{jk}\mathtt{R}_{IC}\lambda_k^i \boldsymbol{\mu}_k^i = \mathbf{s}_{jk} + \mathtt{R}_{jk}\mathbf{p}_{IC} - \mathbf{p}_{IC} \tag{22}$$

Finalmente, sustituyendo la expresión de la velocidad $\mathbf{v}_j$ de la Ecuación 21 en la Ecuación 22, se obtiene la siguiente expresión que constituye la base de la formulación propuesta:

$$\begin{aligned} -\mathtt{R}_j^\top \mathbf{g}_1 t_{jk}(t_{1j} + \frac{t_{jk}}{2}) - \mathtt{R}_j^\top \mathbf{v}_1 t_{jk} + \mathtt{R}_{IC}\lambda_j^i \boldsymbol{\mu}_j^i - \mathtt{R}_{jk}\mathtt{R}_{IC}\lambda_k^i \boldsymbol{\mu}_k^i = \\ = \mathbf{s}_{jk} + \mathtt{R}_{jk}\mathbf{p}_{IC} - \mathbf{p}_{IC} + \mathtt{R}_j^\top t_{jk} \sum_{n=1}^{j} (\mathtt{R}_n(\mathbf{a}_n - \mathbf{b}_a)\Delta t_n) \end{aligned} \tag{23}$$

La Ecuación 23 presenta 4 incógnitas ($\mathbf{g}_1$, $\mathbf{v}_1$, λ_j^i y λ_k^i) y ha sido obtenida a partir de la correspondencia de un *feature*, $\mathbf{X}_i$, entre un par de *frames* $\mathcal{F}_j$ y $\mathcal{F}_k$ (Figura 3.6). Por lo tanto, se pueden obtener tantas ecuaciones como correspondencias entre pares de *frames* se hayan detectado durante el procesamiento de las imágenes (Sección 3.4). Agrupando todas las posibles ecuaciones se obtiene el siguiente sistema lineal:

$$\mathtt{A}\mathbf{X} = \mathbf{D} \tag{24}$$

Y que puede descomponerse como:

$$\begin{pmatrix} \mathtt{A}_{12} \\ \vdots \\ \mathtt{A}_{jk} \\ \vdots \end{pmatrix} \begin{pmatrix} \mathbf{g}_1 \\ \mathbf{v}_1 \\ \boldsymbol{\Lambda} \end{pmatrix} = \begin{pmatrix} \mathbf{d}_{12} \\ \vdots \\ \mathbf{d}_{jk} \\ \vdots \end{pmatrix} \tag{25}$$

Donde $\mathbf{g}_1$ y $\mathbf{v}_1$ son, respectivamente, la gravedad y la velocidad inicial de la primera cámara, $\mathcal{C}_1$, en su sistema de referencia local; $\mathbf{\Lambda} = \begin{pmatrix} \lambda_1^1 & \dots & \lambda_j^i & \dots & \lambda_k^i & \dots \end{pmatrix}^\top$ es un vector que contiene las distancias desde las posiciones $\{1, \dots, j, \dots, k, \dots, M\}$ de la cámara hasta los *features* $\{1, \dots, i, \dots, N\}$. Finalmente, $\mathbf{d}_{jk}$ corresponde a la parte derecha de la igualdad 23 y tiene la siguiente forma:

$$\mathbf{d}_{jk} = \begin{pmatrix} \mathbf{s}_{jk} + \mathtt{R}_{jk}\mathbf{p}_{IC} - \mathbf{p}_{IC} + \mathtt{R}_j^\top t_{jk} \sum_{n=1}^{j} \left(\mathtt{R}_n(\mathbf{a}_n - \mathbf{b}_a)\Delta t_n\right) \\ \vdots \\ \mathbf{s}_{jk} + \mathtt{R}_{jk}\mathbf{p}_{IC} - \mathbf{p}_{IC} + \mathtt{R}_j^\top t_{jk} \sum_{n=1}^{j} \left(\mathtt{R}_n(\mathbf{a}_n - \mathbf{b}_a)\Delta t_n\right) \end{pmatrix} \tag{26}$$

Téngase en cuenta que las incógnitas $\mathbf{g}_1$, $\mathbf{v}_1$ y $\mathbf{\Lambda}$ constituyen el vector de estado del sistema, $\mathbf{X}$, que contiene información sobre el estado inicial del dispositivo ($\mathbf{g}_1$ y $\mathbf{v}_1$) así como de la escala de la escena ($\mathbf{\Lambda}$). En definitiva, se trata de la información buscada por la inicialización.

Cada submatriz $\mathtt{A}_{jk}$ contiene información tanto de la arquitectura del sensor (posición relativa IMU-cámara) como información visual-inercial relativa al intervalo de tiempo t_{jk} y tiene la siguiente forma:

$$\mathtt{A}_{jk} = \left(\begin{array}{c|c|c|c|c|c|c|c|c} \mathtt{T}_{jk}^g & \mathtt{T}_{jk}^v & \dots & \mathtt{R}_{IC}\mu_j^1 & 0 & \dots & -\mathtt{R}_{jk}\mathtt{R}_{IC}\mu_k^1 & 0 & 0 \\ \mathtt{T}_{jk}^g & \mathtt{T}_{jk}^v & \dots & 0 & \mathtt{R}_{IC}\mu_j^2 & \dots & 0 & -\mathtt{R}_{jk}\mathtt{R}_{IC}\mu_k^2 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & 0 \\ \mathtt{T}_{jk}^g & \mathtt{T}_{jk}^v & \dots & 0 & 0 & \dots & \dots & \dots & \dots \end{array}\right) \tag{27}$$

Donde $\mathtt{T}_{jk}^g = -\mathtt{R}_j^\top t_{jk}(t_{1j} + \frac{t_{jk}}{2})$ y $\mathtt{T}_{jk}^v = -\mathtt{R}_j^\top t_{jk}$. Nótese que en el caso de $j = 1$, es decir, de que no haya adición de nuevos *features*, esta formulación es equivalente a la propuesta por [13].

Para mayor claridad, la Figura 3.8 muestra la distribución de elementos en la matriz $\mathtt{A}$ para un caso particular de los experimentos realizados.

