Universidad Zaragoza

Trabajo Fin de Máster

Localización, mapeo y seguimiento de objetos en escenas dinámicas

Localization, mapping and object tracking in dynamic scenes

Autor

Javier Tirado Garín

Director

Javier Civera Sancho

ESCUELA DE INGENIERÍA Y ARQUITECTURA
2021

TRABAJOS DE FIN DE GRADO / FIN DE MÁSTER

Escuela de Ingeniería y Arquitectura Universidad Zaragoza

DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD

(Este documento debe remitirse a seceina@unizar.es dentro del plazo de depósito)

D./Dª. Javier Tirado Garín ,

en aplicación de lo dispuesto en el art. 14 (Derechos de autor) del Acuerdo de 11 de septiembre de 2014, del Consejo de Gobierno, por el que se aprueba el Reglamento de los TFG y TFM de la Universidad de Zaragoza,

Declaro que el presente Trabajo de Fin de Estudios de la titulación de Máster Universitario en Ingeniería Industrial (Título del Trabajo)

Localización, Mapeo y seguimiento de objetos en escenas dinámicas

es de mi autoría y es original, no habiéndose utilizado fuente sin ser citada debidamente.

Zaragoza, 24 de Noviembre de 2021

TIRADO GARIN JAVIER - 77133923L

Firmado digitalmente por TIRADO GARIN JAVIER - 77133923L
Fecha: 2021.11.24 16:21:13 +01'00'

Fdo: Javier Tirado Garín

Agradecimientos

En primer lugar, me gustaría agradecer a mi tutor, muchas gracias Javier por permitirme aprender un año más bajo su guía, por todo el tiempo que ha dedicado en ayudarme,

Muchas gracias a mis padres y a mi hermana. Por su apoyo continuo, ayuda y consejos. Me siento muy afortunado de teneros,

Gracias también a todas las personas que directa o indirectamente me han influido a lo largo de este tiempo, en especial a mis amigos y a aquellos profesores que han sido fuente de motivación, y muchas otras personas que seguro me dejo por nombrar.

Muchas gracias.

Resumen

El SLAM (Localización y Mapeo Simultáneo) y SfM (Estructura a partir de Movimiento), son dos de las técnicas de mayor importancia actual que permiten, a través de información visual y entre otras funciones, localizar a un agente en el entorno que éste atraviesa. Una de las asunciones principales que es común en muchas de sus implementaciones, es que el entorno en el que los sensores capturan la información es predominantemente estático. En la práctica, la mayoría de estos sistemas, implementan técnicas que robustecen las estimaciones ante entidades dinámicas hasta cierto punto. Sin embargo, si la componente dinámica de la escena se vuelve significativa, tanto la localización como la geometría del entorno se vuelven erróneas.

En este TFM se aborda este problema. En concreto, se incorpora la estimación del movimiento, de 6 grados de libertad, que experimentan los objetos dinámicos presentes en la escena. Para ello, nos centramos en objetos rígidos, ajustando sus trayectorias a curvas B-Spline Cumulativas, las cuales presentan, entre otras propiedades, la ventaja de ofrecer estimaciones continuas en el tiempo de posición, orientación, velocidad y aceleración. Diferenciándonos así de los trabajos del estado del arte. Así mismo, se plantean estrategias que reducen el coste computacional de manera significativa, siendo aplicables a cualquier proyecto que emplee este tipo de curvas.

La evaluación de la propuesta muestra las ventajas de nuestro acercamiento: A pesar de estar imponiendo un modelo de trayectoria, tanto con datos sintéticos, como con una base datos pública, se obtienen resultados similares en precisión en cuanto a localización y orientación de los objetos dinámicos, a la vez que mejorando las estimaciones de la velocidad que éstos experimentan, en comparación con las estimaciones en tiempo discreto del estado del arte.

Abstract

SLAM (Simultaneous Localization and Mapping) and SfM (Structure from Motion) are two of the most relevant techniques that, among other use cases, allow the localization of an agent in the environment it traverses. Most of the current implementations assume that the environment where the data was captured remained static during the whole process. In practice, as long as this assumption is not serverely violated, current implementations can handle some amount of scene dynamism thanks to robustification techniques. Otherwise, both the estimated localization and geometry estimation of the surroundings get corrupted.

In this thesis we dive into this problem. Concretely, we take into account the motion of the dynamic objects that may be present in a scene and we estimate their 6 degrees of freedom. To do this, we focus on rigid objects, fitting their trajectories to cubic Cumulative B-Spline curves. This type of curve, among other poperties, has the advantage of offering continuous-time estimations of position, orientation, velocity and acceleration. This work thus, differ from the related works. Moreover, we propose estrategies which significantly reduce the computational cost, being applicable to any topic-related work that makes use of these kind of curves.

The evaluation shows the advantages of our approach: In spite of imposing a trajectory model, and in terms of precision in the estimation of localization and orientation of the objects, we obtain similar results than the state of the art in both public and synthetic data. Showing also better estimations in angular and linear velocity than the current discrete-time estimation works.

Índice general

Capítulo 1

Introducción

Imaginemos que un robot tiene que realizar una tarea en un entorno *desconocido* para él, como por ejemplo recoger ropa del suelo tal y como aparece en la Figura 1.1a. Para poder llevar a cabo esta tarea con éxito, el robot debe ser capaz no solo de conocer el sitio en el que se encuentra, sino también localizarse en él, para así evitar posibles modos de fallo como equivocarse de objeto de interés (ropa en este caso), o colisionar con otros elementos presentes en la escena (muebles).

La mayoría de las tareas a realizar por robots móviles, como el ejemplo mencionado en el párrafo anterior, necesitan un mapa del entorno para que los agentes que lo atraviesen puedan navegar de manera autónoma en él. En este caso, si se realiza conjuntamente la localización del robot con la creación del mapa del entorno y de forma secuencial (conforme el robot recibe datos sensoriales), este acercamiento al problema se conoce como **Localización y Mapeo Simultáneo** (o **SLAM**, de sus siglas en inglés). El SLAM ha experimentado un gran progreso en los últimos 30 años, convirtiéndose en una base fundamental de muchas aplicaciones en las que intervienen robots autónomos, o incluso realidad aumentada y realidad virtual [Cadena et al., 2016] (Figura 1.1b).

[2]https://www.bostondynamics.com/spot
[2]https://www.bbc.co.uk/taster/pilots/civilisations-ar

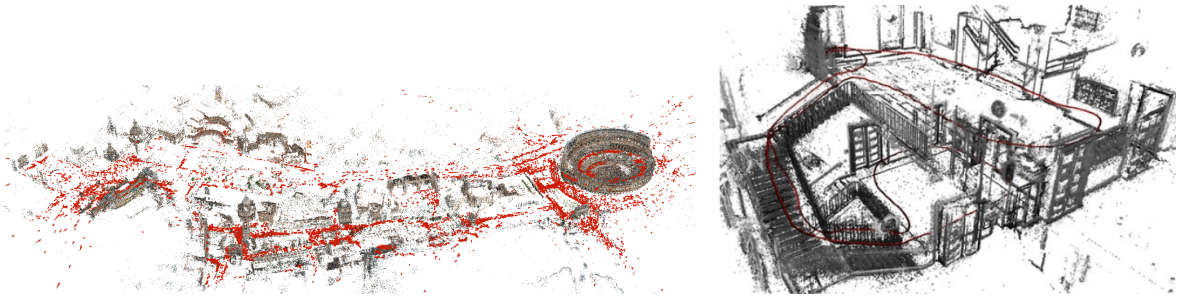
(a) (b) (c)

Figura 1.1: *(a) Robot Spot de la empresa Boston Dynamics[1]. (b) Aplicación de realidad aumentada Civilisations[2] que permite proyectar objetos virtuales relacionados con la historia y el arte. (c) Reconstrucción del Coliseo Romano obtenida aplicando técnicas de SfM [Agarwal et al., 2011]*

En cambio, si la estimación del mapa se realiza con datos sin ordenación temporal ni restricciones de ejecución en tiempo real, este acercamiento se conoce como **Estructura a partir de Movimiento** (o **SfM** de sus siglas en inglés), a partir del cual, además del objetivo mencionado anteriormente, se pueden llevar a cabo reconstrucciones tan impresionantes como las de la Figura 1.1c.

En cuanto a los sensores que se pueden emplear para este fin, en el campo de SLAM, los principales son los siguientes[3]:

- **Cámaras monoculares (RGB)** [Mur-Artal et al., 2015, Engel et al., 2017]: El sensor principal es una única cámara, a partir de la cual se estima la geometría y la localización del sensor, pero sujetas a una escala arbitraria (para obtener medidas reales hace falta multiplicar las mediciones realizadas por dicho factor de escala desconocido a priori),
- **Cámaras estéreo** [Mur-Artal and Tardós, 2017, Pire et al., 2017]: Los sensores principales son dos cámaras cuya posición y orientación relativas son conocidas. Es gracias a esto por lo que si un punto en el espacio es detectado y emparejado por ambas cámaras, es posible conocer su profundidad mediante triangulación, de una forma similar a cómo las personas somos capaces de percibir la profundidad de nuestro entorno cercano a través de nuestros ojos.



(a) (b)

Figura 1.2: *(a)* ***SfM****: Nube de puntos obtenida de una parte de la ciudad de Roma, tras haber usado 75.000 imágenes (los puntos rojos se corresponden con los lugares desde los que se capturaron) [Schonberger and Frahm, 2016]. (b)* ***SLAM****: Nube de puntos obtenida mediante una cámara monocular. Además, debido a que las imágenes se capturaron secuencialmente, es posible recuperar la trayectoria que trazó el sensor (representada como una línea roja) [Engel et al., 2017].*

- **Sensores inerciales** [Forster et al., 2016, Concha et al., 2016]: Constan de unidades de medición inercial (velocidad, orientación y aceleración), que combinados con información visual proveniente de los anteriores tipos de cámaras, dan lugar a estimaciones que actualmente son las más precisas en cuanto a la localización de los sensores en el entorno [Campos et al., 2021].
- **Cámaras RGB-D** [Newcombe et al., 2011, Schops et al., 2019]: Combinan la información visual de una cámara monocular con la información aportada por

[3]Parte de los trabajos citados combinan varios sensores, sin embargo, es dominante el sensor con el que se les ha relacionado. Así mismo, destacar que otros tipos de sensores muy presentes también en trabajos de SLAM como las cámaras de eventos [Kim et al., 2016a] o sensores de rango como el radar o el LIDAR [Wang et al., 2021] no se han incluido por su menor relación con este trabajo.

un sensor de profundidad (basado por ejemplo en la deformación de un patrón conocido de luz infrarroja proyectado en la escena), a partir del cual se obtiene la distancia relativa de la cámara con respecto al entorno visible.

Por otro lado en el campo de SfM, el sensor predominante es la cámara monocular [Schonberger and Frahm, 2016, Agarwal et al., 2011, Wu et al., 2011] son algunos ejemplos.

En la Figura 1.2 se muestran ejemplos de resultados obtenidos por trabajos mencionados anteriormente. En concreto, en este TFM se hace uso de datos provenientes de una cámara RGB-D. Los motivos de esta decisión están estrechamente ligados con los objetivos que se persiguen, los cuales se identifican y exponen en §1.1 - §1.3.

1.1. Motivación

Una de las asunciones principales que es común en muchos de los algoritmos de SLAM y SfM, es que el entorno en el que los sensores capturaron la información es predominantemente estático. Es decir, se asume que en dichos lugares, todas las entidades que aparecen en una escena y que perciben los sensores (estructuras, objetos, personas etc.) siempre se mantienen en el mismo sitio.

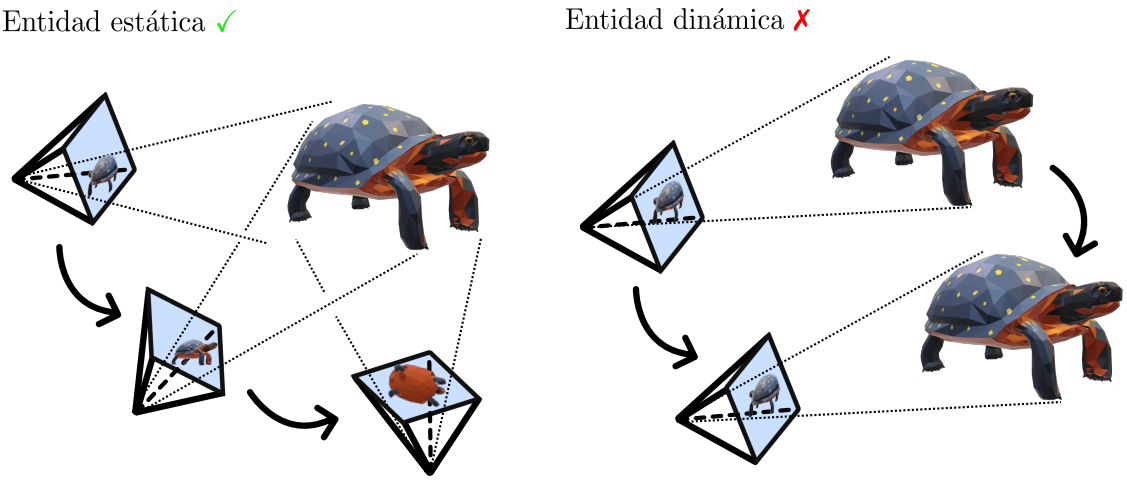


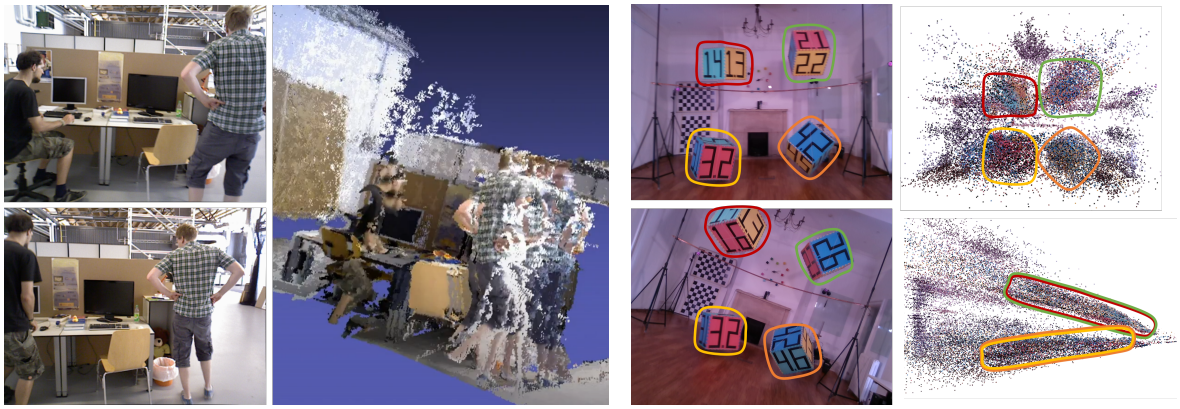
Figura 1.3: *(a) Gracias a observar identidades estáticas desde diferentes puntos de vista, podemos recuperar información como el movimiento de la cámara o la profundidad de lo observado. (b) En cambio, en escenas con entidades dinámicas (en este caso manteniendo constante la posición relativa entre cámara - objeto), al asumir erróneamente que la escena es estática, esto implicaría que, o bien, la cámara no se ha movido, o que el objeto se encuentra a una distancia* $\rightarrow \infty$ *y que por tanto al movernos, su proyección no cambia con respecto a la cámara (representada por:*).

Es gracias a esta asunción por la que, midiendo la localización de ciertas entidades en la referencia del sensor, es posible usar dicha información para mejorar localización

de los sensores y de dichas entidades. Un ejemplo de esto que pretende ser intuitivo, se muestra en la Fig. 1.3a. Sin embargo, si dichas entidades comienzan a moverse, puede dar lugar a un empeoramiento significativo en la localización del sensor, o en la estimación del mapa (Fig. 1.3b).

En la práctica, la mayoría de sistemas de SLAM [Cadena et al., 2016] y de SfM [Schonberger and Frahm, 2016], implementan técnicas que *robustecen* las estimaciones ante entidades dinámicas hasta cierto punto. Sin embargo, si la componente dinámica de la escena se vuelve significativa, tanto la localización como la geometría del entorno se vuelven erróneas. Dos ejemplos reales de esto se muestran en la Figura 1.4.

En este TFM el objetivo es abordar este problema. En concreto, en **incorporar la estimación del movimiento que experimentan los objetos dinámicos presentes en la escena**. Para ello, nos centramos en objetos rígidos, es decir, aquellos cuya deformación a lo largo del tiempo es despreciable, por lo que la distancia entre los puntos que los conforman es constante.



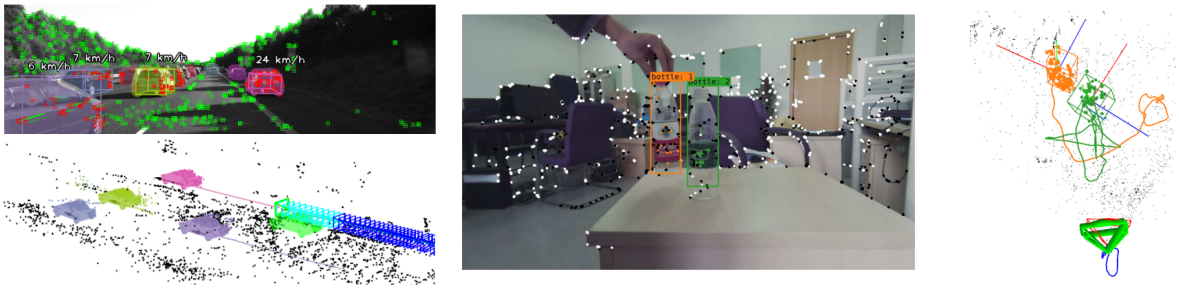
(a) (b)

Figura 1.4: *(a, izda) Par de instantes del dataset [Sturm et al., 2012a], en el que dos personas se mueven por el lugar en el que los sensores están captando información. (a, dcha) Mapa estimado mediante* ***SLAM*** *[Bescos et al., 2018]. Debido a la asunción de estaticidad, las personas aparecen repetidas veces en él. (b, izda) Par de instantes del dataset [Judd and Gammell, 2019] en el que aparen cajas moviéndose mediante un mecanismo de poleas. (b, dcha) Mapa estimado (visto de frente y de perfil) mediante* ***SfM****. En este caso, el movimiento de los objetos provoca la aparición de una estela de puntos que erróneamente se atribuye a entidades estáticas.*

1.2. Trabajo relacionado

Abordar este problema mediante la incorporación del movimiento de los objetos rígidos presentes en la escena, ha sido tratado desde diferentes puntos de vista en la literatura. Por ello, en esta sección se realiza un análisis del estado del arte, con el objetivo de detectar cuestiones que hasta la fecha no han sido estudiadas y que podrían suponer contribuciones al estado del arte.

Uno de los primeros trabajos que trató de estimar el movimiento de las entidades



(a) (b)

Figura 1.5: Resultados obtenidos por el estado del arte. *(a) DynaSLAM II [Bescos et al., 2021]. Obsérvese la estimación conjunta del mapa estático (puntos negros), así como las trayectorias de los objetos dinámicos (coches). (b) ClusterVO [Huang et al., 2020]. Se muestra como este sistema es capaz de seguir la trayectoria de dos botellas que intercambian sus posiciones.*

dinámicas presentes en la escena fue [Tomasi and Kanade, 1992], en el que se introdujo la técnica de **factorización**, permitiéndoles llevar a cabo conjuntamente la detección del movimiento del objeto, así como su reconstrucción. Para ello, se asumía la presencia de un único objeto, así como un modelo de cámara ortográfica, la cual permite simplificar los cálculos pero introduce error al no poder modelar la proyección perspectiva de una cámara real [Hartley and Zisserman, 2004].

Sucesivos trabajos hicieron frente a estas limitaciones, incorporando la estimación del movimiento de múltiples objetos [Han and Kanade, 2004, Zappella et al., 2013], así como el uso de modelos de cámara perspectiva [Sturm and Triggs, 1996], o incluso ambas mejoras [Sabzevari and Scaramuzza, 2014]. Sin embargo, la mayoría de estos métodos presentan inconvenientes como no poder trabajar de manera secuencial, asumir de antemano tipos concretos de movimiento, o tener un alto coste computacional [Saputra et al., 2018].

Otro tipo de trabajos se centran en lo que se conoce como **triangulación de una trayectoria** [Avidan and Shashua, 2000]. Éstos tratan de ajustar la trayectoria que siguen puntos que no son estáticos mediante distintas entidades geométricas predefinidas, tales como líneas y secciones cónicas [Avidan and Shashua, 2000, Park et al., 2010], o superficies más complejas que permiten linealizar la trayectoria [Kaminski and Teicher, 2004]. Sin embargo, estos trabajos se enfocan en estimar la trayectoria de puntos por separado, en vez de asociarla a un objeto, por lo que la rotación que podría experimentar un cuerpo no es considerada.

En el campo de **SLAM**, uno de los primeros trabajos en estimar el movimiento de los objetos fue [Wang et al., 2003] (extendido en [Wang et al., 2007]), en el que dieron un enfoque probabilístico permitiendo incorporar la utilización de filtros Gaussianos. Sus resultados mostraron mejoras con respecto a únicamente estimar la localización del sensor en entornos dinámicos. A la misma conclusión se llega en [Bibby and Reid, 2007], en el que se estima la localización de los puntos dinámicos de la escena. En estos trabajos, así como otros [Bibby and Reid, 2010, Li et al., 2018, Yang and Scherer, 2019] más recientes, los objetos estudiados se mueven en un plano, los cuales son comunes en entornos como los de la conducción autónoma. También nos encontramos con estas

asunciones en el campo de **SfM** [Kundu et al., 2011].

Trabajos más recientes permiten estimar **movimientos generales** de los objetos. En concreto, [Zhang et al., 2020, Bescos et al., 2021] proponen detectar en una primera fase los objetos dinámicos presentes en una escena mediante técnicas de *deep learning* como [He et al., 2017], para a continuación estimar su movimiento y combinarlo con el cálculo de la localización del sensor. En [Huang et al., 2020] añaden un criterio probabilístico para la detección de los objetos. Por otro lado, [Judd et al., 2018, Judd and Gammell, 2020] proponen refinar la detección y seguimiento de la trayectoria de manera conjunta, haciendo uso para ello de optimizadores de modelos geométricos [Amayo et al., 2018]. Estos sistemas han sido evaluados en bases de datos públicas [Judd and Gammell, 2019, Geiger et al., 2012] obteniendo resultados prometedores (en la Figura 1.5 se muestran ejemplos de resultados).

En estos últimos trabajos, los sensores empleados son cámaras **estéreo** o **RGB-D**, los cuales evitan la dificultad de tener que triangular puntos en movimiento en diferentes instantes de tiempo al poder inferir la profundidad con tan solo una imagen. En este TFM seguimos esta tendencia haciendo uso de datos provenientes de cámaras RGB-D.

Algo que tienen en común la práctica totalidad de los trabajos anteriores es que éstos operan en **tiempo discreto**; es decir, estiman su ubicación en el espacio para cada instante del que se dispone una imagen del mismo. De esta forma, las trayectorias que siguen no tienen por qué presentar continuidad en su velocidad y aceleración. En la Figura 1.6a se muestra un ejemplo simple de esto.

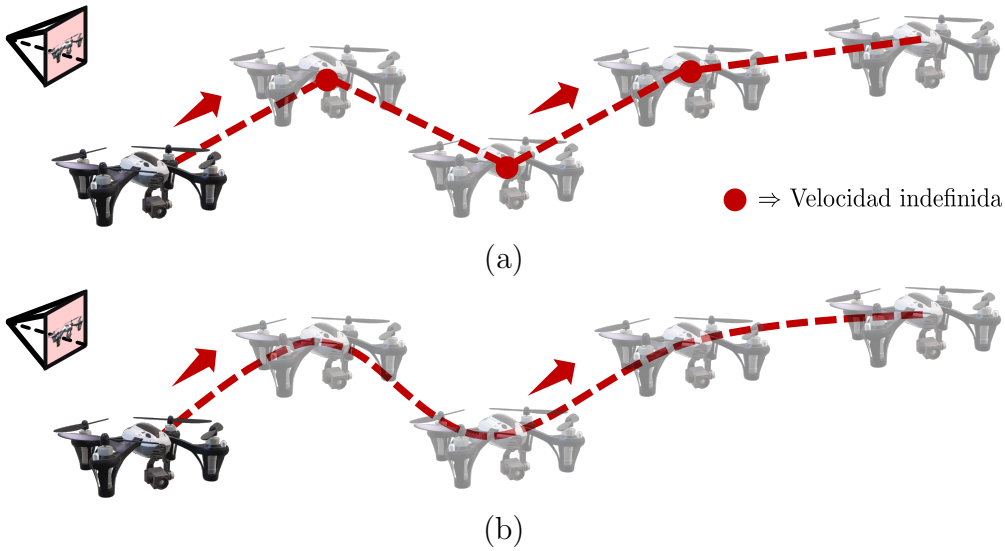


Figura 1.6: *(a)* ***Trayectoria definida en tiempo discreto****: En la trayectoria recuperada, al estar definida por tramos, aparecen puntos donde la velocidad estimada (pendiente) cambia bruscamente de dirección, provocando que no esté definida en dichos puntos. (b)* ***Trayectoria definida en tiempo continuo****: En este caso, la trayectoria estimada nos permite obtener un valor de posición y velocidad (y aceleración, como se verá más adelante) para cualquier instante temporal, no solo en aquellos donde el objeto fue observado.*

Por ello, se pensó que podía ser interesante que las trayectorias estimadas de los

objetos presentasen **continuidad temporal** tanto en su velocidad como en la aceleración (Figura 1.6b). De los trabajos anteriores, únicamente [Bibby and Reid, 2010] trató este tema, pero en él se centraron en objetos que siguen un movimiento plano y por tanto no general (de 6 grados de libertad), además de usar radares como sensor, cuya disponibilidad hoy en día es menor que la de una cámara RGB-D.

En una línea similar, [Lovegrove et al., 2013] propuso ajustar la trayectoria del sensor a una curva continua haciendo uso de **B-Splines Cumulativos**. Debido a sus propiedades (secuencialidad, continuidad en la velocidad y aceleración, entre otras) presentan una opción muy interesante. Trabajos posteriores extendieron sus aplicaciones a diferentes clases de sensores [Kerl et al., 2015, Mueggler et al., 2015, Yang et al., 2021, Droeschel and Behnke, 2018], pero su uso para interpolar los 6 grados de libertad de movimientos rígidos de objetos capturados por sensores visuales queda aún por explorar.

1.3. Objetivos

En base al análisis previo, se proponen los siguientes objetivos a cumplir:

1. **Cuantificar el error** en la construcción del mapa, así como en la localización del sensor, derivado de no considerar los objetos dinámicos presentes en la escena. Usaremos para ello el software COLMAP [Schonberger and Frahm, 2016], uno de los sistemas de SfM con mayor precisión del estado del arte.

2. Proponer un **sistema secuencial que permita estimar en tiempo continuo el movimiento general de los objetos** presentes en una escena, haciendo uso para ello de B-Splines Cumulativos. Así mismo, proponer estrategias que permitan la reducción del coste computacional.

3. **Evaluar la propuesta**, haciendo uso de bases de datos públicas, así como datos sintéticos, que nos permitan comparar los resultados obtenidos con los del resto del estado del arte reciente.

1.4. Estructura

Con los anteriores objetivos en mente, este TFM está estructurado de la siguiente forma:

- **Capítulo** 2: Presentación del sistema propuesto (partes fundamentales del mismo).

- **Capítulo** 3: Formulación matemática de la propuesta. En concreto, se presenta la teoría básica del grupo de Lie de $SE(3)$, empleado para la estimación de trayectorias en 3D (orientación y traslación), relacionándola a continuación con la estimación en tiempo continuo mediante B-Splines.

- **Capítulo** 4: Explicación de 1) variables asociadas a la trayectoria que se optimizan en nuestra propuesta y cómo se relacionan, 2) método de optimización y 3) derivación de las matrices Jacobianas con el objetivo de reducir el tiempo de cómputo.
- **Capítulo** 5: Técnicas usadas para asociar los objetos en distintas imágenes, además de extraer la información geométrica necesaria e introducida en el bloque de optimización.
- **Capítulo** 6: Experimentos realizados, así como los resultados obtenidos, haciendo uso de datos generados artificialmente, así como de una base de datos pública.
- **Capítulo** 7: Conclusiones del trabajo realizado y trabajo futuro.

Capítulo 2

Sistema propuesto

En este capítulo se realiza un visionado general del sistema propuesto, y de cómo se obtienen sus entradas. Se relega a los siguientes capítulos la explicación en detalle de las partes que lo conforman. En la Figura 2.1 se muestra un esquema del mismo.

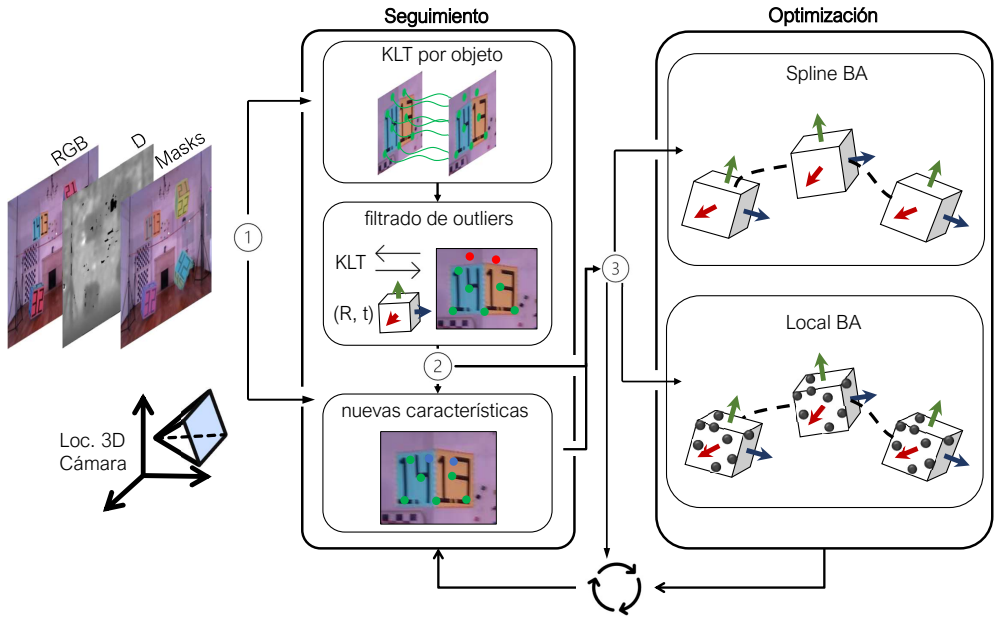


***Figura 2.1: Sistema propuesto**. Esta constituido por dos partes fundamentales: 1) El bloque de seguimiento se encarga de emparejar los datos de entrada a objetos cuya trayectoria ya está siendo seguida, o en su defecto inicializar las trayectorias, y 2) El bloque de optimización, encargado de ajustar la trayectoria en tiempo continuo de cada objeto a los datos de entrada.*

Las **entradas** consideradas son: 1) una imagen RGB-D -y el instante temporal en el que se capturó-, 2) una estimación de la localización 3D del sensor que capturó dicha imagen (obtenida de forma offline) y 3) una estimación del lugar en la imagen en el que se encuentran los objetos a los que hay que realizar el seguimiento (obtenida de forma offline). Como **salidas** se obtienen las trayectorias en tiempo continuo (B-Spline cumulativos) para cada objeto presente en la escena.

2.1. Bloques fundamentales

El sistema consta de dos partes fundamentales: seguimiento y optimización. La parte del **seguimiento** es la encargada de asociar los objetos contenidos en las máscaras de entrada a objetos que ya estaban siendo seguidos, o en su defecto, detectar que un objeto es nuevo, y por tanto inicializar una nueva trayectoria (decisión marcada como ① en la Figura 2.1).

La inicialización de una trayectoria se realiza a través de la detección de *características salientes* [Shi et al., 1994] en la zona de la imagen en la que se encuentra el objeto de interés. Como se verá más adelante, es necesario al menos disponer de 4 instantes en los que el objeto ha sido capturado por el sensor. Por ello, como inicialmente no se cumple esta condición (representada por ③), se espera a la llegada de la siguiente imagen en la que aparezca el objeto.

Cuando llega una nueva imagen que contiene a un objeto con una trayectoria inicializada, se calcula el *flujo óptico* [Lucas et al., 1981] de las características salientes detectadas en el frame (imagen) anterior. Las características cuyo flujo ha sido calculado con éxito, se someten a un filtrado para evitar que puntos que no pertenezcan al objeto se traten como tal, lo que empeoraría la estimación de su trayectoria. A estos puntos se les conoce como *outliers*.

A continuación, si se dispone de un número suficiente de características se procede a la optimización de la trayectoria del objeto. Si esta condición no se cumple (representada por ② en la Fig. 2.1), se procede a detectar nuevas características salientes, imponiendo uniformidad en su detección para que así éstas no se concentren únicamente en zonas concretas.

El bloque de **optimización** recibe la ubicación de la cámara, así como la ubicación relativa a ella de los puntos que han sido inicializados/ seguidos en una ventana de los N últimos frames en los que el objeto ha aparecido. El objetivo de este bloque es estimar/ actualizar la curva en tiempo continuo (B-Spline Cumulativo) que mejor se ajusta a estos datos. Para ello se emplean técnicas de optimización no lineal de segundo orden.

En función del número de puntos recibido (condición ③) se lleva a cabo una optimización de tipo *Spline BA* (solo se optimiza la trayectoria) o *Local BA* (se optimiza además la localización de los puntos del objeto). La formulación matemática de este bloque se explica en §3.

2.2. Modelo de cámara

Un modelo de cámara determina cómo la realidad 3D se proyecta en las imágenes 2D. En este TFM se considera un modelo de **cámara pinhole**, el cual es uno de los más extendidos en el campo de visión por computador [Corke, 2011]. Este modelo, asume que los rayos de luz provenientes de los puntos que observa una cámara convergen en

un único punto, *el centro óptico* $C \in \mathbb{R}^3$, y que éstos se proyectan en un plano (*plano imagen*) formando así la imagen que nosotros observamos.