3.6. Cálculo del *bias* del acelerómetro y del giróscopo

Los *bias* del acelerómetro, $\mathbf{b}_a$, y del giróscopo, $\mathbf{b}_g$, son datos de gran relevancia durante la inicialización. Muestra de ello es su presencia explícita en la Ecuación 23. Normalmente se trata de valores desconocidos y que, por tanto, es necesario estimar

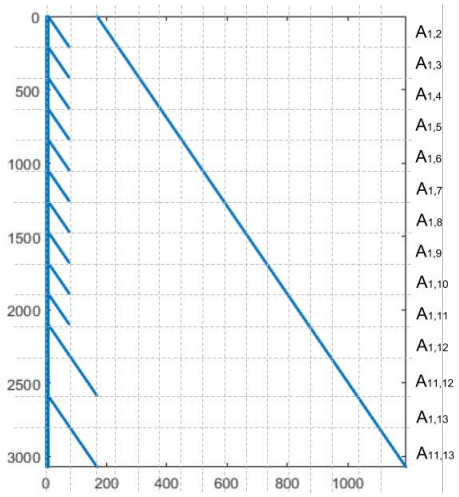


Figura 3.8: Ejemplo ilustrativo de la estructura de la matriz $\mathtt{A}$. El caso concreto del que se ha obtenido la imagen presenta 13 frames y dos extracciones de *featurees* en los *frames* #1 y #11. Los elementos distintos de 0 se presentan en azul. Para cada submatriz $\mathtt{A}_{jk}$ la diagonal izquierda corresponde a $\boldsymbol{\mu}_j^i$ y la derecha a $\boldsymbol{\mu}_k^i$.

durante la inicialización. Para su estimación se ha seguido una aproximación similar a la utilizada por [13].

Tal y como concluye [13], se ha observado que el *bias* del acelerómetro no tiene influencia apreciable sobre los resultados. Para ello se han alterado las medidas de la aceleración lineal con valores aleatorios de magnitud igual a la de *bias* típicos sin observarse variaciones en la solución. Es por esto que puede omitirse su estimación y establecer un *bias* nulo $\mathbf{b}_a = [0, 0, 0]^\top$. La razón de la escasa influencia, tal y como también explica [13], es que la magnitud del *bias* es muy pequeña en comparación con la de otros efectos sobre el acelerómetro como la gravedad o la aceleración debida a las rotaciones. Por ello, su influencia se mantiene inapreciable siempre que el *bias* sea despreciable frente a la suma de la gravedad y la aceleración debida a la rotación.

Para estimar el *bias* del giróscopo se busca el valor que minimiza el residuo de la siguiente función de coste derivada del sistema lineal de ecuaciones recogido en la Ecuación 24:

$$\hat{\mathbf{b}}_g = \arg\min_{\mathbf{b}_g} \|\mathtt{A}\mathbf{X} - \mathbf{D}\|^2 \tag{28}$$

Como valor inicial para llevar a cabo la minimización de la Ecuación 28 se toma $\mathbf{b}_g^0 = [0, 0, 0]^\top$. Se trata de un valor inicial razonable puesto que el valor real del *bias*

es, casi siempre, muy cercano a 0 en los tres ejes y, por lo tanto, se reduce el riesgo de que el proceso quede detenido en un mínimo local. Muestra de ello es la convergencia ilustrada en la Figura 3.9 y que pertenece a uno de los experimentos realizados. De manera similar a lo mencionado por [13], se observa que el *bias* del giróscopo converge en unos $2-3$ segundos bajo condiciones similares.

Tal y como ilustra la Figura 3.4, el *bias* se recalcula cada vez que se añade un nuevo *frame* a la inicialización. El valor inicial en en cada iteración es el resultado de la iteración anterior. Es decir, para cada nueva minimización $i > 1$, $\mathbf{b}_{gi}^{0} = \hat{\mathbf{b}}_{gi-1}$.

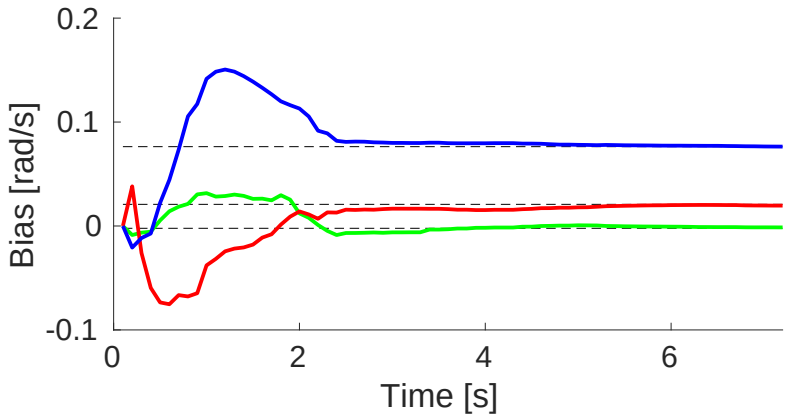


Figura 3.9: Estimación del *bias* del giróscopo mediante la minimización mostrada en 28. El *bias* real según el *ground truth* es $\mathbf{b}_g = [-0{,}00222, 0{,}02082, 0{,}07632]$ rad/s y está representado por las líneas discontinuas.

Es importante tener en cuenta que el *bias* puede divergir a medida que el tiempo de integración aumenta. Para evitar esto, [13] propone el uso de un valor del *bias* conocido previamente. Como en este trabajo se asume que no existe conocimiento del *bias* en ningún momento, se ha limitado el uso de la formulación propuesta a secuencias de, como máximo, 5,5 segundos (110 *frames*) con el fin de evitar esta divergencia. Tal y como muestran los experimentos, los valores estimados del *bias* del giróscopo son lo suficientemente precisos en secuencias de dichas duraciones.

3.7. Análisis de complejidad y coste de la inicialización

Cada submatriz $\mathtt{A}_{jk}$ (Ecuación 26), correspondiente a un par de posiciones de la cámara $\mathcal{C}_j$ y $\mathcal{C}_k$, contiene $3N_{jk}$ ecuaciones y $3+3+2N_{jk}$ incógnitas (3 correspondientes al vector de la gravedad, $\mathbf{g}_1$, 3 al de la velocidad, $\mathbf{v}_1$, y $2N_{jk}$ a las dos distancias λ_j^i y λ_k^i relativas a cada correspondencia entre las cámaras $\mathcal{C}_j$ y $\mathcal{C}_k$). Para todo el sistema, definido en la Ecuación 24, se pueden sumar las ecuaciones relativas a todos los pares de posiciones de la cámara, $\mathtt{P}$, obteniendo así $3\sum_P N_{jk}$ ecuaciones y $3+3+2\sum_P N_{jk}$ incógnitas.