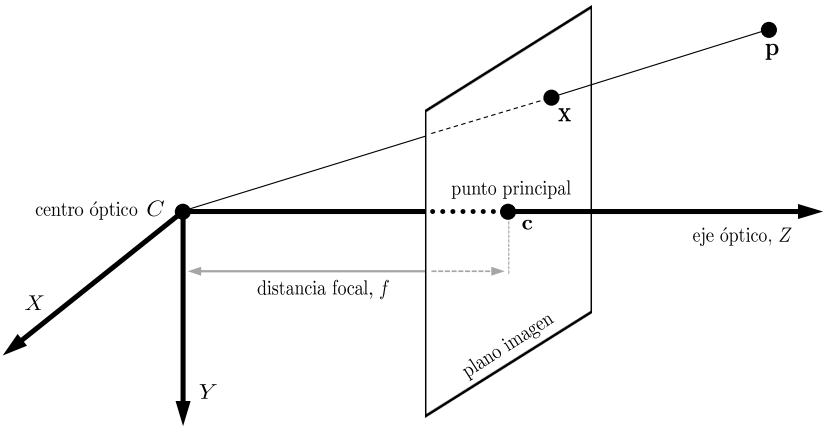


***Figura 2.2: Modelo de cámara pinhole**. Al converger en el centro óptico C el rayo de luz que proviene de un punto* $\mathbf{p}$*, éste aparece proyectado en el plano imagen en las coordenadas* $\mathbf{x}$*. Los ejes* (X, Y, Z) *definen, por convención, el sistema de coordenadas de la cámara. El plano imagen es perpendicular al eje* Z *(su intersección define la ubicación del centro óptico* $\mathbf{c}$*) y está situado a una distancia* f *(distancia focal) de* C*.*

Por convención [Hartley and Zisserman, 2004, Corke, 2011], para localizar en el espacio a la cámara, el sistema de coordenadas que lo determina es el representado en la Figura 2.2, es decir, el origen se encuentra en el centro óptico, C, el eje Z (denominado como eje óptico) es perpendicular al plano imagen y apunta hacia él, y los ejes X e Y son paralelos a los bordes de la imagen. El corte del eje óptico con el plano imagen determina la ubicación del *punto principal*: $\mathbf{c} = [c_x, c_y]^T$, y la distancia que los separa se denomina *distancia focal*: $\mathbf{f} = [f_x, f_y]^T$ (ambas en unidades de píxeles)[1].

Por otro lado, para representar un punto, $\mathbf{x} \in \mathbb{R}^2$, en la imagen, se utiliza un sistema de referencia de dos dimensiones cuyo origen se encuentra en el centro del píxel superior izquierdo, con ambos ejes paralelos a los bordes de la imagen. De esta forma: $\mathbf{x} = [u, v]^T$.

Reuniendo todo lo anterior, la proyección de un punto en el espacio, $\mathbf{p} \in \mathbb{R}^3$, viene dada por la Ec. 2.1.

$$\lambda \tilde{\mathbf{x}} = \mathbf{K} \begin{bmatrix} \mathbf{R}_{cw} & \mathbf{t}_{cw} \end{bmatrix} \tilde{\mathbf{p}}_w \tag{2.1}$$

Donde $\mathbf{R}_{cw} \in SO(3)$ y $\mathbf{t}_{cw} \in \mathbb{R}^3$ representan la matriz de rotación y vector de traslación que transforman puntos desde un sistema de coordenadas w (mundo) al sistema de coordenadas de la cámara c. La tilde $(\tilde{\cdot})$ sirve para expresar un punto en coordenadas homogéneas: $\mathbf{p} \in \mathbb{R}^3 \rightarrow \tilde{\mathbf{p}} \in \mathbb{P}^3$, por lo que λ representa un factor de escala arbitrario. Finalmente, $\mathbf{K}$ es la *matriz de calibración* de la cámara, la cual viene dada por:

$$\mathbf{K} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.2}$$

[1] En realidad solo hay una distancia focal, f (dada en mm por ej.), sin embargo las dos componentes f_x, f_y surgen de expresarla en unidades de píxeles. Se usa de esta última versión, ya que la proyección de un punto consiste en una transformación de coord. *métricas* a coord. *píxel*. La conversión viene dada por: $f_x = f/\rho_w, f_y = f/\rho_h$, donde ρ_w y ρ_h son la anchura y altura de cada píxel.

Para expresar la Ec. 2.1 en coordenadas cartesianas, se utiliza la función de proyección $\pi(\cdot)$:

$$\mathbf{x} = \pi(\begin{bmatrix}\mathbf{R}_{cw} & \mathbf{t}_{cw}\end{bmatrix}, \mathbf{p}) = \pi(\mathbf{T}_{cw}, \mathbf{p}_w) \tag{2.3}$$

Donde $\mathbf{T}_{cw} \in SE(3)$ representa la *matriz de transformación* que igualmente permite transformar puntos desde la referencia mundo w, a la referencia cámara c:

$$\mathbf{T}_{cw} = \begin{bmatrix}\mathbf{R}_{cw} & \mathbf{t}_{cw} \\ \mathbf{0}_{1\times 3} & 1\end{bmatrix} \tag{2.4}$$

2.3. Segmentación de objetos dinámicos

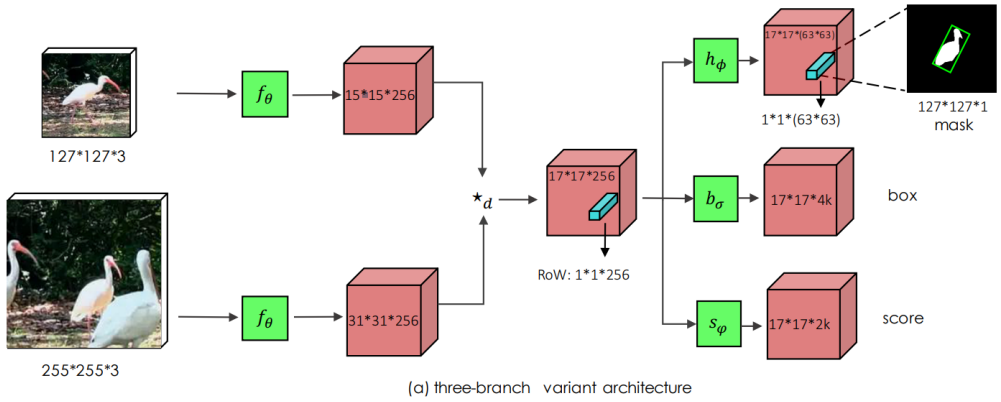


Figura 2.3: Arquitectura propuesta en SiamMask *[Wang et al., 2019]. Las entradas se corresponden con una imagen ejemplar (más pequeña), z, y una imagen de búsqueda, x. Ambas son procesadas por la misma red neuronal, obteniendo los tensores $f_\theta(x)$ y $f_\theta(z)$. A continuación se calcula la correlación entre cada par de canales: $f_\theta(x) \star f_\theta(z)$. El tensor resultante es procesado por separado en 3 redes convolucionales: h_ϕ, b_σ y s_ϕ. La red h_ϕ ofrece múltiples predicciones de posibles máscaras, y s_ϕ puntúa cada una de ellas. La predicción final de la arquitectura se corresponde con la máscara de mayor puntuación.*

Tal y como se ha comentado en la sección anterior, una de las entradas al sistema se corresponde con *máscaras* que estiman la posición en la imagen de los objetos que queremos estimar su movimiento. En nuestro caso, estas máscaras las obtenemos a partir de la arquitectura propuesta en [Wang et al., 2019], conocida como **SiamMask** (ver Figura 2.3). En concreto la implementación ofrecida en la librería *pysot*[2].

La razón principal por la que se eligió esta arquitectura se debe a que puede detectar/ seguir objetos de cualquier clase. Es decir, no está limitada a detectar objetos específicos tales como: silla, coche, gato etc. pudiendo ser, por tanto, utilizada en objetos que son difíciles de categorizar, como los de la Figura 2.4c. Gracias a esto es posible ahorrar el tiempo necesario que requiere re-entrenar redes que sí dependen de las clases de los objetos ([He et al., 2017] es un ejemplo de este otro tipo de arquitecturas).

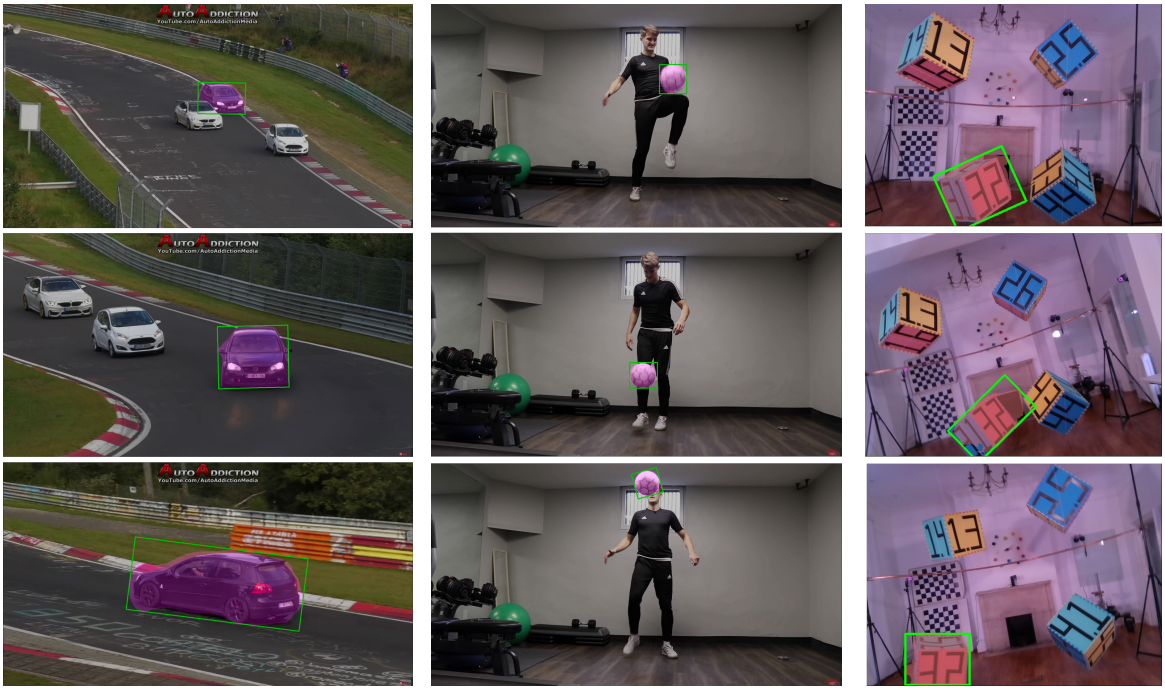
Para tener una ligera intuición de cómo SiamMask consigue esta versatilidad, a continuación se expone *brevemente* su funcionamiento. Tal y como se ve en su arquitectura

[2] `https://github.com/STVIR/pysot`

(Figura 2.3), se utilizan dos entradas: una imagen que contiene el objeto a encontrar (dado por el usuario en el primer frame), z, y una imagen más grande, x, en la que hay que encontrar el objeto contenido en z.

Ante esto, una posible opción sería comparar *directamente* la imagen z sobre distintas zonas de la imagen x e ir calculando cómo de similares son ambas, basándonos en su correlación. Valores más altos de correlación indicarían zonas que presentan una mayor relación lineal y que por tanto podríamos pensar que son más similares. En este caso, z actúa como filtro (o *kernel*).

Sin embargo, esta comparación algunos autores [Bolme et al., 2010] la denominan como ingenua ya que puede generar valores altos de correlación no solo en la zona correcta, sino también en diferentes zonas de la imagen. Para hacer frente esto, lo que se propone en SiamMask, es aprender este filtro a través de una red siamesa convolucional [Bertinetto et al., 2016] que transforma las imágenes x y z en dos representaciones $f_\theta(x)$ y $f_\theta(z)$ tales que su correlación (representada en la Fig. 2.3 como $\star$d) discrimine únicamente al objeto de interés.



(a) (b) (c)

Figura 2.4: Resultados obtenidos con SiamMask *[Wang et al., 2019]. El sistema es capaz de seguir correctamente a los objetos en la mayor parte de la secuencia, incluso en situaciones donde el punto de vista de los mismos es diferente - por ej. ver (a). Sin embargo, en ciertas imágenes, la presencia de outliers puede ser significativa - ver (c). Nuestro sistema deberá ser robusto ante este tipo de situaciones.* links[3]

A continuación, el resultado de esta correlación es propagado a través de 3 redes convolucionales [Goodfellow et al., 2016] que vienen representadas en la Fig. 2.3 por: h_ϕ, b_σ y s_ϕ. h_ϕ da como resultado múltiples estimaciones de la máscara del objeto, y

[3]Links a los vídeos sobre los que se les a aplicado SiamMask: `Vídeo (a)`, `Vídeo (b)`, `Vídeo (c)`.

s_ϕ la puntuación de cada una de ellas, eligiendo finalmente como predicción, aquella que presenta una mayor puntuación.

En la Figura 2.4 se muestran ejemplos de resultados obtenidos tras aplicar SiamMask en diferentes secuencias. Destacar que la breve explicación anterior solo tiene como objetivo dar una ligera intuición sobre cómo estos datos de entrada de nuestro sistema son obtenidos. Detalles como las propiedades de la arquitectura o su forma de entrenarla han sido omitidas, referimos al lector al excelente artículo original [Wang et al., 2019] para más información.

2.4. Estimación del movimiento de la cámara

La otra de las *entradas* a nuestro sistema que debemos obtener a partir de las imágenes del sensor RGB-D, consiste en la localización en el espacio que tenía la cámara para cada uno de los instantes de los que disponemos una imagen. Esta estimación se realiza a partir del software de SfM **COLMAP** [Schonberger and Frahm, 2016]. Las razones principales detrás de esta elección son su mayor precisión en comparación con el resto del estado del arte, además de ser de código abierto[4].

Al igual que en la sección anterior, con el objetivo de ganar intuición acerca de cómo estos datos de entrada son estimados, se va explicar brevemente las partes fundamentales del software. Para más detalles del mismo, referimos al lector al excelente artículo original.

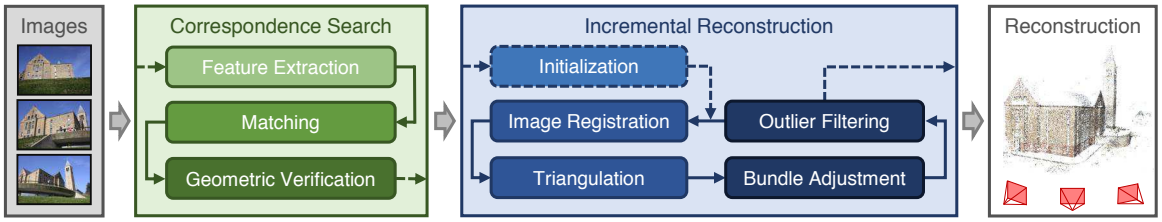
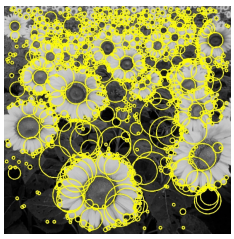


***Figura 2.5: Secuencia de bloques que conforman la estructura de COLMAP**. Figura extraída del propio artículo [Schonberger and Frahm, 2016].*

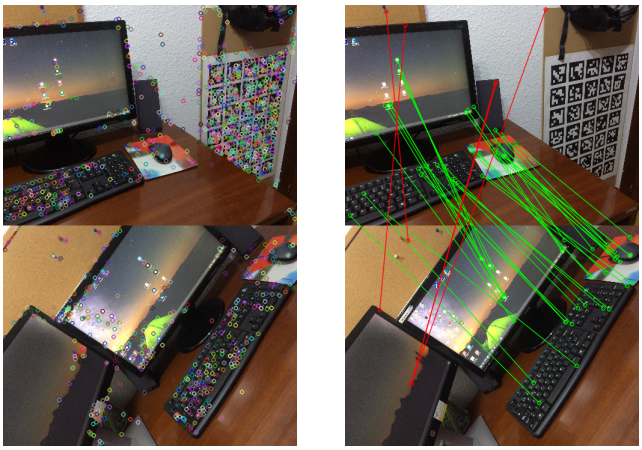
La estructura de COLMAP (o del SfM *incremental*) viene dada por la secuencia de bloques que aparecen en la Figura 2.5. El primero de ellos (*Feature Extraction*) consiste en la extracción de *características salientes* en la imagen. Éstas son puntos (píxeles) que satisfacen determinadas condiciones dependiendo del algoritmo usado. En concreto, COLMAP emplea características SIFT [Lowe, 2004b], las cuales seleccionan puntos cuyo entorno (píxeles vecinos) presentan unas propiedades $\sim$ constantes (en color o brillo) que difieren del resto de píxeles que los rodean. Estas zonas son conocidas como "*blobs*". A la derecha se muestra un ejemplo intuitivo extraído de [Lowe, 2004a], en el que los "*blobs*" detectados aparecen rodeados por círculos amarillos.



[4] `https://github.com/colmap/colmap`

Estas características son extraídas para toda la secuencia, con el objetivo de producir detecciones de un mismo punto en distintas imágenes. Estas redundancias se detectan, o mejor dicho, se emparejan, mediante el segundo bloque (*Matching*) en el cual se comparan los *descriptores* de cada característica SIFT detectada en imágenes diferentes [Arandjelović and Zisserman, 2012] (para disminuir el tiempo de cómputo, solo se consideran las imágenes más similares visualmente, de acuerdo a un criterio de vocabulario visual [Nister and Stewenius, 2006]).

Un emparejamiento entre dos características (de dos imágenes distintas) se obtiene si mutuamente sus descriptores son los más próximos entre sí -cuantificados por la suma de sus diferencias cuadráticas). En la Figura 2.6 se muestra un ejemplo de detección y emparejamiento de características SIFT entre dos imágenes.



(a) (b)

***Figura 2.6: Ejemplo de detección y emparejamiento de características SIFT**. (a) Detecciones en ambas vistas. Notar que aquellas zonas que se asemejan a "blobs" (letras del teclado, iconos de la pantalla etc.) son detectados. (b) Emparejamientos. En verde se muestran aquellos verificados geométricamente mediante la matriz fundamental que relaciona ambas vistas.*

Como los emparejamientos realizados anteriormente solamente se basan en información visual, es posible que dos puntos diferentes sean emparejados si sus entornos son similares (por ejemplo zonas con patrones repetitivos como el panel de corcho de la Figura 2.6a). Por ello, éstos se verifican en el siguiente bloque (*Geometric Verification*), atendiendo a información geométrica de la escena.

Esta verificación consiste en calcular la *geometría epipolar* existente entre las dos vistas (imágenes) a través de la *matriz fundamental*, $\mathbf{F}$, que la define [Hartley and Zisserman, 2004]. Esta verificación se puede entender visualmente a través de la Figura 2.7. En ella, un punto, $\mathbf{p} \in \mathbb{R}^3$ es proyectado en dos vistas, obteniendo sus coordenadas $\mathbf{x}_a, \mathbf{x}_b \in \mathbb{P}^2$. En esta situación *genérica*, el plano que contiene a estos puntos, contiene además a los centros ópticos, $\mathbf{c}_1, \mathbf{c}_2$, de las dos vistas, lo que obliga a que $\mathbf{x}_b$ deba encontrarse en la línea definida por el corte de este plano con la segunda vista, $\mathbf{l}_b$ (en coordenadas homogéneas), y viceversa. Estas líneas se denominan *epipolares*.

En esta situación, la matriz fundamental (estimada de manera *robusta* [Hartley and Zisserman, 2004]) es la que define la transformación de un punto $\mathbf{x}_i$ cualquiera de una

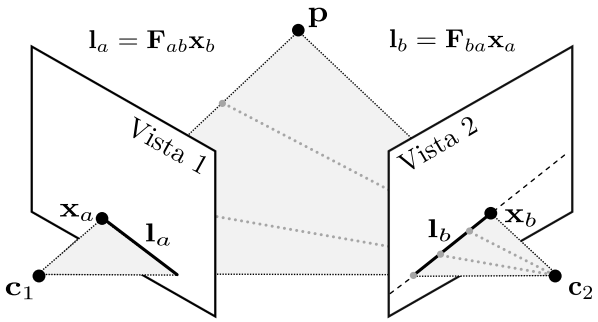


Figura 2.7: *El plano que contiene a un punto* $\mathbf{p}$ *y a sus proyecciones en dos vistas diferentes* $\mathbf{x}_a, \mathbf{x}_b$*, contiene además los centros ópticos correspondientes a ambas vistas* $\mathbf{c}_1, \mathbf{c}_2$*, obligando a que* $\mathbf{x}_a$ *y* $\mathbf{x}_b$ *se encuentren en las líneas* $\mathbf{l}_a$ *y* $\mathbf{l}_b$ *(intersecciones del plano con las vistas). La matriz fundamental* $\mathbf{F}$*, define la transformación de un punto en la imagen* $\mathbf{x}_i$ *a la correspondiente línea epipolar.*

imagen, a su línea epipolar correspondiente en la otra imagen. En el caso de la Figura 2.7:

$$\mathbf{l}_b = \mathbf{F}_{21}\mathbf{x}_a, \qquad \mathbf{l}_a = \mathbf{F}_{12}\mathbf{x}_b \tag{2.5}$$

Para que un emparejamiento entre imágenes sea verificado geométricamente, debe haber un mínimo número de puntos que cumplan esta condición (dentro de un margen de error). A modo de ejemplo, en la Figura 2.6b se muestran en rojo, los emparejamientos que no cumplen esta condición, coincidiendo con aquellos que son incorrectos.

Finalmente, para cada par de imágenes emparejado que ha sido verificado geométricamente, COLMAP, calcula las matrices *esencial* y de *homografía* [Hartley and Zisserman, 2004] que las relaciona, permitiendo, entre otras funciones, discernir si el movimiento relativo entre las dos vistas ha sido únicamente de rotación, o si la escena observada es predominante plana (por ejemplo, si se está observando una pared).

Esta información es aprovechada en el siguiente bloque, *Initialization*, el cual tiene como objetivo la inicialización del mapa. Para ello, COLMAP elige un par de imágenes verificado que presenta un alto número de características emparejadas, y una traslación suficiente entre las mismas favoreciendo así la triangulación de los puntos observados por ambas vistas[5]. Esto se realiza a través de las matrices esenciales, siguiendo el procedimiento explicado en [Nistér, 2004, Hartley and Zisserman, 2004]. Un ejemplo de inicialización se muestra en la Fig. 2.8.

De aquí en adelante, COLMAP trata de aumentar el número de puntos del mapa, así como el número de cámaras (la estimación de su localización). Para ello, en primer lugar, el bloque de *Image Registration*, se encarga de estimar (registrar) la ubicación de una nueva cámara. Esta estimación se realiza a partir del mapa ya creado, de manera robusta haciendo uso de P3P [Gao et al., 2003] . Para elegir, qué cámara registrar, COLMAP impone un doble criterio: un número suficiente de sus características SIFT verificadas deben haber sido ya trianguladas,

[5]De manera intuitiva, si dos imágenes están tomadas desde un mismo sitio, los rayos asociados a los puntos observados serían coincidentes en ambas imágenes ⇒ No habría un único punto de corte y por tanto sería imposible triangular. Por ello se fija un traslación entre cámaras (o ángulo entre los rayos) mínima para favorecer la triangulación.

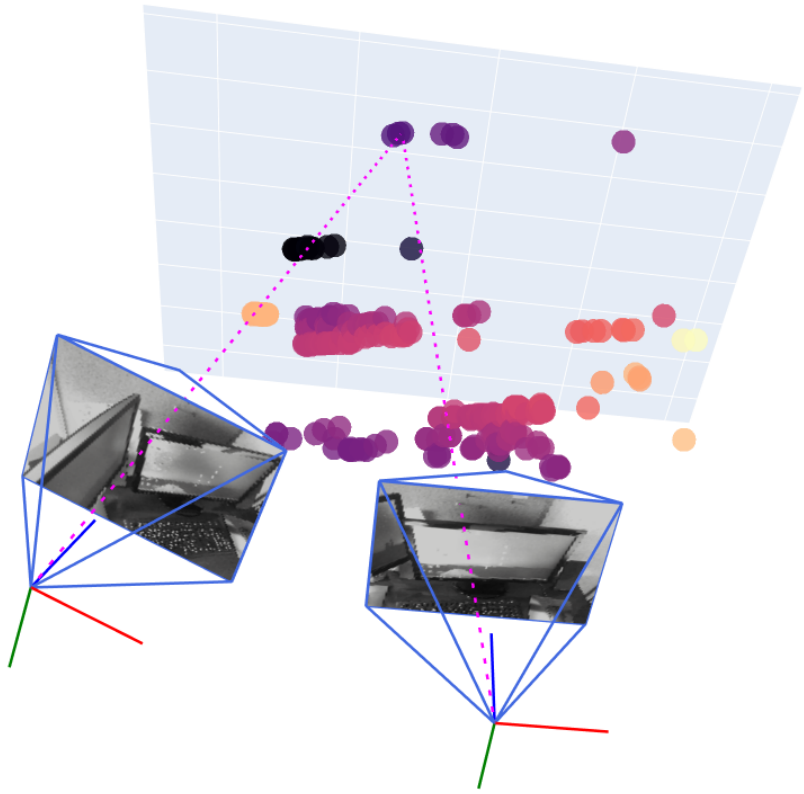
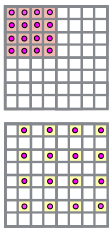


***Figura 2.8: Ejemplo simplificado de inicialización**. Partiendo de dos imágenes tomadas con suficiente separación y características emparejadas: 1) Se extrae la matriz esencial que relaciona las vistas, 2) se extrae la posición y orientación relativa de ambas cámaras, y 3) se triangulan las características emparejadas, obteniendo así un mapa inicial (nube de puntos) sobre el que se irán añadiendo más cámaras/ puntos. En este ejemplo también se muestran el par de rayos (▬ ▬) proveniente de un punto triangulado.*

y éstas deben presentar una distribución uniforme en su imagen. Por ello, de las dos "imágenes" de la derecha, la inferior sería elegida antes para registrar (Figura extraída del artículo original).



Al haber estimado la localización de más cámaras, es posible que *nuevos* puntos sean triangulados (bloque *Triangulation*), aumentando así el mapa reconstruido. Para ello, COLMAP impone que un punto candidato a ser triangulado debe ser observado por al menos tres imágenes, y además presentar un ángulo de triangulación suficiente para asegurar su fiabilidad. Esto se realiza de manera robusta mediante DLT [Hartley and Zisserman, 2004].

Para refinar las estimaciones realizadas en los dos bloques anteriores, COLMAP utiliza *Ajuste de haces* o ***Bundle Adjustment*** (**BA**) [Triggs et al., 1999]. BA es un método de optimización no lineal a partir del cual el error de reproyección es minimizado, optimizando así tanto la localización de las cámaras como la de los puntos triangulados. En este caso, COLMAP lo aplica localmente, es decir, solamente sobre el conjunto de imágenes que comparten información visual. Este tipo de optimización también se utiliza en nuestra propuesta por lo que se explicará con más detalle más adelante (§4).

Este tipo de optimización se repite de manera global (*BA global*) cada vez que el mapa ha crecido un determinado porcentaje, refinando así *todos* los parámetros (localizaciones de los puntos y cámaras) estimados hasta el momento. Si tras cualquiera de estas optimizaciones, hay puntos que presentan un error de reproyección elevado (o un ángulo de triangulación pequeño en cualquiera de sus emparejamientos), éstos son filtrados y no intervienen más en la reconstrucción[6].

Estos últimos cuatro bloques se repiten hasta que no quedan imágenes que cumplan las condiciones para ser incluidas en la reconstrucción, finalizando así las estimaciones. Un ejemplo de reconstrucción realizado por COLMAP se muestra en la Figura 2.9.

***Figura 2.9: Ejemplo de reconstrucción mediante COLMAP**. La línea roja representa la trayectoria estimada que siguió la cámara que capturó las fotos usadas en la reconstrucción 3D de la escena (3 de ellas se muestran en la parte inferior). Destacar como el software es capaz de reconstruir tanto la estructura global de la escena como pequeños detalles (p.ej. teclas).*

[6]Esto no es del todo cierto, ya que COLMAP aplica un proceso de *re-triangulación* [Wu, 2013] antes y después de aplicar BA global, con el objetivo de triangular puntos que quizás no han podido ser triangulados con éxito anteriormente por no disponer de unas poses de la cámara estabilizadas.

Capítulo 3

Formulación

Una vez introducidos los aspectos principales de la propuesta, en este capítulo se formalizan las herramientas matemáticas necesarias para cumplir los objetivos planteados. En primer lugar, se presenta el Grupo de Lie de $SE(3)$ (Espacio Euclídeo Especial) así como sus propiedades que atañen a nuestro problema, ya que éste representa el núcleo del bloque de optimización (Figura 2.1). A continuación se introducen los *B-Splines Cumulativos*, utilizados para interpolar los elementos de SE(3) para así modelar la trayectoria de los objetos en tiempo continuo.

3.1. Grupo de Lie, SE(3)

Una de las formas de localizar un objeto en el espacio consiste en definir un sistema de coordenadas (o de referencia) "pegado" a él, $\{o\}$, y expresar cuál es su orientación y posición con respecto a un sistema de referencia fijo (mundo), $\{w\}$. En la Figura 3.1 se muestra un esquema con lo que nos referimos. Los sistemas de referencia usados siguen la regla de la mano derecha y están formados por 3 ejes unitarios y ortogonales entre sí $\{\hat{x}, \hat{y}, \hat{z}\}$, por lo que:

$$\hat{x} \times \hat{y} = \hat{z}, \qquad \|\hat{x}\| = \|\hat{y}\| = \|\hat{z}\| = 1 \tag{3.1}$$

Donde $\|\cdot\|$, $\times$ representan la norma euclídea y el producto vectorial respectivamente.

Siguiendo la notación de la Fig. 3.1, $\mathbf{t}$, representa la localización del objeto (origen de su sistema de referencia). Expresándolo desde el sistema de referencia mundo:

$$\mathbf{t} = t_1\hat{x}_w + t_2\hat{y}_w + t_3\hat{z}_w \tag{3.2}$$

Ahora, para definir la orientación del objeto, expresamos los ejes de su sistema de coordenadas en términos del sistema de coordenadas fijo (mundo):

$$\hat{x}_o = r_{11}\hat{x}_w + r_{21}\hat{y}_w + r_{31}\hat{z}_w \tag{3.3}$$
$$\hat{y}_o = r_{12}\hat{x}_w + r_{22}\hat{y}_w + r_{32}\hat{z}_w \tag{3.4}$$
$$\hat{z}_o = r_{13}\hat{x}_w + r_{23}\hat{y}_w + r_{33}\hat{z}_w \tag{3.5}$$

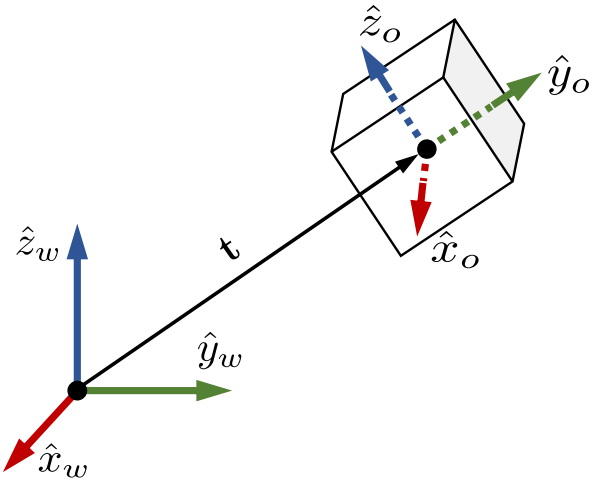


Figura 3.1: *Posición y orientación del sistema de referencia, $\{o\}$ de un objeto con respecto a un sistema de referencia fijo, $\{w\}$.*

De forma matricial:

$$\mathbf{t}_{wo} = \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix}, \qquad \mathbf{R}_{wo} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{3.6}$$

Donde se han usado los subíndices (wo) para especificar que se está expresando el sistema de referencia $\{o\}$ con respecto a $\{w\}$. De esta forma, $\mathbf{R}_{wo}$ es la matriz de rotación cuyas columnas son los ejes $\{\hat{x}_o, \hat{y}_o, \hat{z}_o\}$ expresados desde $\{w\}$[1], y $\mathbf{t}_{wo}$ es el vector de traslación que define la posición del origen de $\{o\}$ expresado desde $\{w\}$.

Sabiendo esto, podemos *transformar* un punto que inicialmente está definido en la referencia $\{o\}$, $\mathbf{p}_o \in \mathbb{R}^3$, a su equivalente en $\{w\}$ mediante:

$$\mathbf{p}_w = \mathbf{R}_{wo}\mathbf{p}_o + \mathbf{t}_{wo} \tag{3.7}$$

Ya que $\mathbf{R}_{wo}\mathbf{p}_o$ son las proyecciones del punto $\mathbf{p}$ en cada uno de los ejes del sistema de referencia mundo, a las que se les añade $\mathbf{t}_{wo}$ para tener en cuenta que los orígenes de ambas referencias no son coincidentes.

De manera equivalente, si expresamos $\mathbf{p}_o$ en coordenadas homogéneas: $\tilde{\mathbf{p}} = [\mathbf{p}^T, 1]^T$, este cambio de coordenadas viene dado por la *matriz de transformación* $\mathbf{T}_{wo}$:

$$\tilde{\mathbf{p}}_w = \mathbf{T}_{wo}\tilde{\mathbf{p}}_o, \qquad \mathbf{T}_{wo} = \begin{bmatrix} \mathbf{R}_{wo} & \mathbf{t}_{wo} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} \tag{3.8}$$

Algo que también se puede aplicar a otras matrices de transformación:

$$\mathbf{T}_{ac} = \mathbf{T}_{ab}\mathbf{T}_{bc} = \begin{bmatrix} \mathbf{R}_{ab} & \mathbf{t}_{ab} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R}_{bc} & \mathbf{t}_{bc} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{ab}\mathbf{R}_{bc} & \mathbf{R}_{ab}\mathbf{t}_{bc} + \mathbf{t}_{ab} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} \tag{3.9}$$

Para cualesquiera sistemas de referencia $\{a\}$, $\{b\}$ y $\{c\}$. Como se verá más adelante, otro de los usos de las matrices de transformación es el de aplicar una rotación seguida de una traslación a un vector o a un sistema de coordenadas.

[1]Equivalentemente, sus filas son los ejes $\{\hat{x}_w, \hat{y}_w, \hat{z}_w\}$ expresados en $\{o\}$.

A partir de las ecuaciones anteriores, vemos que para definir la localización de un objeto, se están empleando 12 parámetros (los 9 elementos de $\mathbf{R}_{wo}$ + los 3 elementos de $\mathbf{t}_{wo}$), sin embargo, los cuerpos rígidos tienen 6 grados de libertad [Lynch and Park, 2017] y no el doble, por lo que esta representación es redundante. Por ello, deben existir 6 restricciones independientes en esta representación.

Estas restricciones se encuentran en la matriz de rotación $\mathbf{R}$, ya que como se ha comentado antes, sus columnas (o filas) se corresponden con ejes de coordenadas, por lo que éstas deben ser de norma unidad y ortogonales entre sí, y por tanto cumplir:

$$\mathbf{R}^T\mathbf{R} = \mathbf{I}_{3\times3} \tag{3.10}$$

Donde $\mathbf{I}_{3\times3}$ representa la matriz identidad de 3×3 dimensiones. Estas restricciones implican que[2] $\det(\mathbf{R}) = \pm1$, pero al estar utilizando sistemas de referencia que siguen la regla de la mano derecha (Ec. 3.1), esto obliga que $\det(\mathbf{R}) = r_1^T(r_2 \times r_3) = r_1^T r_1 = +1$, donde r_i representa la columna $i \in \{1, 2, 3\}$ de $\mathbf{R}$.