Tal y como muestra la Figura 3.8, la matriz $\mathtt{A}$ es dispersa. Cada fila tiene, como máximo, 8 elementos distintos de 0 para cualquier submatriz $\mathtt{A}_{jk}$. El menor ratio de dispersión (número de elementos iguales a 0 sobre el numero total de elementos) es $1-\sum_P \frac{8}{N_{jk}}$, el cual es muy próximo a 1 ya que, normalmente, se dispone de un gran número de correspondencias entre imágenes, N_{jk}. Usando los paquetes de álgebra lineal estándar como, por ejemplo, *Eigen*, [9], el sistema puede ser resuelto de manera muy eficiente, Tabla 3.1.

Frames	Features	Incógnitas	Ec's	Preint.	Construcción sistema	Resolución sistema	Estimación bias	Tiempo total
5	94	758	1128	0.46	122	0.98	731	866
10	94	1698	2538	0.50	234	1.42	1402	1650
15	94	2638	3948	0.55	499	2.16	2986	3499
20	94	3578	5358	0.65	875	2.90	4362	5254
25	93	4470	6696	0.69	1282	3.69	6433	7732
30	92	5342	8004	0.70	1851	4.46	9285	11154
35	126	8574	12852	0.92	1793	4.37	10859	12673
40	119	9288	13923	0.99	1992	4.80	11005	13016
45	119	10478	15708	1.08	2107	5.02	11681	13803
50	110	10786	16170	1.12	2315	5.89	12811	15149
55	102	11022	16524	1.21	2612	6.52	14459	17094

Tabla 3.1: Tiempos registrados para los principales procesos del algoritmo en el caso concreto de un experimento representativo. Implementación en C++. Unidades de tiempo en [ms]. Especificaciones hardware: procesador Intel Core i7-8750H CPU 2,2GHz x 12, RAM 16GB

El algoritmo ha sido implementado tanto en MatLAB como en C++ para medir su eficiencia. En la Tabla 3.1 se muestra un ejemplo de los tiempos que se han registrado para uno de los experimentos más representativos con la implementación

en C++. Aunque los tiempos varían según las características de cada secuencia, no se han observado grandes desviaciones con respecto a los mostrados en la Tabla 3.1. Nótese que el tiempo depende directamente del tamaño del sistema a resolver y que los procesos que más tiempo requieren son la construcción del sistema de ecuaciones y la estimación del *bias*.

Para evaluar los tiempos, es importante tener en cuenta el gran tamaño del sistema a tratar. Obsérvese que el cálculo del *bias* implica la minimización del residuo de una función de coste que incluye a todo el sistema (Ecuación 28). Esto hace que la etapa de estimación del *bias* sea la más costosa computacionalmente.

Además, si bien tiempos superiores a ~ 5 segundos pueden parecer elevados para la inicialización, téngase en cuenta que no existen, hasta la fecha, trabajos con los que comparar estos tiempos ya que aquellos que tratan la inicialización de sistemas VI-SLAM, no se ocupan de la estimación pura del *bias* tal y como se ha referido en la Sección 2.2.

Para resolver la minimización de la Ecuación 28 se ha utilizado el algoritmo de Levenberg-Marquardt por ser el que mejores resultados muestra en cuanto a precisión y eficiencia.

Según las consideraciones mencionadas, se puede concluir que los tiempos son coherentes con la formulación desarrollada. La búsqueda de una metodología que optimice la estimación del *bias* puede suponer una de las líneas futuras de este trabajo.

Capítulo 4

Experimentos

4.1. Condiciones experimentales

Para evaluar la formulación propuesta se ha utilizado la base de datos pública *EuRoC MAV* [3]. Esta base de datos contiene 11 secuencias grabadas en dos escenas diferentes y con dos cámaras incorporadas en un dron. Las escenas son, por un lado, una habitación con diferentes objetos que favorecen la extracción de *features* y correspondencias (*"Vicon Room"*) y, por otro lado, un entorno industrial de tamaño mucho mayor (*"Machine Hall"*). Las secuencias grabadas en cada escena son clasificadas como fáciles, medias o difíciles en función de las condiciones de luminosidad y agresividad del movimiento.

Todas las secuencias contienen, además de los propios *frames*, las medidas inerciales de la IMU (velocidad angular y aceleración lineal) así como un *ground truth* de gran precisión de la posición, orientación y *bias* de giróscopo y acelerómetro.

	MH_01_easy	MH_02_easy	MH_03_medium
Inicial	900	800	400
Final	3600	2900	2500
	MH_04_difficult	MH_05_difficult	V1_01_easy
Inicial	500	500	100
Final	1900	2100	2700
	V1_02_medium	V1_03_difficult	V2_01_easy
Inicial	100	200	100
Final	1600	2000	2100
	V2_02_medium	V2_03_difficult	
Inicial	100	100	
Final	2200	1800	

Tabla 4.1: *Frames* inicial y final usados para la evaluación de cada secuencia de *EuRoC*. Las nomenclaturas "V" y "MH" se refieren a la habitación y al entorno industrial, respectivamente.

Para la evaluación de las secuencias se han utilizado los *frames* de la cámara izquierda y los datos inerciales registrados por la IMU. Además, se han tenido en cuenta únicamente los fragmentos de las secuencias en los que el dron está en movimiento ya

que los primeros y últimos *frames* corresponden al despegue y aterrizaje, respectivamente. La inicialización propuesta ha sido evaluada cada 20 *frames* en los intervalos indicados en la Tabla 4.1 obteniéndose así un conjunto de 1080 casos de gran variedad en cuanto a movimientos y condiciones de iluminación.