Estas restricciones sirven para comprobar que las matrices de transformación, $\mathbf{T}$, con la multiplicación como operación interna, cumplen los axiomas de grupo. Es decir, considerando 3 matrices de transformación $\mathbf{T}_A, \mathbf{T}_B, \mathbf{T}_C$ cualesquiera (elementos del grupo), se cumplen [Sola et al., 2018, Lynch and Park, 2017]:

- ***Existe un elemento identidad*** *perteneciente al grupo,* $\mathbf{T}_I$ *tal que* $\mathbf{T}_A\mathbf{T}_I = \mathbf{T}_I\mathbf{T}_A = \mathbf{T}_A$. Se satisface, ya que $\mathbf{T}_I = \mathbf{I}_{4\times4}$ pertenece al grupo:

  $$\mathbf{T}_I = \mathbf{I}_{4\times4} \Rightarrow \mathbf{R}_I = \mathbf{I}_{3\times3} \Rightarrow \text{satisface} \begin{cases} \mathbf{R}_I^T\mathbf{R}_I = \mathbf{I}_{3\times3} \\ \det(\mathbf{R}_I) = 1 \end{cases}$$

  y además cumple este axioma, al ser una propiedad de la matrices identidad.

- ***Clausura****, o cierre:* $\mathbf{T}_A\mathbf{T}_B$ *pertenece al grupo.* Se cumple, pues:

  $$(\mathbf{R}_A\mathbf{R}_B)^T\mathbf{R}_A\mathbf{R}_B = \mathbf{R}_B^T(\mathbf{R}_A^T\mathbf{R}_A)\mathbf{R}_B = \mathbf{R}_B^T\mathbf{R}_B = \mathbf{I}_{3\times3}$$
  $$\det(\mathbf{R}_A\mathbf{R}_B) = \det(\mathbf{R}_A)\det(\mathbf{R}_B) = 1 \cdot 1 = 1$$

  y a partir de la Ec. 3.9, vemos que éstas son las condiciones suficientes al ser $\mathbf{R}_A\mathbf{R}_B$ la matriz de rotación resultante de la multiplicación de elementos del grupo.

- ***Existencia de un elemento inverso****,* $\mathbf{T}_A^{-1}$ *perteneciente al grupo, tal que* $\mathbf{T}_A^{-1}\mathbf{T}_A = \mathbf{T}_A\mathbf{T}_A^{-1} = \mathbf{T}_I$. Existe, y viene dado por:

  $$\mathbf{T}_A^{-1} = \begin{bmatrix} \mathbf{R}_A^T & -\mathbf{R}_A^T\mathbf{t}_A \\ \mathbf{0}_{1\times3} & 1 \end{bmatrix} \quad \rightarrow \quad \mathbf{T}_A^{-1}\mathbf{T}_A = \mathbf{T}_A\mathbf{T}_A^{-1} \overset{(3.9)}{=} \mathbf{I}_{4\times4}$$

  Pertenece al grupo, puesto que se satisface que $\mathbf{R}_A^T\mathbf{R}_A = \mathbf{I}_{3\times3}$ y $\det(\mathbf{R}_A^T) = \det(\mathbf{R}_A) = 1$.

- ***Asociatividad****:* $(\mathbf{T}_A\mathbf{T}_B)\mathbf{T}_C = \mathbf{T}_A(\mathbf{T}_B\mathbf{T}_C)$. Se cumple por las propias propiedades de la multiplicación de matrices.

[2] $\det(\cdot)$ representa el determinante de una matriz.

Este grupo es conocido como el **Grupo Euclídeo Especial** o $SE(3)$, siendo además un **Grupo de Lie** al no solo cumplir los anteriores axiomas, sino también ser una *variedad diferencial/ suave* (***smooth manifold***) [Sola et al., 2018].

Esto tiene gran importancia, ya que a pesar de que el espacio al que pertenecen las matrices de transformación no sea *globalmente* un espacio vectorial, sí que es posible aproximarlo *locamente* como tal, facilitando así la optimización de las estimaciones de $SE(3)$ a través de su plano tangente, pues al ser éste un espacio vectorial, podemos llevar a cabo cálculo de manera más directa en él [Sola et al., 2018, Strasdat, 2012].

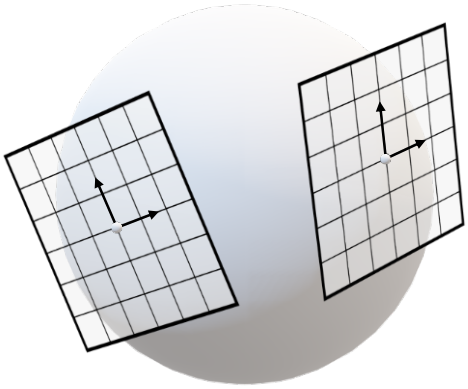


Figura 3.2: Ejemplo de variedad suave *(smooth manifold). Localmente se puede aproximar como un espacio vectorial (su plano tangente), pero globalmente es una estructura no Euclídea.*

Un ejemplo intuitivo de este tipo de variedad que se puede visualizar, es la de una esfera (Fig. 3.2). Localmente puede ser representada como un plano, en el que por ejemplo, nos podemos mover en direcciones perpendiculares (así como en la superficie de la Tierra), pero sin embargo su estructura global dista de serlo [Strasdat, 2012].

El (hiper)plano tangente de una matriz de transformación, $d\mathbf{T}(t)/dt = \dot{\mathbf{T}}$ lo podemos definir mediante la diferenciación de la condición que cumplen todos los elementos del grupo:

$$\mathbf{T}\mathbf{T}^{-1} = \mathbf{I} \quad \Rightarrow \quad \dot{\mathbf{T}}\mathbf{T}^{-1} + \mathbf{T}\dot{\mathbf{T}}^{-1} = \mathbf{0}_{4\times4} \tag{3.11}$$

$$\Rightarrow \quad \begin{bmatrix} \dot{\mathbf{R}} & \dot{\mathbf{t}} \\ \mathbf{0} & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R}^T & -\mathbf{R}^T\mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} + \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \dot{\mathbf{R}}^T & -\dot{\mathbf{R}}^T\mathbf{t} - \mathbf{R}^T\dot{\mathbf{t}} \\ \mathbf{0} & 0 \end{bmatrix} = \mathbf{0}_{4\times4} \tag{3.12}$$

Fijándonos en las matrices de rotación, se deriva la siguiente condición:

$$\dot{\mathbf{R}}\mathbf{R}^T + \mathbf{R}\dot{\mathbf{R}}^T = \mathbf{0}_{3\times3} \quad \Rightarrow \quad \dot{\mathbf{R}}\mathbf{R}^T = -(\dot{\mathbf{R}}\mathbf{R}^T)^T \tag{3.13}$$

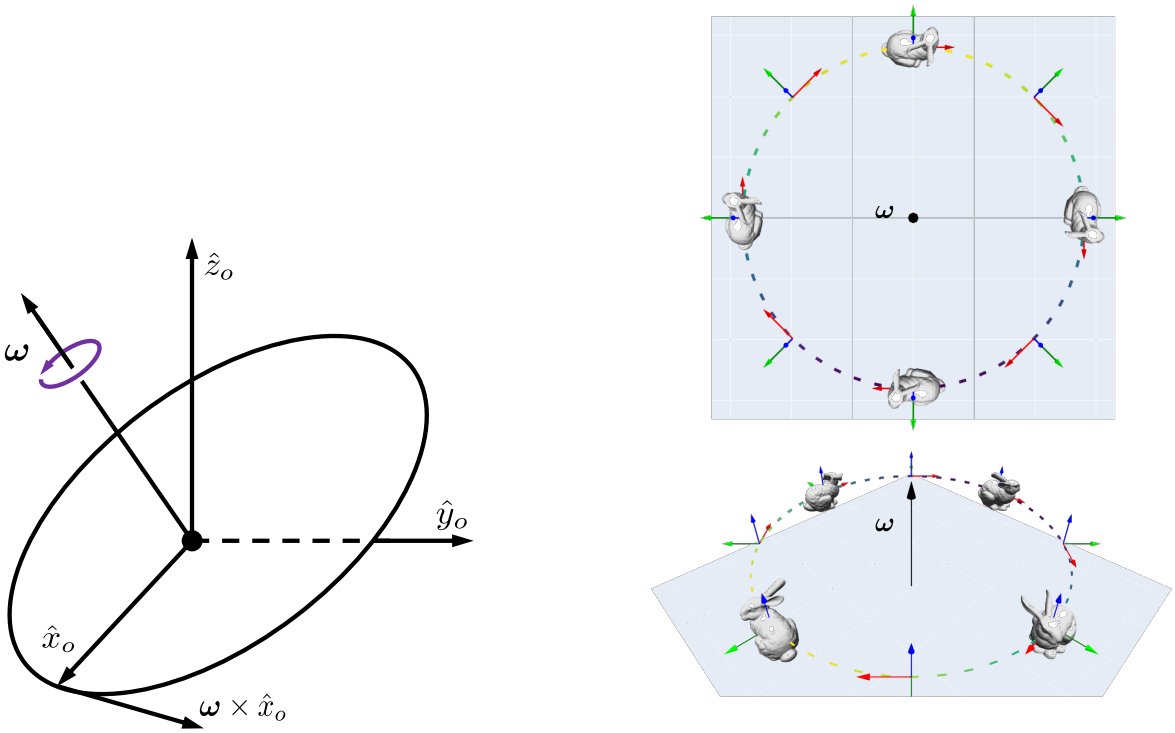
Es decir, $\dot{\mathbf{R}}\mathbf{R}^T$ es el negativo de su traspuesta, lo que implica que es una matriz antisimétrica, $\boldsymbol{\omega}^\wedge$:

$$\boldsymbol{\omega}^\wedge = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix}, \qquad \begin{cases} \dot{\mathbf{R}}\mathbf{R}^T & = \boldsymbol{\omega}^\wedge \\ \dot{\mathbf{R}} & = \boldsymbol{\omega}^\wedge\mathbf{R} \end{cases} \tag{3.14}$$

De esta forma los (hiper)planos tangentes a la variedad diferenciable $SE(3)$ cumplen esta condición. En la Ec. 3.14 se ha introducido además el operador "sombrero" $(\cdot)^\wedge$, para indicar que se está expresando el vector $\boldsymbol{\omega} = [\omega_1, \omega_2, \omega_3]^T$ como matriz antisimétrica.

3.1.1. Cinemática

El vector que acabamos de definir, $\boldsymbol{\omega}$, carga con un sentido físico relevante, pues representa la **velocidad angular** que experimenta el objeto, indicando de esta forma, el cambio de orientación (medida angular) por unidad de tiempo de los ejes del sistema de coordenadas sujeto a él.



(a) (b)

***Figura 3.3: Velocidad angular**. (a) Giro relativo que experimenta el eje $\hat{x}_o$ de un sistema de coordenadas al experimentar éste una velocidad angular $\boldsymbol{\omega}$. (b) Rotación de un objeto (y el sistema de coordenadas sujeto a él) por acción de $\boldsymbol{\omega}$.*

Para visualizar esto, consideremos un objeto moviéndose con una determinada velocidad angular, como en la Fig. 3.3b. Si elegimos como punto de referencia el origen de su sistema de coordenadas, la velocidad relativa del resto de puntos del objeto, se puede calcular por composición de movimientos:

$$\dot{\hat{p}} = \boldsymbol{\omega} \times \hat{p} \tag{3.15}$$

Donde $\hat{p}$ representa el vector que parte del origen (punto de referencia) y va hasta el punto, y $\dot{\hat{p}}$ por tanto representa la velocidad relativa lineal de la punta de este vector, debida únicamente al cambio de orientación que experimenta conforme pasa el tiempo[3].

[3] Al estar considerando un sólido rígido, la distancia entre los puntos que lo conforman es constante en el tiempo, por lo que solo puede variar la orientación del vector.

Aplicándolo a los ejes del sistema de coordenadas:

$$\dot{\hat{x}} = \boldsymbol{\omega} \times \hat{x} \tag{3.16}$$

$$\dot{\hat{y}} = \boldsymbol{\omega} \times \hat{y} \tag{3.17}$$

$$\dot{\hat{z}} = \boldsymbol{\omega} \times \hat{z} \tag{3.18}$$

Describiendo de manera relativa, una circunferencia como la mostrada en la Figura 3.3a[4].

Hasta ahora, no se ha considerado expresar la velocidad angular o los ejes en ningún sistema de referencia en concreto. Si elegimos el sistema de referencia mundo $\{w\}$, las direcciones de los ejes del sistema de coordenadas objeto, se corresponden con las columnas de $\mathbf{R}_{wo}$ (Ecs. 3.3-3.6). Por ello, $\dot{\mathbf{R}}_{wo}$ (variación temporal de los elementos de $\mathbf{R}_{wo}$), se puede expresar como:

$$\dot{\mathbf{R}}_{wo} = \begin{bmatrix} \boldsymbol{\omega}_w \times r_1 & \boldsymbol{\omega}_w \times r_2 & \boldsymbol{\omega}_w \times r_3 \end{bmatrix} = \boldsymbol{\omega}_w \times \mathbf{R}_{wo} = \boldsymbol{\omega}_w^{\wedge} \mathbf{R}_{wo} \tag{3.19}$$

Donde $\boldsymbol{\omega}_w$ es la velocidad angular expresada en la referencia $\{w\}$, y r_i representa la columna i de $\mathbf{R}_{wo}$. La última igualdad se debe a que un producto vectorial entre dos vectores, $\mathbf{a} \times \mathbf{b}$, es equivalente a $\mathbf{a}^{\wedge}\,\mathbf{b}$ (es decir, con el vector $\mathbf{a}$ expresado como matriz antisimétrica). Llegando así al mismo resultado que en la Ec. 3.14.

De igual forma, si queremos expresar la velocidad angular que experimenta el objeto en otro sistema de referencia, como en el suyo propio: $\boldsymbol{\omega}_o$, al ser un vector (y no un punto en el espacio) recurrimos a la matriz de rotación que relaciona ambos sistemas:

$$\boldsymbol{\omega}_o = \mathbf{R}_{wo}^T \boldsymbol{\omega}_w, \tag{3.20}$$

que en forma de matriz antisimétrica [Lynch and Park, 2017] viene dada por la Ec. 3.21:

$$\boldsymbol{\omega}_o^{\wedge} = \mathbf{R}_{wo}^T \dot{\mathbf{R}}_{wo} \tag{3.21}$$

Algo importante a aclarar es que $\boldsymbol{\omega}_o$ **no** es la velocidad angular relativa al sistema de coordenadas del objeto en movimiento, sino que representa la velocidad angular relativa a un sistema de coordenadas fijo que *instantáneamente* es coincidente con el sistema sujeto al objeto.

Por lo que, a modo de resumen, si $\mathbf{R}_{wo}$ define la orientación de los ejes del sistema de referencia sujeto al objeto con respecto a la referencia mundo, la velocidad angular que está experimentando el mismo, viene dada por:

$$\boldsymbol{\omega}_w^{\wedge} = \dot{\mathbf{R}}_{wo} \mathbf{R}_{wo}^T, \tag{3.22}$$

$$\boldsymbol{\omega}_o^{\wedge} = \mathbf{R}_{wo}^T \dot{\mathbf{R}}_{wo}, \tag{3.23}$$

en función de si queremos representarla en el sistema de referencia mundo, o en un sistema de referencia (también fijo), que es instantáneamente coincidente con el sujeto al objeto. La aceleración angular se puede obtener por tanto diferenciando las anteriores expresiones con la regla del producto.

[4] Dicha figura está inspirada en la Fig. 3.10 (p. 74) de [Lynch and Park, 2017].

Tras haber comprobado el significado físico de pre- y post-multiplicar $\dot{\mathbf{R}}_{wo}$ por $\mathbf{R}_{wo}^T$, nos podemos plantear lo mismo con $\dot{\mathbf{T}}_{wo}$. Las conclusiones son similares e involucran a la velocidad lineal que experimenta el objeto. En primer lugar, si pre-multiplicamos por $\mathbf{T}_{wo}^{-1}$:

$$\mathbf{T}_{wo}^{-1}\,\dot{\mathbf{T}}_{wo} = \begin{bmatrix} \mathbf{R}_{wo}^T & -\mathbf{R}_{wo}^T\,\mathbf{t}_{wo} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \dot{\mathbf{R}}_{wo} & \dot{\mathbf{t}}_{wo} \\ \mathbf{0} & 0 \end{bmatrix} \tag{3.24}$$

$$= \begin{bmatrix} \mathbf{R}_{wo}^T\dot{\mathbf{R}}_{wo} & \mathbf{R}_{wo}^T\dot{\mathbf{t}}_{wo} \\ \mathbf{0} & 0 \end{bmatrix} \tag{3.25}$$

$$= \begin{bmatrix} \boldsymbol{\omega}_o^\wedge & \mathbf{v}_o \\ \mathbf{0} & 0 \end{bmatrix} \tag{3.26}$$

Es decir, $\mathbf{T}_{wo}^{-1}\,\dot{\mathbf{T}}_{wo}$ contiene la velocidad angular del objeto expresada en el sistema de referencia estacionario e instantáneamente coincidente con su sistema de coordenadas ($\boldsymbol{\omega}_o^\wedge$), así como la velocidad lineal del mismo, $\dot{\mathbf{t}}$, expresada en dicho sistema: $\mathbf{R}_{wo}^T\dot{\mathbf{t}}_{wo} = \mathbf{v}_o$.

Ahora, analizando el caso de la post-multiplicación de $\dot{\mathbf{T}}_{wo}$ por $\mathbf{T}_{wo}^{-1}$,

$$\dot{\mathbf{T}}_{wo}\,\mathbf{T}_{wo}^{-1} = \begin{bmatrix} \dot{\mathbf{R}}_{wo} & \dot{\mathbf{t}}_{wo} \\ \mathbf{0} & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R}_{wo}^T & -\mathbf{R}_{wo}^T\,\mathbf{t}_{wo} \\ \mathbf{0} & 1 \end{bmatrix} \tag{3.27}$$

$$= \begin{bmatrix} \dot{\mathbf{R}}_{wo}\mathbf{R}_{wo}^T & \dot{\mathbf{t}}_{wo} - \dot{\mathbf{R}}_{wo}\mathbf{R}_{wo}^T\,\mathbf{t}_{wo} \\ \mathbf{0} & 0 \end{bmatrix} \tag{3.28}$$

$$= \begin{bmatrix} \boldsymbol{\omega}_w^\wedge & \mathbf{v}_w \\ \mathbf{0} & 0 \end{bmatrix} \tag{3.29}$$

nuevamente se obtiene la velocidad angular (expresada en el sistema de referencia mundo, $\boldsymbol{\omega}_w$). Sin embargo, no se obtiene la velocidad lineal del objeto expresada en dicho sistema, $\dot{\mathbf{t}}_{wo}$, sino que se obtiene $\mathbf{v}_w = \dot{\mathbf{t}}_{wo} - \boldsymbol{\omega}_w \times \mathbf{t}_{wo}$ (ya que $\dot{\mathbf{R}}_{wo}\mathbf{R}_{wo}^T = \boldsymbol{\omega}_w^\wedge$), representando por tanto, la velocidad que tendría un punto del objeto ubicado en el origen del sistema de referencia mundo $\{w\}$.

Por otro lado, al igual que con la velocidad angular, se usa el operador $(\cdot)^\wedge$ para pasar de su representación en $\mathbb{R}^3$ a su representación como matriz antisimétrica, podemos aplicarlo igualmente para pasar de una representación $\boldsymbol{\tau} = [\mathbf{v}, \boldsymbol{\omega}]^T \in \mathbb{R}^6$ a la estructura de las Ecs. 3.26, 3.29:

$$\boldsymbol{\tau}^\wedge = \begin{bmatrix} \mathbf{v} \\ \boldsymbol{\omega} \end{bmatrix}^\wedge = \begin{bmatrix} \boldsymbol{\omega}^\wedge & \mathbf{v} \\ \mathbf{0} & 0 \end{bmatrix} \in se(3), \tag{3.30}$$

Los elementos de esta forma pertenecen al **álgebra de Lie** de $SE(3)$, denominado $se(3)$, los cuales conforman el plano tangente a $SE(3)$ en la identidad (sustituyendo $\mathbf{T} = \mathbf{I}$ por ej. en la Ec. 3.26, obtenemos que $\dot{\mathbf{T}} = \boldsymbol{\tau}^\wedge$), los cuales se pueden definir localmente -en un punto cualquiera $\mathbf{T} \in SE(3)$-, o globalmente -en $\mathbf{I}$ [Sola et al., 2018]. Un ejemplo visual se muestra en la Fig. 3.4.

En este caso, para expresar $\boldsymbol{\tau}_w$ en la referencia del objeto, o $\boldsymbol{\tau}_o$ en la referencia global, no podemos recurrir a la propia matriz de transformación que los relaciona.

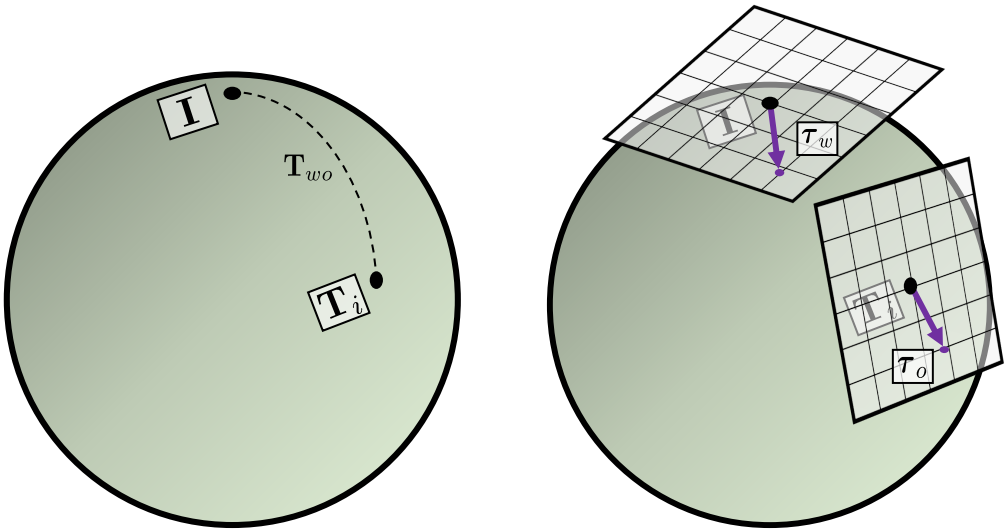


***Figura 3.4: Visualización simple planos tangentes en diferentes puntos de* SE(3).** *(a) Elemento identidad* $\mathbf{I}$ *y matriz de transformación* $\mathbf{T}_i \in SE(3)$ *(representada de manera simple como una esfera). (b) Álgebras de Lie, definidas en la identidad, con coordenadas* $\boldsymbol{\tau}_w$*, y en el punto tangente de* $\mathbf{T}_i$*, con coordenadas locales* $\boldsymbol{\tau}_o$*.*

Una evidencia clara de esto es que $\mathbf{T} \in SE(3)$ tiene dimensiones de 4×4, mientras que $\boldsymbol{\tau}$ es un vector $\in \mathbb{R}^6$. Es decir, necesitamos *a priori* una matriz de dimensiones 6×6.

Dicha matriz recibe el nombre de **matriz adjunta** de SE(3), y es la que define la transformación de $\boldsymbol{\tau}_o$ a $\boldsymbol{\tau}_w$; es decir, transforma los elementos del plano tangente definido localmente en $\mathbf{T} \in SE(3)$ al elemento del álgebra de Lie definido en la identidad $\mathbf{I}$ [Sola et al., 2018]. Es común representarla como $\mathrm{Ad}_{\mathbf{T}}$ [Lynch and Park, 2017, Sola et al., 2018], para así denotar que el punto de $SE(3)$ asociado a la transformación es $\mathbf{T}$:

$$\boldsymbol{\tau}_w = \begin{bmatrix} \mathbf{v}_w \\ \boldsymbol{\omega}_w \end{bmatrix} = \mathrm{Ad}_{\mathbf{T}_{wo}} \begin{bmatrix} \mathbf{v}_o \\ \boldsymbol{\omega}_o \end{bmatrix} = \mathrm{Ad}_{\mathbf{T}_{wo}} \boldsymbol{\tau}_o \tag{3.31}$$

La matriz adjunta de $SE(3)$ la podemos obtener sabiendo que se debe cumplir la Ec. 3.31, por lo que partiendo de las Ecs. 3.26 y 3.29:

$$\mathbf{v}_w = \mathbf{R}_{wo}\mathbf{v}_o + \mathbf{t}^{\wedge}\,\mathbf{R}_{wo} \tag{3.32}$$

$$\boldsymbol{\omega}_w = \mathbf{R}_{wo}\,\boldsymbol{\omega}_o \tag{3.33}$$

significa que $\mathrm{Ad}_{\mathbf{T}_{wo}}$ viene dada por:

$$\mathrm{Ad}_{\mathbf{T}_{wo}} = \begin{bmatrix} \mathbf{R}_{wo} & \mathbf{t}^{\wedge}\,\mathbf{R}_{wo} \\ \mathbf{0} & \mathbf{R}_{wo} \end{bmatrix} \tag{3.34}$$

A partir de esto, y recordando la Ec. 3.29, se cumple que: $\dot{\mathbf{T}}_{wo}\,\mathbf{T}_{wo}^{-1} = (\mathrm{Ad}_{\mathbf{T}_{wo}}\boldsymbol{\tau}_o)^{\wedge}$. En §3.1.2, con la introducción del mapeo exponencial y logarítmico, se presenta otro significado físico de $\mathrm{Ad}_{\mathbf{T}}$, que además es de utilidad en el bloque de optimización.

Antes de finalizar esta parte, conviene mencionar que también existe un operador, llamado *vee*, $(\cdot)^{\vee}$, utilizado para expresar un elemento del álgebra de Lie como un vector

$\boldsymbol{\tau} \in \mathbb{R}^6$:

$$\left(\begin{bmatrix} \boldsymbol{\omega}^\wedge & \mathbf{v} \\ \mathbf{0} & 0 \end{bmatrix}\right)^\vee = \begin{bmatrix} \mathbf{v} \\ \boldsymbol{\omega} \end{bmatrix} = \boldsymbol{\tau} \tag{3.35}$$

3.1.2. Mapeo exponencial y logarítmico

En la sección anterior se ha analizado cómo se traduce la velocidad lineal y angular que experimenta un objeto a los elementos, definidos de manera global y local, del álgebra de Lie $se(3)$. En esta sección se analiza el efecto que éstos tienen sobre el sistema de coordenadas (o matriz de transformación que la define), derivando así los **mapeos exponencial** y **logarítmico**, como forma de relacionar los elementos de estos planos tangentes a los elementos de la variedad y viceversa.

Para ello, partimos de la ecuación diferencial definida anteriormente:

$$\dot{\mathbf{T}}_{wo} = \boldsymbol{\tau}_w^\wedge \, \mathbf{T}_{wo}(t) \tag{3.36}$$

La solución a esta ecuación, que explica la evolución temporal de $\mathbf{T}_{wo}(t)$ bajo el efecto de las velocidades lineal y angular que definen a $\boldsymbol{\tau}_w^\wedge$, viene dada por (asumiendo como condición inicial $\mathbf{T}_{wo}(0) = \mathbf{T}_{wo}$) [Sola et al., 2018, Lynch and Park, 2017][5]:

$$\mathbf{T}_{wo}(t) = \exp(\boldsymbol{\tau}_w^\wedge \, t)\mathbf{T}_{wo} \tag{3.37}$$

Donde $\exp(\cdot)$ representa la *exponencial de una matriz*. Como $\mathbf{T}_{wo}(t) \in SE(3)$ $\forall t$, esto implica que $\exp(\boldsymbol{\tau}_w^\wedge \, t) \in SE(3)$ también. Es decir, nos permite transferir elementos del álgebra de Lie $(\boldsymbol{\tau}_w^\wedge \, t)$, definido en la identidad $\mathbf{I}$, a la variedad.

De esta forma, la velocidad, expresada en la referencia mundo $\{w\}$, que experimenta el sistema de coordenadas sujeto al objeto se traduce en un movimiento rígido (rotación y traslación) por medio del mapeo exponencial. Si en su lugar, las velocidades son expresadas en la referencia del objeto, llegamos a la misma conclusión:

$$\dot{\mathbf{T}}_{wo} = \mathbf{T}_{wo}(t) \, \boldsymbol{\tau}_o^\wedge \tag{3.38}$$

$$\mathbf{T}_{wo}(t) = \mathbf{T}_{wo} \exp(\boldsymbol{\tau}_o^\wedge \, t), \tag{3.39}$$

solo que en esta situación, el incremento $\exp(\boldsymbol{\tau}_o^\wedge \, t)$ está definido con respecto al sistema de coordenadas $\{o\}$, por lo que transferencia se realiza desde el plano tangente a $\mathbf{T}_{wo}$ a la variedad.

Intuitivamente [Sola et al., 2018], la exponencial de un elemento $\boldsymbol{\tau}^\wedge \in se(3)$, lo proyecta en la variedad siguiendo una geodésica (ver Fig. 3.5). Así mismo, para invertir dicha proyección -transferencia de $SE(3)$ a $se(3)$-, se utiliza el *logaritmo de una matriz* o **mapeo logarítmico** $\log(\cdot)$. Es decir:

$$\exp \quad : \quad se(3) \mapsto SE(3) \quad ; \quad \boldsymbol{\tau}^\wedge \mapsto \mathbf{T} = \exp(\boldsymbol{\tau}^\wedge) \tag{3.40}$$

$$\log \quad : \quad SE(3) \mapsto se(3) \quad ; \quad \mathbf{T} \mapsto \boldsymbol{\tau}^\wedge = \log(\mathbf{T}) \tag{3.41}$$

[5]Para comprobar que en efecto es la solución, podemos diferenciarla con respecto a t. Sabiendo que $\partial \exp(\boldsymbol{\tau}^\wedge \, t)/\partial t = \boldsymbol{\tau}^\wedge \exp(\boldsymbol{\tau}^\wedge \, t)$, se obtiene: $\boldsymbol{\tau}^\wedge \exp(\boldsymbol{\tau}^\wedge \, t)\mathbf{T}_{wo}$, y como $\mathbf{T}_{wo}(t) = \exp(\boldsymbol{\tau}^\wedge \, t)\mathbf{T}_{wo}$, significa que $\dot{\mathbf{T}}_{wo} = \boldsymbol{\tau}^\wedge \, \mathbf{T}_{wo}(t)$, coincidiendo así con la Ec. 3.36.

Por comodidad, para poder expresarlos con los elementos del álgebra de Lie en forma de vector $\boldsymbol{\tau} \in \mathbb{R}^6$, es habitual [Sola et al., 2018, Forster et al., 2016] definirse una versión alternativa de los anteriores mapeos, indicándolo con la *primera letra mayúscula*:

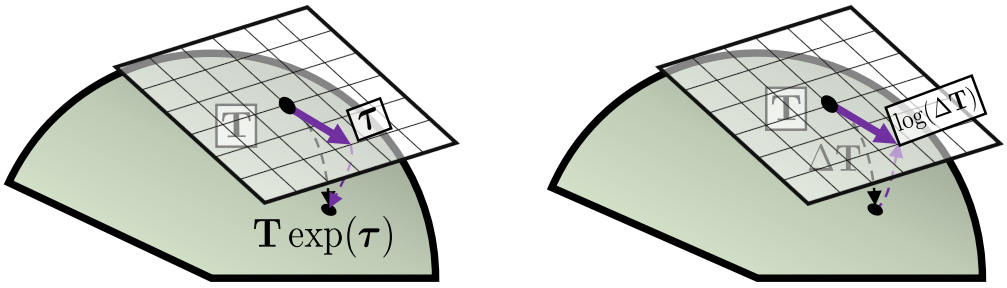
$$\text{Exp} \quad : \quad \mathbb{R}^6 \quad \mapsto SE(3) \quad ; \quad \boldsymbol{\tau} \mapsto \mathbf{T} = \text{Exp}(\boldsymbol{\tau}) \tag{3.42}$$

$$\text{Log} \quad : \quad SE(3) \mapsto \mathbb{R}^6 \quad ; \quad \mathbf{T} \mapsto \boldsymbol{\tau} = \text{Log}(\mathbf{T}) \tag{3.43}$$

Cumpliéndose por tanto:

$$\mathbf{T} = \text{Exp}(\boldsymbol{\tau}) = \exp(\boldsymbol{\tau}^\wedge) \tag{3.44}$$

$$\boldsymbol{\tau} = \text{Log}(\mathbf{T}) = (\log(\mathbf{T}))^\vee \tag{3.45}$$



(a) (b)

Figura 3.5: Mapeo exponencial y logarítmico. *(a) Un incremento* $\boldsymbol{\tau}^\wedge \in se(3)$ *definido en el álgebra de Lie local al punto* $\mathbf{T} \in SE(3)$ *se transfiere a la variedad por medio de* $\exp(\boldsymbol{\tau}^\wedge)$. *(b) Un incremento local* $\Delta\mathbf{T} \in SE(3)$ *se transfiere al plano tangente (álgebra de Lie) local por medio de* $\log(\Delta\mathbf{T})$.

En el caso de $SE(3)$ existen expresiones cerradas para ambos mapeos [Blanco, 2010, Barfoot, 2017], derivándose a partir del desarrollo en serie de potencias de ambas funciones. En el caso de la matriz exponencial:

$$\exp(\boldsymbol{\tau}^\wedge) = \exp\left(\begin{bmatrix}\mathbf{v}\\ \boldsymbol{\omega}\end{bmatrix}^\wedge\right) = \begin{bmatrix}\exp(\boldsymbol{\omega}^\wedge) & \mathbf{V}\mathbf{v}\\ \mathbf{0} & 1\end{bmatrix}, \tag{3.46}$$

$$\text{donde} \begin{cases}\exp(\boldsymbol{\omega}^\wedge) & = \mathbf{I}_3 + \frac{\sin\theta}{\theta}\boldsymbol{\omega}^\wedge + \frac{1-\cos\theta}{\theta^2}(\boldsymbol{\omega}^\wedge)^2\\ \mathbf{V} & = \mathbf{I}_3 + \frac{1-\cos\theta}{\theta^2}\boldsymbol{\omega}^\wedge + \frac{\theta-\sin\theta}{\theta^3}(\boldsymbol{\omega}^\wedge)^2\end{cases}, \text{ con } \theta = \|\boldsymbol{\omega}\| \tag{3.47}$$

Y del mapeo logarítmico[6]:

$$\log(\mathbf{T}) = \log\left(\begin{bmatrix}\mathbf{R} & \mathbf{t}\\ \mathbf{0} & 1\end{bmatrix}\right) = \begin{bmatrix}\log(\mathbf{R}) & \mathbf{V}^{-1}\mathbf{t}\\ \mathbf{0} & 0\end{bmatrix}, \tag{3.48}$$

$$\text{donde } \log(\mathbf{R}) = \frac{\theta}{2\sin\theta}(\mathbf{R} - \mathbf{R}^T), \text{ con } \theta = \arccos\left(\frac{tr(\mathbf{R})-1}{2}\right) \tag{3.49}$$

Juntando estas definiciones con la de la *matriz adjunta* (Ec. 3.31), significa que los incrementos en la variedad, $\exp(\boldsymbol{\tau})$, definidos a través de las álgebras de Lie local y global se relacionan por:

$$\mathbf{T}_{wo}\exp(\boldsymbol{\tau}_o) = \exp(\text{Ad}_{\mathbf{T}_{wo}}\boldsymbol{\tau}_o)\mathbf{T}_{wo} = \exp(\boldsymbol{\tau}_w)\mathbf{T}_{wo} \tag{3.50}$$

[6]La operación $tr(\mathbf{R})$ expresa la traza de la matriz $\mathbf{R}$ (la suma de los elementos de su diagonal).