4.2. Resultados experimentales

En primer lugar, conviene mostrar de forma cualitativa una comparación de los aspectos más relevantes de la formulación propuesta y de la utilizada como referencia, [13]. Para mostrar el funcionamiento general de las inicializaciones se incluye la Figura 4.1 que muestra el error cometido en la estimación de la trayectoria de la cámara en función del número de *frames* utilizados para la inicialización. Aunque la figura muestra un ejemplo concreto de los experimentos realizados, por norma general se pueden observar 3 regiones:

- Región 1. Corresponde a una fase en la que tanto el número de *frames* disponibles como el desplazamiento del dispositivo son muy pequeños. El elevado error se debe, por lo tanto, a la escasa traslación de la cámara y, en consecuencia, al reducido paralaje.

- Región 2. En la segunda región el error comienza a reducirse conforme se añaden nuevos *frames* y la deriva de la IMU todavía no es demasiado elevada. El error en esta región es menor cuanto mayor es el número de correspondencias y mejor repartidas están por la imagen. Téngase en cuenta que la formulación propuesta permite disponer de un elevado número de correspondencias durante un mayor número de *frames*.

- Región 3. En la tercera región el error crece debido, principalmente, a dos factores. En primer lugar, a la desaparición de un gran número de correspondencias y, en segundo lugar, a la inevitable deriva de la IMU. Nótese que la formulación propuesta elimina el problema de la pérdida de correspondencias.

La Figura 4.1 muestra también cómo el algoritmo propuesto (curvas azul y verde) mejora los resultados del algoritmo de referencia [13] (curva roja) gracias a los efectos expuestos en la descripción de las regiones. Las dos novedades que pueden ser observadas en la figura son la reducción general del error cometido y la ampliación de la zona de bajo error correspondiente al "valle" de la curva (región 2).

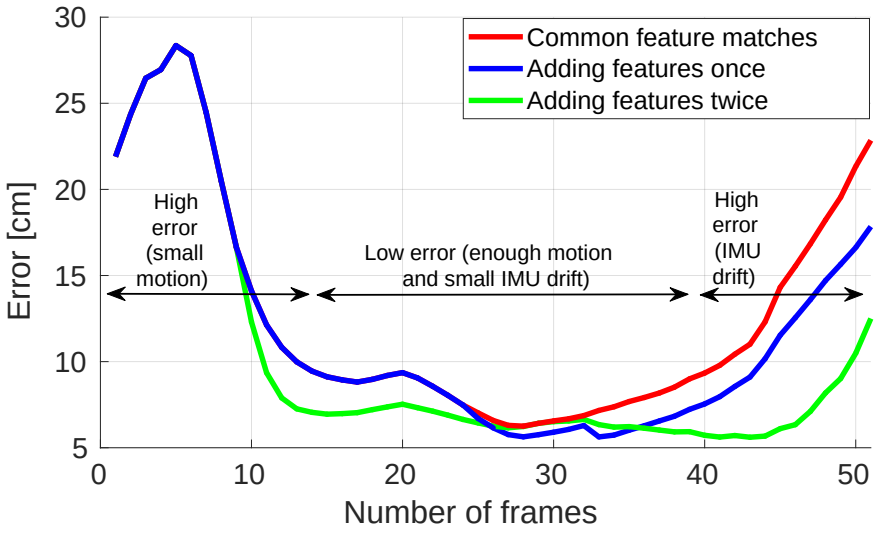


Figura 4.1: Ejemplo del error cometido por la inicialización en función del número de *frames* utilizados. Nótese la mejora obtenida sobre el estado del arte mediante la adición de nuevos *features*.

La región de menor error depende tanto del movimiento del sensor como de la profundidad de la escena por lo que es muy difícil predecir el número de *frames* ideal para una correcta inicialización. La formulación propuesta amplía notablemente el tamaño de ésta región de bajo error, lo que aumenta considerablemente las probabilidades de realizar una correcta inicialización.

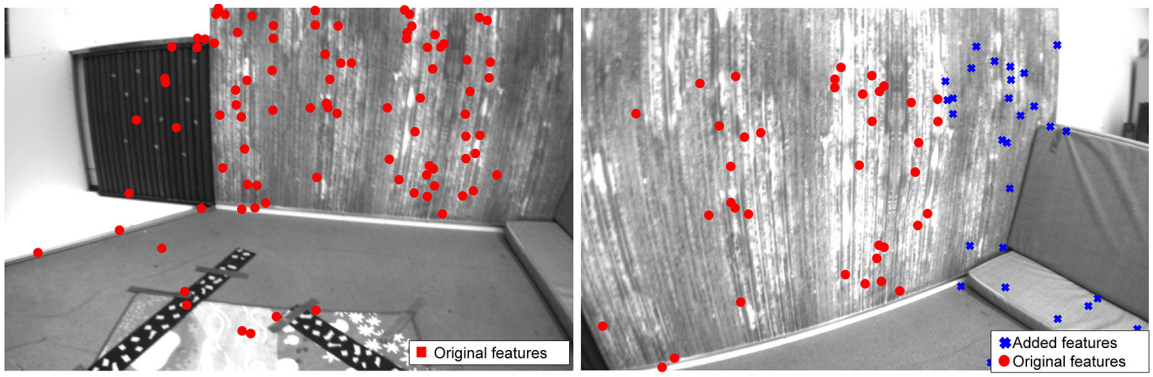
Aunque lo más relevante de la Figura 4.1 es su aspecto cualitativo, el error que representa se calcula como:

$$\mathrm{E}rror = \sum_{i=2}^{M} (\hat{\mathbf{p}}_i - \mathbf{p}_i) \tag{29}$$

Donde $\mathbf{p_i}$ y $\hat{\mathbf{p}}_\mathbf{i}$ son las posiciones estimada y real de la cámara, respectivamente, en un determinado instante, i. M es el número de *frames* utilizados.

Hasta ahora se han contemplado los casos en los que la inicialización, con mayor o menor error, llega a completarse. Sin embargo, tal y como ilustra la Figura 4.2, esto no siempre se consigue. En estos casos, ninguno de los *features* extraídos en el primer *frame* (en rojo en la Figura 4.2a) puede ser seguido hasta el último (Figura 4.2d) debido a movimientos bruscos o con grandes ángulos, malas condiciones de iluminación o secuencias demasiado largas.