Siendo además una propiedad algebraica útil, al poder expresar el mismo incremento "transfiriendo" su multiplicación del lado derecho, al izquierdo. De la misma forma, esta transferencia se puede invertir haciendo uso de la inversa de la matriz adjunta $(\mathrm{Ad}_{\mathbf{T}_{wo}})^{-1} = \mathrm{Ad}_{(\mathbf{T}_{wo})^{-1}}$ [Sola et al., 2018]:

$$\exp(\boldsymbol{\tau}_w)\mathbf{T}_{wo} = \mathbf{T}_{wo}\exp(\mathrm{Ad}_{(\mathbf{T}_{wo})^{-1}}\boldsymbol{\tau}_w) = \mathbf{T}_{wo}\exp(\boldsymbol{\tau}_o) \tag{3.51}$$

Otra propiedad de interés del mapeo exponencial de cara a la siguiente sección, es la derivada temporal de una matriz de transformación[7] cuyo elemento asociado en el álgebra de Lie varía de forma lineal en el tiempo, t [Strasdat, 2012]:

$$\frac{\partial}{\partial t}\exp(t\boldsymbol{\tau}^\wedge) = \boldsymbol{\tau}^\wedge\exp(t\boldsymbol{\tau}^\wedge) = \exp(t\boldsymbol{\tau}^\wedge)\boldsymbol{\tau}^\wedge \tag{3.52}$$

3.1.3. Jacobianos derecho e izquierdo de SE(3)

Antes de introducir cómo se lleva a cabo la optimización de los elementos $\in SE(3)$, conviene introducir el **jacobiano derecho** e **izquierdo** de $SE(3)$ debido al uso de los mismos en nuestra propuesta.

Estos jacobianos representan cómo afecta de forma infinitesimal una perturbación, $\delta\boldsymbol{\tau}$, en el elemento del álgebra de Lie $\boldsymbol{\tau} = \mathrm{Log}(\mathbf{T})$ con la perturbación $\mathrm{Exp}(\delta\boldsymbol{\phi})$ que ocasiona en la variedad[8]. Es decir, se definen como $\partial\delta\boldsymbol{\phi}/\partial\delta\boldsymbol{\tau}$, difiriendo la definición de cada jacobiano según el plano tangente sobre el que se representa la perturbación $\mathrm{Exp}(\delta\boldsymbol{\phi})$ [Sola et al., 2018, Sola, 2017b].

Para el jacobiano derecho, $\mathbf{J}_r(\boldsymbol{\tau})$, se considera la perturbación $\mathrm{Exp}(\delta\boldsymbol{\phi})$ en el plano tangente al punto $\mathbf{T}$, mientras que el jacobiano izquierdo considera el plano tangente a la identidad:

$$\mathrm{Exp}(\boldsymbol{\tau})\mathrm{Exp}(\delta\boldsymbol{\phi}) = \mathrm{Exp}(\boldsymbol{\tau}+\delta\boldsymbol{\tau}), \qquad \text{para } \mathbf{J}_r(\boldsymbol{\tau}), \tag{3.53}$$

$$\mathrm{Exp}(\delta\boldsymbol{\phi})\mathrm{Exp}(\boldsymbol{\tau}) = \mathrm{Exp}(\boldsymbol{\tau}+\delta\boldsymbol{\tau}), \qquad \text{para } \mathbf{J}_l(\boldsymbol{\tau}), \tag{3.54}$$

De esta forma, despejando $\delta\boldsymbol{\phi}$ de las anteriores ecuaciones para así poder representar su variación infinitesimal, se obtiene:

$$\mathbf{J}_r(\boldsymbol{\tau}) = \left.\frac{\mathrm{Log}(\mathrm{Exp}(\boldsymbol{\tau})^{-1}\mathrm{Exp}(\boldsymbol{\tau}+\delta\boldsymbol{\tau}))}{\delta\boldsymbol{\tau}}\right|_{\delta\boldsymbol{\tau}=\mathbf{0}} \tag{3.55}$$

$$\mathbf{J}_l(\boldsymbol{\tau}) = \left.\frac{\mathrm{Log}(\mathrm{Exp}(\boldsymbol{\tau}+\delta\boldsymbol{\tau})\mathrm{Exp}(\boldsymbol{\tau})^{-1})}{\delta\boldsymbol{\tau}}\right|_{\delta\boldsymbol{\tau}=\mathbf{0}} \tag{3.56}$$

Ambos jacobianos cuentan con soluciones cerradas [Barfoot, 2017]. Cumpliéndose para

[7] Esta propiedad no se restringe solo a elementos de $SE(3)$, la cumplen todos los grupo de Lie.

[8] Esta es la aplicación de los jacobianos derecho e izquierdo a la variedad de SE(3). Su aplicaciones se extienden de manera general al resto de grupos de Lie, que en este TFM no son presentados. La definición formal de estos jacobianos se encuentra en [Sola et al., 2018].

pequeñas perturbaciones de $\delta\boldsymbol{\tau}$ [Sola et al., 2018, Barfoot, 2017]:

$$\mathrm{Exp}(\boldsymbol{\tau}+\delta\boldsymbol{\tau}) \approx \mathrm{Exp}(\boldsymbol{\tau})\mathrm{Exp}(\mathbf{J}_r(\boldsymbol{\tau})\delta\boldsymbol{\tau}) \tag{3.57}$$

$$\mathrm{Exp}(\boldsymbol{\tau}+\delta\boldsymbol{\tau}) \approx \mathrm{Exp}(\mathbf{J}_l(\boldsymbol{\tau})\delta\boldsymbol{\tau})\mathrm{Exp}(\boldsymbol{\tau}) \tag{3.58}$$

$$\mathrm{Log}(\mathrm{Exp}(\boldsymbol{\tau})\mathrm{Exp}(\delta\boldsymbol{\tau})) \approx \boldsymbol{\tau} + \mathbf{J}_r^{-1}(\boldsymbol{\tau})\delta\boldsymbol{\tau} \tag{3.59}$$

$$\mathrm{Log}(\mathrm{Exp}(\delta\boldsymbol{\tau})\mathrm{Exp}(\boldsymbol{\tau})) \approx \boldsymbol{\tau} + \mathbf{J}_l^{-1}(\boldsymbol{\tau})\delta\boldsymbol{\tau} \tag{3.60}$$

3.2. B-Splines cumulativos en SE(3)

Uno de los objetivos de este TFM es el de estimar las trayectorias (constituidas por elementos de $SE(3)$) de los objetos a través de curvas continuas en el tiempo. Para este fin, el tipo de curva elegida es recomendable que cumpla las siguientes características [Haarbach et al., 2018, Patron-Perez et al., 2015]:

- **Control local** → Cambios locales en la curva no afectan globalmente a la trayectoria, permitiendo trabajar tanto de manera online como offline.
- **Continuidad $\mathcal{C}^2$** → Estimaciones de velocidad y aceleración continuas.
- **Libre de singularidades** → Cualquier movimiento puede ser interpolado.
- **Derivadas temporales analíticas** → Posibilidad de estimar la velocidad y aceleración en cualquier instante de tiempo.

Uno de los tipos de curvas más populares que se ajusta a estas necesidades son los **B-splines** [Kim et al., 1995, Haarbach et al., 2018]. En su definición base, cada punto de una curva B-Spline es una combinación lineal de k *funciones base B-Spline* $B_{i,k}(t)$, las cuales son polinomios de *grado* $k-1$, por lo que la continuidad resultante es de $\mathcal{C}^{k-2}$ [Kim et al., 1995]. Cada $B_{i,k}(t)$ tiene asociada un *punto de control* $\mathbf{p}_i \in \mathbb{R}^n$, ponderándolo así a lo largo del tiempo. La suma de todas las ponderaciones da lugar la curva B-Spline:

$$\mathbf{p}(t) = \sum_{i=0}^{n} \mathbf{p}_i B_{i,k}(t) \tag{3.61}$$

Donde n es el número de puntos de control y cada $B_{i,k}(t)$ se define según la fórmula recursiva de De Boor-Cox [De Boor, 1972, Cox, 1972]:

$$B_{i,1}(t) = \begin{cases} 1 & \text{si } t \in [t_i, t_{i+1}) \\ 0 & \text{en otro caso,} \end{cases} \tag{3.62}$$

$$B_{i,k}(t) = \frac{t-t_i}{t_{i+k-1}-t_i} B_{i,k-1}(t) + \frac{t_{i+k}-t}{t_{i+k}-t_{i+1}} B_{i+1,k-1}(t) \tag{3.63}$$

Los términos t_i se denominan *nudos*. A partir de las Ecs. 3.62-3.63 se infiere que cada $B_{i,k}(t)$ solamente es no nula cuando $t \in [t_i, t_{i+k})$.

En concreto, como estamos interesados en una curva que tenga continuidad $\mathcal{C}^2$, necesitamos $k=4$, es decir, funciones B-Spline con dependencia cúbica en el tiempo. Sustituyéndolo en las Ecs. 3.61-3.63 se deriva que para un instante $t \in [t_i, t_{i+1})$, el

valor de la curva $\mathbf{p}(t)$ está influida por: $[B_{i-3}(t), B_{i-2}(t), B_{i-1}(t), B_i(t)]$, siendo el resto nulas[9].

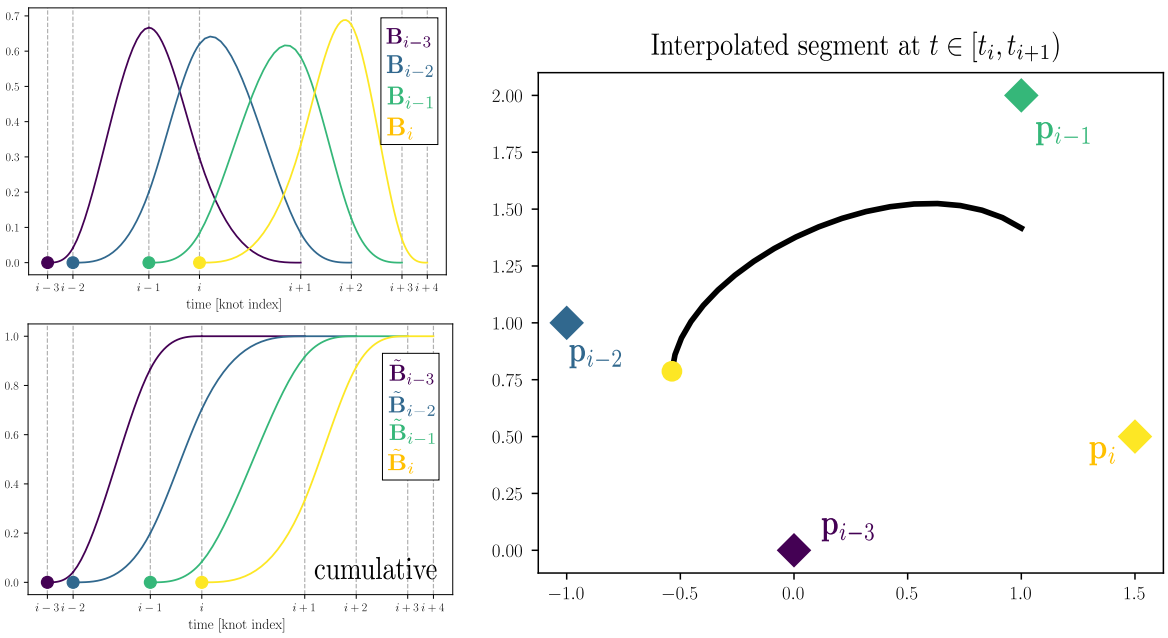
A partir del trabajo [Qin, 2000], el cálculo de estas funciones se puede realizar de forma matricial según la Ec. 3.64.

$$\mathbf{B} = \begin{bmatrix} B_{i-3}(t) & B_{i-2}(t) & B_{i-1}(t) & B_i(t) \end{bmatrix}^T = \mathbf{C}\mathbf{u} \tag{3.64}$$

Donde $\mathbf{u} = \begin{bmatrix} 1 & u & u^2 & u^3 \end{bmatrix}^T$, con $u = \frac{t-t_i}{t_{i+1}-t_i} \in [0, 1)$, y la matriz $\mathbf{C}$ (demostración en [Qin, 2000]), viene dada por:

$$\mathbf{C} = \begin{bmatrix} \frac{(t_{i+1}-t_i)^2}{(t_{i+1}-t_{i-1})(t_{i+1}-t_{i-2})} & -3c_{00} & 3c_{00} & -c_{00} \\ 1 - c_{00} - c_{20} & 3c_{00} - c_{21} & -3c_{00} - c_{22} & c_{00} - c_{23} - c_{33} \\ \frac{(t_i-t_{i-1})^2}{(t_{i+2}-t_{i-1})(t_{i+1}-t_{i-1})} & \frac{3(t_{i+1}-t_i)(t_i-t_{i-1})}{(t_{i+2}-t_{i-1})(t_{i+1}-t_{i-1})} & \frac{3(t_{i+1}-t_i)^2}{(t_{i+2}-t_{i-1})(t_{i+1}-t_{i-1})} & c_{23} \\ 0 & 0 & 0 & \frac{(t_{i+1}-t_i)^2}{(t_{i+3}-t_i)(t_{i+2}-t_i)} \end{bmatrix} \tag{3.65}$$

Donde los términos c_{ij} hacen referencia a términos ya definidos de la matriz en la fila i y columna j. Un ejemplo de interpolación en $\mathbb{R}^2$ se muestra en la Fig. 3.6.



(a) (b)

Figura 3.6: Muestra de funciones base B-Spline cúbicas e interpolación. *(a) Funciones originales (sup.) y su versión cumulativa (inf.), $(k = 4)$. (b) Interpolación resultante para 4 puntos de control $p_j \in \mathbb{R}^2, j \in \{i-3, \ldots, i\}$ para $t \in [t_i, t_{i+1})$. Los colores relacionan a cada función base con su punto de control asociado (en la versión cumulativa, la relación es con el punto de control cuyo signo no es invertido).*

Sin embargo, la formulación anterior no es directamente aplicable a $SE(3)$. Fijándonos en la Ec. 3.61, al sustituir los puntos de control $\mathbf{p}_i$ por matrices de transformación

[9] Para favorecer la claridad, se ha eliminado el valor de k de cada $B_{i,k}(t)$. De aquí en adelante, si no se indica, se asume que siempre es 4.

$\mathbf{T}_i$, el resultado no tiene porque pertenecer a $SE(3)$, ya sea por la operación suma, o por la multiplicación de un escalar $B_{i,k}$; es decir, en su forma base, no es posible interpolar matrices de transformación mediante una curva B-Spline.

Para hacer frente a esto, en [Kim et al., 1995], se propuso la versión **cumulativa**:

$$\mathbf{p}(t) = \mathbf{p}_0(t)\tilde{B}_{0,k} + \sum_{i=1}^{n}(\mathbf{p}_i - \mathbf{p}_{i-1})\tilde{B}_{i,k}(t) \tag{3.66}$$

que sí permite trabajar con grupos de Lie. Particularizando a $SE(3)$, el incremento entre dos puntos de control $\mathbf{T}_{i-1}$ y $\mathbf{T}_i$ expresado en el plano tangente de $\mathbf{T}_{i-1}$ viene dado por $\mathbf{\Omega}_i = \log(\mathbf{T}_{i-1}^{-1}\mathbf{T}_i)$. Es en este espacio Euclídeo en el que este incremento se pondera por la función base cumulativa. El último ingrediente para concatenar los incrementos es el mapeo exponencial visto en §3.1.2. Llegando finalmente a la versión equivalente en $SE(3)$ [Kim et al., 1995, Lovegrove et al., 2013]:

$$\mathbf{T}(t) = \exp(\tilde{B}_{0,k}\log(\mathbf{T}_0))\prod_{i=1}^{n}\exp(\tilde{B}_{i,k}(t)\mathbf{\Omega}_i) \tag{3.67}$$

En [Kim et al., 1995] se demostró que las funciones base B-Spline cumulativas $\tilde{B}_{i,k}(t)$ (asociadas al nudo t_i) se obtienen a través de la ecuación 3.68:

$$\tilde{B}_{i,k}(t) = \sum_{j=i}^{n} B_{j,k}(t) = \begin{cases} \sum_{j=i}^{i+k} B_{j,k}(t) & \text{si } t_i < t < t_{i+k-1} \\ 1 & \text{si } t \geq t_{i+k-1} \\ 0 & \text{si } t \leq t_i \end{cases} \tag{3.68}$$

Las condiciones anteriores para $k = 4$ se pueden observar de manera visual en la Fig. 3.6a.

Para trasladar esta versión a su forma matricial, con $k = 4$, (la que se utilizó en este trabajo), basta por tanto con sumar los elementos de la misma columna de $\mathbf{C}$ en la Ec. 3.65, ya que éstos son multiplicados por términos del polinomio temporal del mismo grado (por ej., los elementos de la última columna son multiplicados por u^3) y por tanto se les puede aplicar factor común. Es decir:

$$\tilde{\mathbf{B}} = \begin{bmatrix}\tilde{B}_{i-3}(t) & \tilde{B}_{i-2}(t) & \tilde{B}_{i-1}(t) & \tilde{B}_i(t)\end{bmatrix}^T = \tilde{\mathbf{C}}\mathbf{u} \tag{3.69}$$

$$\tilde{\mathbf{C}} = \begin{bmatrix} \sum_{f=0}^{3} c_{f0} & \sum_{f=0}^{3} c_{f1} & \sum_{f=0}^{3} c_{f2} & \sum_{f=0}^{3} c_{f3} \\ \sum_{f=1}^{3} c_{f0} & \sum_{f=1}^{3} c_{f1} & \sum_{f=1}^{3} c_{f2} & \sum_{f=1}^{3} c_{f3} \\ \sum_{f=2}^{3} c_{f0} & \sum_{f=2}^{3} c_{f1} & \sum_{f=2}^{3} c_{f2} & \sum_{f=2}^{3} c_{f3} \\ \sum_{f=3}^{3} c_{f0} & \sum_{f=3}^{3} c_{f1} & \sum_{f=3}^{3} c_{f2} & \sum_{f=3}^{3} c_{f3} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 - c_{00} & 3c_{00} & -3c_{00} & c_{00} \\ c_{20} & c_{21} & c_{22} & c_{23} + c_{33} \\ 0 & 0 & 0 & c_{33} \end{bmatrix} \tag{3.70}$$

Donde el subíndice f expresa la fila de la matriz $\mathbf{C}$. Los términos c_{ij} se corresponden con los de la Ec. 3.64.