La formulación propuesta evita este modo de fallo mediante la adición y seguimiento de nuevos *features*. En el ejemplo mostrado en la Figura 4.2 se añaden nuevos *features* en dos ocasiones (en azul en las Figuras 4.2b y 4.2c) evitando así la pérdida de *features* iniciales y, por lo tanto, el fallo de la inicialización. De esta manera la robustez es mejorada gracias a la utilización de un mayor número de *frames* y, además, se consigue una mayor precisión gracias a un mejor reparto de *features* en todas las imágenes.



(a) Primer *frame* (b) *Frame* #35

(c) *Frame* #78 (d) *Frame* #104

Figura 4.2: Ejemplo de la adición de *features*. (a) muestra, en rojo, los *features* extraídos en el primer *frame*. En (b), tras 35 *frames*, aparecen nuevas regiones de la escena en la parte derecha de la imagen y se produce una primera adición de nuevos *features* (en azul). En (c) se realiza una segunda adición de *features* en la parte derecha de la imagen. Nótese que tras 104 *frames*, no queda ningún *feature* común a toda la secuencia.

Los resultados cuantitativos de la evaluación de los 1080 casos se muestran en la Tabla 4.2. Las dos métricas utilizadas para la comparación de la formulación propuesta con la inicialización de referencia, [13], son el número de inicializaciones fallidas frente al número de intentos y el error cometido en la estimación de la trayectoria de la cámara. Por un lado, un intento de inicialización se considera fallido si se pierden todas las correspondencias (*matches*) en algún momento del mismo. Por otro lado, el error en la estimación de la trayectoria se evalúa mediante la raíz cuadrada del error cuadrático medio normalizado, NRMSE (del inglés "*Normalized Root Mean Square Error*"):

$$NRMSE = \frac{RMSE}{\sum_{i=1}^{M-1} |\mathbf{p}_{i+1} - \mathbf{p}_i|} \tag{30}$$

Donde:

- $RMSE$ es la raíz cuadrada del error cuadrático medio ("*Root Mean Square Error*" en inglés) y se define como $RMSE = \sqrt{\frac{\sum_{\mathtt{i}=2}^{\mathtt{M}}(\hat{\mathbf{p}}_\mathtt{i} - \mathbf{p}_\mathtt{i})}{\mathtt{M}-1}}$. $\mathbf{p_i}$ y $\hat{\mathbf{p}}_\mathtt{i}$ denotan las posiciones estimada y real de la cámara en un determinado instante, i, respectivamente. El número de *frames*, M, coincide con el número de posiciones medidas.
- El denominador $\sum_{\mathtt{i}=1}^{\mathtt{M}-1} |\mathbf{p}_{\mathtt{i}+1} - \mathbf{p}_\mathtt{i}|$ es la longitud de la trayectoria recorrida por la cámara. Se trata del término utilizado para la normalización del $RMSE$.

Se utiliza la normalización del $RMSE$ para poder comparar secuencias de diferentes longitudes. Esta normalización facilita la comparación de conjuntos de datos de diferentes escalas como lo es este caso. Además, para clarificar los resultados, la Tabla 4.2 establece diferentes categorías en función de la dificultad de la secuencia evaluada así como en función de la escena ("*Vicon Room*" o "*Machine Hall*").

De la Tabla 4.2 se pueden obtener varias conclusiones atendiendo a las dos metricas utilizadas. Obsérvese, en primer lugar, la gran reducción de fallos que supone la formulación propuesta frente al estado del arte representado por [13]. La inicialización propuesta por [13] presenta un reducido ratio de fallos cuando se evalúa en secuencias cortas donde es sencillo mantener (correspondencias) comunes a lo largo de todas ellas. No obstante, el ratio crece rápidamente con el número de *frames* y dificultad de las secuencias. Por el contrario, el ratio de fallos de la formulación propuesta se mantiene cercano a cero en todos los casos, lo que supone una importante mejora en la robustez de la inicialización.

			Vicon Room				Machine Hall			
	Longitud		Fallos (Evaluaciones totales)		NRMSE (%) (Evaluaciones con éxito)		Fallos (Evaluaciones totales)		NRMSE (%) (Evaluaciones con éxito)	
	Frames	Seg.	[13]	Propuesta	[13]	Propuesta	[13]	Propuesta	[13]	Propuesta
Secuencias fáciles	10	0.5	0 (0 %)	0 (0 %)	118.8	118.8	0 (0 %)	0 (0 %)	120.8	120.8
	30	1.5	5 (2.2 %)	**0 (0 %)**	36.7	**35.3**	0 (0 %)	0 (0 %)	48.0	48.0
	50	2.5	9 (3.9 %)	**0 (0 %)**	21.0	**17.8**	1 (0.4 %)	**0 (0 %)**	30.1	**30.0**
	60	3	14 (6.0 %)	**0 (0 %)**	17.5	**16.4**	1 (0.4 %)	**0 (0 %)**	20.6	**20.0**
	70	3.5	24 (10.3 %)	**0 (0 %)**	15.0	**12.9**	1 (0.4 %)	**0 (0 %)**	22.1	**21.9**
	90	4.5	40 (17.2 %)	**0 (0 %)**	11.5	**9.4**	6 (2.5 %)	**0 (0 %)**	20.1	**18.7**
	110	5.5	82 (35.3 %)	**0 (0 %)**	10.2	**6.8**	14 (5.9 %)	**0 (0 %)**	18.4	**18.1**
Secuencias medias	10	0.5	9 (5.1 %)	**0 (0 %)**	100.7	**99.6**	7 (6.6 %)	**0 (0 %)**	131.2	131.2
	30	1.5	43 (24.3 %)	**0 (0 %)**	30.1	**28.2**	8 (7.5 %)	**0 (0 %)**	51.2	**51.1**
	50	2.5	87 (47.8 %)	**0 (0 %)**	17.6	**16.8**	27 (25.5 %)	**0 (0 %)**	28.4	**28.1**
	60	3	93 (51.1 %)	**0 (0 %)**	14.6	**11.0**	36 (34.0 %)	**0 (0 %)**	25.2	**25.0**
	70	3.5	108 (59.4 %)	**0 (0 %)**	13.7	**11.5**	37 (34.9 %)	**0 (0 %)**	22.7	**22.4**
	90	4.5	141 (77.5 %)	**0 (0 %)**	8.0	**7.1**	40 (37.7 %)	**0 (0 %)**	12.9	**12.6**
	110	5.5	157 (88.7 %)	**1 (0.6 %)**	5.2	**3.8**	54 (50.1 %)	**1 (0.9 %)**	8.7	**7.5**
Secuencias difíciles	10	0.5	79 (43.4 %)	**0 (0 %)**	98.8	**98.6**	0 (0 %)	0 (0 %)	142.3	142.3
	30	1.5	132 (72.5 %)	**0 (0 %)**	32.7	**32.6**	2 (1.6 %)	**0 (0 %)**	71.0	**70.7**
	50	2.5	150 (82.4 %)	**0 (0 %)**	17.3	**16.7**	5 (3.3 %)	**0 (0 %)**	52.4	**52.3**
	60	3	162 (89.0 %)	**1 (0.5 %)**	15.0	**12.9**	19 (12.5 %)	**0 (0 %)**	47.2	**45.7**
	70	3.5	165 (90.7 %)	**1 (0.5 %)**	10.5	**10.4**	22 (14.5 %)	**0 (0 %)**	41.3	**40.6**
	90	4.5	172 (94.5 %)	**2 (1.1 %)**	5.9	**5.7**	51 (33.6 %)	**0 (0 %)**	35.1	**33.1**
	110	5.5	177 (97.3 %)	**3 (1.6 %)**	3.7	**3.5**	71 (46.7 %)	**0 (0 %)**	28.5	**27.9**

Tabla 4.2: Número de fallos y NRMSE(%) tanto de la inicialización propuesta como de la desarrollada en [13]. Se utilizan dos escenarios extraidos del dataset EuRoC (*Vicon Room* y *Machine Hall*) así como tres niveles de dificultad de las secuencias. El NRMSE(%) solo se evalua en las inicializaciones completadas por ambas formulaciones.

En segundo lugar, se observa que el $NRMSE$ es muy elevado en secuencias cortas, tanto en el caso de [13] como en el propuesto. Las secuencias de 10 *frames* (de cualquier dificultad y escena) son ejemplos donde se obtiene un $NRMSE$ cercano o mayor del 100 %. En secuencias muy cortas, un $NRMSE$ tan elevado indica que el error es similar la longitud de las trayectorias, es decir, de tan solo unos pocos centímetros. Los errores comienzan a decrecer y ser aceptables al cabo de $2-3$ segundos donde, precisamente, la formulación propuesta muestra una notable mejora con respecto a [13], especialmente en el ratio de fallos.

La dependencia del $NRMSE$ con el numero de frames se aprecia de manera clara en la Figura 4.3. En dicha figura se muestra un histograma del $NRMSE$ acumulado en secuencias de diferente duración. Se aprecia una gran reducción del error entre las curvas de 10 y de 30 *frames* debido al escaso paralaje en las secuencias más cortas. Este efecto es coherente con el observado en la transición entre las regiones 1 y 2

de la Figura 4.1. Si se sigue aumentando el número de *frames*, el $NRMSE$ sigue reduciéndose, aunque en menor medida, hasta alcanzar un mínimo para secuencias de unos 110 *frames*. Si se continúa aumentando la duración de las secuencias, el error comienza a crecer debido a la deriva de la IMU (región 3 de la Figura 4.1).

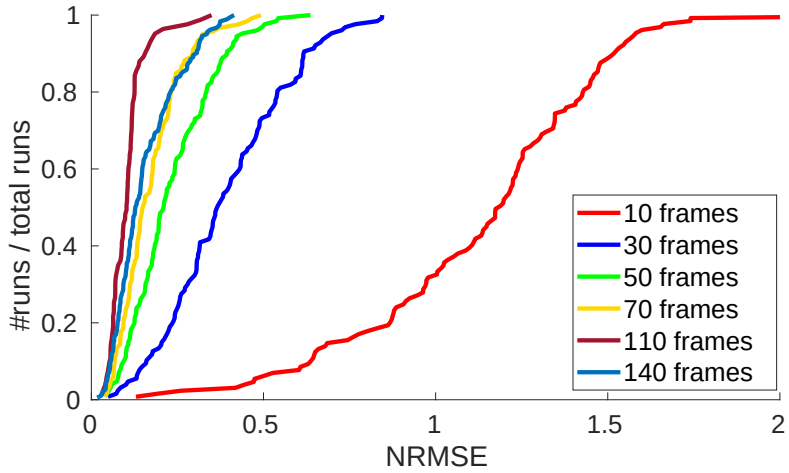
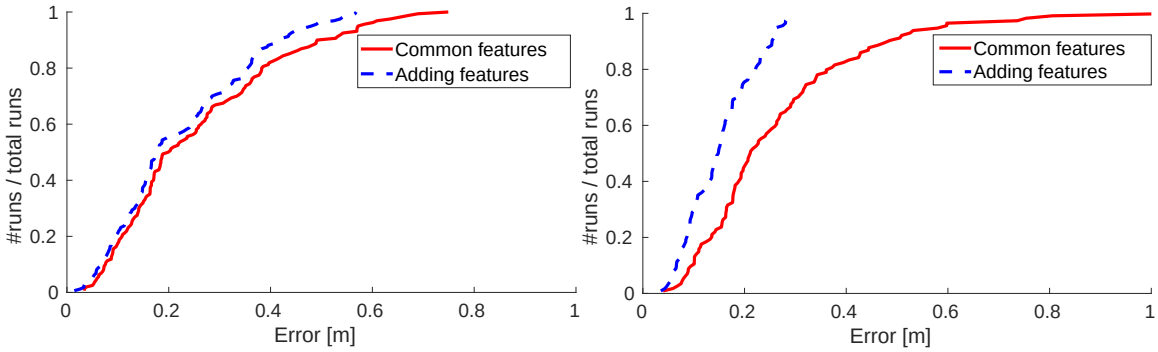


Figura 4.3: Histograma del *NRMSE* acumulado para todas las secuencias de diferentes longitudes. Se observa un mínimo para secuencias de 110 *frames*. Nótese que la relación "error-número de *frames*" muestra el mismo comportamiento que en la Figura 4.1.

Tal y como refleja la Figura 4.3, el menor error se obtiene en secuencias de entre 70 y 110 *frames*, lo que supone una inicialización de entre 3,5 y 5,5 seg. No obstante, en aplicaciones donde el tiempo es crítico, puede optarse por inicializar tan pronto como el error sea aceptable, es decir, pasados $1{,}5 - 2$ seg ($30 - 50$ *frames*). En cualquiera de los dos casos, la formulación propuesta mejora los resultados de [13] tal y como muestran la Tabla 4.2 y la Figura 4.4.

Por último, la Figura 4.5 muestra un ejemplo de la trayectoria estimada en un caso concreto. Además, la semilla calculada en la inicialización puede mejorarse mediante algoritmos de optimización no lineal como el propuesto en [6] obteniéndose así una estimación incluso más cercana al *ground truth*.



(a) 50 *frames* (b) 100 *frames*

Figura 4.4: Histogramas del $NRMSE$ acumulado para las secuencias de longitud 50 *frames*, (a), y 100 *frames*, (b). La reducción del error que se consigue es mayor en secuencias largas puesto que es en estas donde más correspondencias se pierden, lo que se consigue solucionar añadiendo *features*.

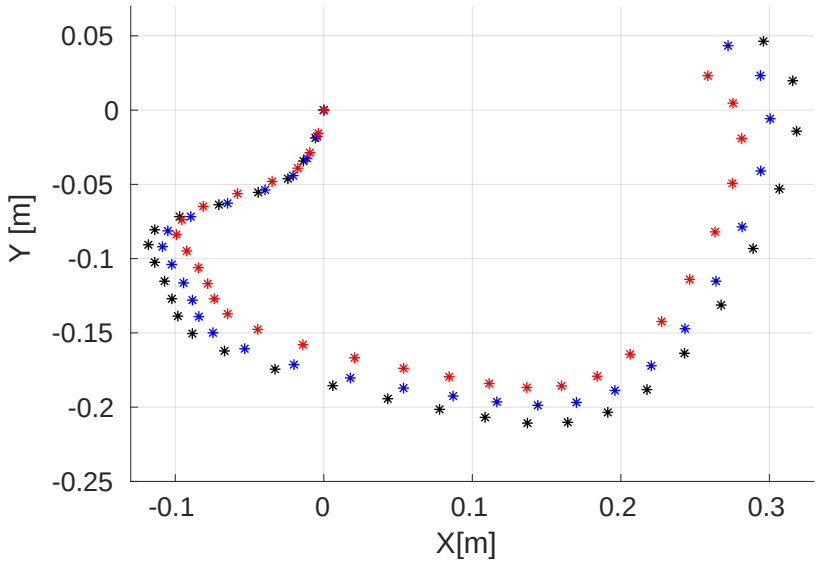


Figura 4.5: Ejemplo concreto de la trayectoria calculada por la inicialización. En negro se muestra el *ground truth*, en rojo la trayectoria estimada por la inicialización y en azul la trayectoria tras una optimización no lineal, [6]. Se trata de una trayectoria 3D y en la imagen se muestra una vista superior.

Capítulo 5

Conclusiones y trabajo futuro

5.1. Conclusiones

En este trabajo se ha propuesto un algoritmo para la inicialización de sistemas VI-SLAM que mejora notablemente la robustez y precisión conseguidos por el estado del arte actual. Como se ha mencionado a lo largo de estas páginas, la principal novedad de la formulación propuesta es que no son necesarias correspondencias comunes entre todos los *frames*.

Mediante las imágenes tomadas por la cámara monocular y las medidas inerciales registradas por la IMU, la formulación propuesta permite calcular un vector de estado del sistema que sirve como semilla inicial para el posterior proceso de optimización no lineal en el que se basan los sistemas VI-SLAM. Dicho vector de estado se compone de la velocidad inicial de la cámara, el vector de la gravedad, el *bias* del giróscopo y la localización de una serie de puntos que permite calcular la escala de la escena.

También se han realizado una gran cantidad de experimentos utilizando las secuencias del *dataset EuRoC* para medir en qué grado se mejora el estado del arte. Los resultados indican una mejora en el ratio de fallos, que se reduce a cero en las secuencias más sencillas y que pasa de más de un 50 % a menos de un 5 % en las secuencias de dificultad media y alta. Además, también se mejora la precisión general de la solución.

Por último, la implementación de la formulación propuesta tanto en MatLAB como en C++ ha permitido realizar los análisis y comparaciones con el máximo rigor y precisión.

5.2. Trabajo futuro

Este trabajo supone un paso más hacia la implantación de sistemas VI-SLAM en aplicaciones reales cada vez más diversas (VR, AR, UAV, etc.). A partir de los últimos avances en sistemas VI-SLAM y, en particular, de la formulación propuesta, existen varias direcciones en las que se puede seguir avanzando.

Por un lado, lo expuesto en estas páginas es una inicialización *offline* mientras

que gran parte de las aplicaciones reales precisan de procesos *online*, es decir, capaces de procesar la informacion y generar resultados en tiempo real. Para muchas aplicaciones, es necesario inicializar correctamente lo antes posible para dar paso al resto de procesos del sistema VI-SLAM. Para conseguirlo, hay que superar las dificultades que presentan los procesos más costosos computacionalmente, especialmente la estimación del *bias*.

Si bien este trabajo incluye el cálculo del *bias* sin ningún tipo de conocimiento ni estimación previa, todavía es necesaria una optimización del proceso. Para ello pueden estudiarse, por ejemplo, más alternativas al algoritmo de minimización de Levenberg-Marquardt. Otros procesos costosos como la construcción del sistema de ecuaciones, 24, también pueden ser estudiados y modificados.

Por otro lado, los resultados aquí mostrados se han realizado a partir de un *dataset* "ideal"donde la IMU es de gran calidad y está perfectamente sincronizada con la cámara. En una situación real, las medidas inerciales son considerablemente peores y no están sincronizadas con la cámara. Antes de implementar esta formulación en un sistema VI-SLAM, es necesario realizar experimentos y verificar su funcionamiento sobre dispositivos de menor precisión como, por ejemplo, teléfonos móviles o gafas de AR y VR. La sincronización entre cámara e IMU es otro de los problemas típicos para el cual es necesario proponer un método de sincronización en el caso de una aplicación real.

Por último, se puede estudiar la integración de esta inicialización en un sistema VI-SLAM completo de tal forma que se pueda conocer con exactitud los errores máximos admisibles en la inicialización así como otros parámetros de interés.

Bibliografía

[1] Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M. Seitz, and Rick Szeliski. Building rome in a day. *Communications of the ACM*, 54:105–112, 2011.

[2] Clemens Arth, Christian Pirchheim, Jonathan Ventura, Dieter Schmalstieg, and Vincent Lepetit. Instant outdoor localization and SLAM initialization from 2.5D maps. *IEEE transactions on visualization and computer graphics*, 21(11):1309–1318, 2015.

[3] Michael Burri, Janosch Nikolic, Pascal Gohl, Thomas Schneider, Joern Rehder, Sammy Omari, Markus W Achtelik, and Roland Siegwart. The EuRoC micro aerial vehicle datasets. *The International Journal of Robotics Research*, 35(10):1157–1163, 2016.

[4] J. Civera, A.J. Davison, and JMM Montiel. Interacting multiple model monocular SLAM. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3704–3709, 2008.

[5] Piet C. de Groen. Using artificial intelligence to improve adequacy of inspection in gastrointestinal endoscopy. *Techniques in Gastrointestinal Endoscopy*, page 150640, 2019.

[6] Javier Domínguez-Conti, Jianfeng Yin, Yacine Alami, and Javier Civera. Visual–inertial slam initialization: A general linear formulation and a gravity–observing non-linear optimization. In *IEEE International Symposium for Mixed and Augmented Reality*, 2018.

[7] Christian Forster, Luca Carlone, Frank Dellaert, and Davide Scaramuzza. On-manifold preintegration for real-time visual–inertial odometry. *IEEE Transactions on Robotics*, 33(1):1–21, 2017.

[8] Steffen Gauglitz, Chris Sweeney, Jonathan Ventura, Matthew Turk, and Tobias Hollerer. Live tracking and mapping from both general and rotation-only camera motion. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 13–22. IEEE, 2012.

[9] Gaël Guennebaud, Benoît Jacob, et al. Eigen v3. http://eigen.tuxfamily.org, 2010.

[10] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, USA, 2 edition, 2003.

[11] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask r-cnn. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[12] Vadim Indelman, Stephen Williams, Michael Kaess, and Frank Dellaert. Information fusion in navigation systems via factor graph based incremental smoothing. *Robotics and Autonomous Systems*, 61(8):721–738, 2013.

[13] Jacques Kaiser, Agostino Martinelli, Flavio Fontana, and Davide Scaramuzza. Simultaneous state initialization and gyroscope bias calibration in visual inertial aided navigation. *IEEE Robotics and Automation Letters*, 2(1):18–25, 2017.

[14] G. Klein and D. Murray. Parallel tracking and mapping for small AR workspaces. In *Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality*, 2007.

[15] Stefan Leutenegger, Simon Lynen, Michael Bosse, Roland Siegwart, and Paul Furgale. Keyframe-based visual–inertial odometry using nonlinear optimization. *The International Journal of Robotics Research*, 34(3):314–334, 2015.

[16] Mingyang Li and Anastasios I Mourikis. High-precision, consistent EKF-based visual-inertial odometry. *The International Journal of Robotics Research*, 32(6):690–711, 2013.

[17] Haomin Liu, Guofeng Zhang, and Hujun Bao. Robust keyframe-based monocular SLAM for augmented reality. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 1–10, 2016.

[18] Agostino Martinelli. Vision and imu data fusion: Closed-form solutions for attitude, speed, absolute scale, and bias determination. *IEEE Transactions on Robotics*, 28(1):44–60, 2012.

[19] Agostino Martinelli. Closed-form solution of visual-inertial structure from motion. *International journal of computer vision*, 106(2):138–152, 2014.

[20] Raúl Mur-Artal and Juan D Tardós. ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. *IEEE Transactions on Robotics*, 2017.

[21] Raúl Mur-Artal and Juan D Tardós. Visual-inertial monocular SLAM with map reuse. *IEEE Robotics and Automation Letters*, 2(2):796–803, 2017.

[22] Alonso Patron-Perez, Steven Lovegrove, and Gabe Sibley. A spline-based trajectory representation for sensor fusion and rolling shutter cameras. *International Journal of Computer Vision*, 113(3):208–219, 2015.

[23] Pedro Piniés, Todd Lupton, Salah Sukkarieh, and Juan D Tardós. Inertial aiding of inverse depth slam using a monocular camera. In *2007 IEEE International Conference on Robotics and Automation*, pages 2797–2802. IEEE, 2007.

[24] Tong Qin, Peiliang Li, and Shaojie Shen. Vins-mono: A robust and versatile monocular visual-inertial state estimator. *arXiv preprint arXiv:1708.03852*, 2017.

[25] Tong Qin and Shaojie Shen. Robust initialization of monocular visual-inertial estimation on aerial robots. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4225–4232, 2017.

[26] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 91–99. Curran Associates, Inc., 2015.

[27] Thomas Schoeps, Torsten Sattler, Christian Häne, and Marc Pollefeys. 3d modeling on the go: Interactive 3d reconstruction of large-scale scenes on mobile devices. 10 2015.

[28] Jianbo Shi et al. Good features to track. In *IEEE Computer Vision and Pattern Recognition*, pages 593–600, 1994.

[29] Chengzhou Tang, Oliver Wang, and Ping Tan. GlobalSLAM: Initialization-robust Monocular Visual SLAM. *arXiv preprint arXiv:1708.04814*, 2017.

[30] Carlo Tomasi and Takeo Kanade. Detection and tracking of point features. Technical report, CMU, 1991.

[31] Prune Truong, Stefanos Apostolopoulos, Agata Mosinska, Samuel Stucky, Carlos Ciller, and Sandro De Zanet. Glampoints: Greedily learned accurate match points. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[32] Oliver J. Woodman. An introduction to inertial navigation. Technical Report UCAM-CL-TR-696, University of Cambridge, Computer Laboratory, aug 2007.

[33] Zhenfei Yang and Shaojie Shen. Monocular visual–inertial state estimation with online initialization and camera–imu extrinsic calibration. *IEEE Transactions on Automation Science and Engineering*, 14(1):39–51, 2017.