Algo a destacar es que $\tilde{B}_{i-3} = 1,\ \forall t > t_i$. De esta observación se deriva que podemos expresar la Ec. 3.67 de forma equivalente para un instante $t \in [t_i, t_{i+1})$ partir de la Ec.

3.71:

$$\mathbf{T}(t) = \mathbf{T}_{i-3} \prod_{j=1}^{3} \exp(\tilde{B}_{i-3+j}(t)\mathbf{\Omega}_{i-3+j}) \tag{3.71}$$

Un ejemplo visual de interpolación en $SE(3)$ se muestra en la Fig. 3.7. Las funciones base cumulativas empleadas son las de la Fig. 3.6a.

Juntando lo anterior con el modelo de proyección de una cámara pinhole (Ec. 2.3), significa que un punto $\mathbf{p}_o$ (expresado en la referencia del objeto), es proyectado en un punto $\mathbf{x} \in \mathbb{R}^2$ de la imagen según la Ec. 3.72.

$$\mathbf{x} = \pi(\mathbf{T}_{wc}^{-1}\, \mathbf{T}_{wo}(t), \mathbf{p}_o) \tag{3.72}$$

Donde $\mathbf{T}_{wo}(t)$ es la matriz de transformación interpolada en el instante t (Ec. 3.71),

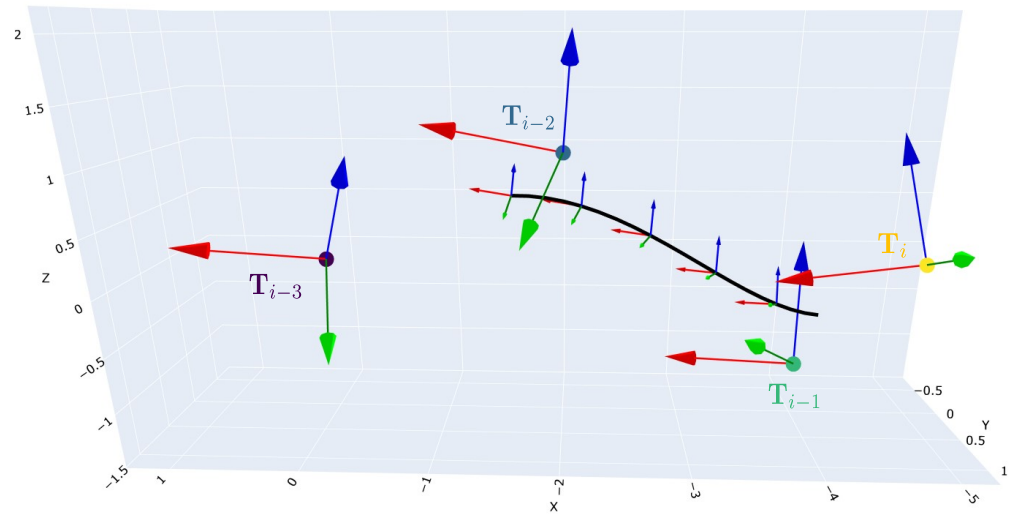


Figura 3.7: Ejemplo de interpolación en $SE(3)$. *Los 4 puntos de control,* $\mathbf{T}_j \in SE(3), j \in \{i-3, \ldots, i\}$, *son los sistemas de coordenadas de mayor tamaño, el resto son sistemas de coordenadas interpolados para un tiempo* $t \in [t_i, t_{i+1})$. *Las funciones base cumulativas empleadas son las de la Fig. 3.6a.*

que expresa el sistema de coordenadas del objeto con respecto a la referencia mundo $\{w\}$, y $\mathbf{T}_{wc}$ expresa la referencia de la cámara con respecto a $\{w\}$ en dicho instante (la cual es obtenida mediante COLMAP, §2.4).

3.2.1. Cinemática

Una de las ventajas de trabajar con trayectorias interpoladas con B-Splines es que éstas facilitan el cálculo de las derivadas temporales de una matriz de transformación $\mathbf{T}_{wo}(t)$ [Lovegrove et al., 2013]. Por ejemplo, si abstraemos cada incremento local $\exp\left(\tilde{B}_j(t)\mathbf{\Omega}_j\right)$ de la Ec. 3.71 como $\mathbf{A}_j$ con $j \in \{i-3, \ldots, i\}$:

$$\mathbf{T}_{wo}(t) = \mathbf{T}_{i-3} \exp\left(\tilde{B}_{i-2}(t)\mathbf{\Omega}_{i-2}\right) \exp\left(\tilde{B}_{i-1}(t)\mathbf{\Omega}_{i-1}\right) \exp\left(\tilde{B}_i(t)\mathbf{\Omega}_i\right) \tag{3.73}$$

$$\mathbf{T}_{wo}(t) = \mathbf{T}_{i-3}\mathbf{A}_{i-2}(t)\mathbf{A}_{i-1}(t)\mathbf{A}_i(t) \tag{3.74}$$

Significa que aplicando la regla del producto[10]:

$$\dot{\mathbf{T}}_{wo} = \mathbf{T}_{i-3}\left(\dot{\mathbf{A}}_{i-2}\mathbf{A}_{i-1}\mathbf{A}_i + \mathbf{A}_{i-2}\dot{\mathbf{A}}_{i-1}\mathbf{A}_i + \mathbf{A}_{i-2}\mathbf{A}_{i-1}\dot{\mathbf{A}}_i\right) \tag{3.75}$$

$$\ddot{\mathbf{T}}_{wo} = \mathbf{T}_{i-3}\begin{pmatrix}\ddot{\mathbf{A}}_{i-2}\mathbf{A}_{i-1}\mathbf{A}_i + \mathbf{A}_{i-2}\ddot{\mathbf{A}}_{i-1}\mathbf{A}_i + \mathbf{A}_{i-2}\mathbf{A}_{i-1}\ddot{\mathbf{A}}_i + \\ 2(\dot{\mathbf{A}}_{i-2}\dot{\mathbf{A}}_{i-1}\mathbf{A}_i + \mathbf{A}_{i-2}\dot{\mathbf{A}}_{i-1}\dot{\mathbf{A}}_i + \dot{\mathbf{A}}_{i-2}\mathbf{A}_{i-1}\dot{\mathbf{A}}_i)\end{pmatrix} \tag{3.76}$$

Cada término $\dot{\mathbf{A}}_j = \partial/\partial t(\exp(\tilde{B}_j(t)\boldsymbol{\Omega}))$ es calculado través de la propiedad 3.52 y la regla de la cadena:

$$\dot{\mathbf{A}}_j = \frac{\partial}{\partial t}\exp(\tilde{B}_j(t)\boldsymbol{\Omega}_j) = \frac{\partial \mathbf{A}_j}{\partial \tilde{B}_j}\frac{\partial \tilde{B}_j}{\partial t} \overset{3,52}{=} \boldsymbol{\Omega}\exp(\tilde{B}_j(t)\boldsymbol{\Omega})\frac{\partial \tilde{B}_j}{\partial t} \tag{3.77}$$

donde el término $\partial\tilde{B}_j/\partial t$ proviene de acceder con el índice correspondiente a la derivada temporal $\dot{\tilde{\mathbf{B}}}$, definida como:

$$\dot{\tilde{\mathbf{B}}} \overset{3,69}{=} \tilde{\mathbf{C}}\dot{\mathbf{u}}, \qquad \dot{\mathbf{u}} = \frac{1}{t_{j+1}-t_j}\begin{bmatrix}0 & 1 & 2u & 3u^2\end{bmatrix}^T \tag{3.78}$$

Finalmente, los términos $\ddot{\mathbf{A}}_j$ se obtienen diferenciando con respecto al tiempo cada $\dot{\mathbf{A}}_j$, y aplicando nuevamente la propiedad 3.52 y la regla del producto:

$$\ddot{\mathbf{A}}_j = \dot{\mathbf{A}}_j\boldsymbol{\Omega}_j\dot{\tilde{B}}_j(t) + \mathbf{A}_j\boldsymbol{\Omega}_j\ddot{\tilde{B}}_j(t), \quad \text{con} \tag{3.79}$$

$$\ddot{\tilde{\mathbf{B}}} = \tilde{\mathbf{C}}\ddot{\mathbf{u}}, \qquad \ddot{\mathbf{u}} = \frac{1}{(t_{j+1}-t_j)^2}\begin{bmatrix}0 & 0 & 2 & 6u\end{bmatrix}^T \tag{3.80}$$

De esta forma, gracias a los B-Splines somos capaces de conocer en cualquier instante de tiempo $\dot{\mathbf{T}}_{wo}, \ddot{\mathbf{T}}_{wo}$, y por tanto, la velocidad y aceleración lineal del objeto (a través de las derivadas temporales del vector de translación) y la velocidad y aceleración angular (a través de las derivadas temporales de la matriz de rotación y de la Ec. 3.22).

[10]Por mayor claridad la dependencia temporal de los términos no ha sido añadida.

Capítulo 4

Optimización

En este capítulo nos centramos en cómo nuestra propuesta optimiza las estimaciones de los puntos de control $\mathbf{T}_i \in SE(3)$ de la trayectoria del objeto en tiempo continuo (introducidos en §3.2), así como los puntos propios del objeto $\mathbf{p}_o \in \mathbb{R}^3$ en su referencia.

En la primera parte se formalizan los parámetros que son optimizados, así como la función de coste empleada. A continuación, se hace hincapié en el método de optimización elegido (Gauss-Newton), para finalmente explicar la derivación de los jacobianos necesarios en el cómputo de las actualizaciones de los parámetros que son optimizados.

4.1. Estado del sistema y funciones de coste: SplineBA, LocalBA

El conjunto de parámetros a optimizar se conoce formalmente como **vector de estado**, $\mathcal{X}$. En nuestro caso, las variables/ parámetros contenidos en dicho vector, son los puntos de control de las trayectorias de los objetos (§3.2), y los puntos de cada objeto que han sido seleccionados $\mathbf{p}_o$. Para aliviar la notación, y sin pérdida de generalidad (al no existir parámetros comunes entre objetos), se hace referencia a un único objeto o. Por ello, la formulación explicada en esta sección es igual e independiente para cada objeto.

Al conjunto de imágenes o *frames* usados para seguir a un objeto hasta un instante t lo denominamos como $\mathcal{I}_t = \{a, a+1, \ldots, n\}$, con $a, n \in \mathbb{N}$, representando respectivamente el número del primer y último frame de la secuencia que han sido empleados en la estimación de la trayectoria del objeto. Siguiendo esto, al conjunto de instantes temporales asociados a cada imagen lo representamos como $\mathcal{K}_t = \{t_a, \ldots, t_n\}$.

Nuestro sistema incluye un punto de control nuevo por cada nueva imagen o *frame* que recibe. La motivación de esto reside en ser capaces de estimar no solo movimientos suaves, sino también movimientos significativos ocurridos en un número de frames reducido. De esta decisión se deriva que en la Ec. 3.71 $t = t_i$ durante la optimización, por lo que el punto de control $\mathbf{T}_i$ no influye en ésta, reduciéndose así la carga computacional.

No incrementamos más el número de puntos de control dado que se perdería una significativa capacidad de interpolación entre frames. Por ej. si duplicásemos los puntos de control, por cada par de imágenes, habría uno que estaría influido únicamente por las observaciones de un frame, mientras que con nuestro planteamiento, cada punto de control está influenciado por información más rica, al provenir ésta de 3 imágenes.

A partir de este protocolo, y asumiendo un tiempo entre frames cuasi-constante, para un instante t_i el punto de control $\mathbf{T}_{i-2}$ es el que tiene una mayor influencia en la interpolación. Esto se observa en la Fig. 3.6a donde las funciones base asociadas al primer y segundo incremento reciben una mayor ponderación.

Por esta observación, en cada imagen recibida en el instante t_i inicializamos el vector de traslación del punto de control $\mathbf{T}_{i-2}$ al centro de masas de la nube de puntos segmentada del objeto. La matriz de rotación la inicializamos con la de $\mathbf{T}_{i-3}$, exceptuando al primer punto de control de todos, $\mathbf{T}_{a-2}$, que se inicializa con las direcciones principales [Géron, 2019] de la nube de puntos del objeto. De esta forma, el conjunto de puntos de control de la trayectoria viene dado por $\mathcal{T}_t = \{\mathbf{T}_{a-2}, \ldots, \mathbf{T}_{n-2}\}$.

Por otro lado, por cada nueva imagen recibida, se extraen un conjunto de *observaciones* $\mathcal{P}_z = \{\mathcal{P}_a, \ldots, \mathcal{P}_n\}$. Cada subconjunto $\mathcal{P}_k$ (extraído en un frame k) esta constituido por la localización 3D en la referencia de la cámara de cada uno de los puntos, $\mathbf{p}_c$, del objeto que se han extraído/ seguido en dicho frame.

Un mismo punto del objeto $\mathbf{p}_o$ puede ser seguido en imágenes consecutivas, dando lugar a distintas observaciones $\mathbf{p}_c$. Solo aquellos $\mathbf{p}_o$ de los que se disponga más de una observación $\mathbf{p}_c$ son considerados como parámetros a optimizar. La motivación de esto reside en utilizar puntos que han superado el filtrado de espúreos (explicado en §5) y que por tanto consideramos como estables.

Además, al compartirse estos puntos entre frames, se genera una mayor correlación con los puntos de control, aportando así información valiosa de cara a la interpolación. A este subconjunto lo denominamos $\mathcal{P}_t = \{\mathbf{p}_{o_0}, \ldots, \mathbf{p}_{o_{m-1}}\}$, con m el número de puntos en la referencia objeto que se han considerado para optimizar.

Para impedir que el coste computacional crezca de forma indefinida, la optimización no siempre se lleva a cabo con la totalidad de $\mathcal{T}_t$ y $\mathcal{P}_t$. En su lugar, consideramos una ventana temporal en la que los parámetros contenidos en ella son los optimizados. En concreto, en nuestros experimentos dicha ventana se corresponde con las últimas 20 imágenes. A este subconjunto de observaciones lo denominamos $\mathcal{P}_{zo} \subseteq \mathcal{P}_z$, y de él se derivan los subconjuntos de parámetros a optimizar: $\mathcal{T}_{to} \subseteq \mathcal{T}_t$ y $\mathcal{P}_{to} \subseteq \mathcal{P}_t$.

El refinamiento de las estimaciones se realiza a través de la minimización de una versión *robusta* del *error cuadrático*, $E(\mathcal{X})$, en la estimación de la *localización 3D de los puntos*:

$$E(\mathcal{X}) = \frac{1}{2} \sum_{\mathbf{p}_c \in \mathcal{P}_{zo}} \rho \left(\| \mathbf{p}_c - \text{proj} \left(\mathbf{T}_{wc}^{-1} \, \mathbf{T}_{wo}(t) \, \tilde{\mathbf{p}}_o \right) \|^2_{\Sigma_{\mathbf{p}_c}^{-1}} \right) \tag{4.1}$$

Donde $\mathbf{T}_{wc}$, $\mathbf{p}_o$ y $t \in \mathcal{K}_t$ son respectivamente la matriz de transformación de la ref. cámara a la ref. mundo, el punto del objeto y el instante temporal asociados a la observación $\mathbf{p}_c$. El significado de la matriz $\Sigma_{\mathbf{p}_c}^{-1}$ se presenta en §4.1.1. $\rho : \mathbb{R} \to \mathbb{R}$, es la

función robusta de Huber [Huber, 2004], explicada en §4.2.1. Por último, proj : $\mathbb{P}^3 \rightarrow \mathbb{R}^3$ transforma un punto de coordenadas homogéneas a cartesianas.

A modo de abstracción, para representar el error (o *residuo*) en cada observación, definimos:

$$\mathbf{r}_{\mathbf{p}_c} = \mathbf{p}_c - \text{proj}\left(\mathbf{T}_{wc}^{-1}\,\mathbf{T}_{wo}(t)\,\tilde{\mathbf{p}}_o\right) \tag{4.2}$$

Por lo que:

$$E(\mathcal{X}) = \frac{1}{2}\sum_{\mathbf{p}_c \in \mathcal{P}_{zo}} \rho\left(\|\mathbf{r}_{\mathbf{p}_c}\|^2_{\Sigma^{-1}_{\mathbf{p}_c}}\right) = \frac{1}{2}\sum_{\mathbf{p}_c \in \mathcal{P}_{zo}} \rho\left(\mathbf{r}^T_{\mathbf{p}_c}\Sigma^{-1}_{\mathbf{p}_c}\mathbf{r}_{\mathbf{p}_c}\right) \tag{4.3}$$

Haremos referencia a **spline BA** si la Ec. 4.1 es usada para optimizar únicamente los puntos de control ($\mathcal{X} = \mathcal{T}_{to}$) y **local BA** cuando se optimizan tanto puntos de control como puntos del objeto ($\mathcal{X} = \{\mathcal{T}_{to}, \mathcal{P}_{to}\}$). Los nombres provienen de la técnica *Bundle Adjustment* (*BA*) [Triggs et al., 1999], utilizada para optimizar variables de estado minimizando el error cuadrático de reproyección de las observaciones.

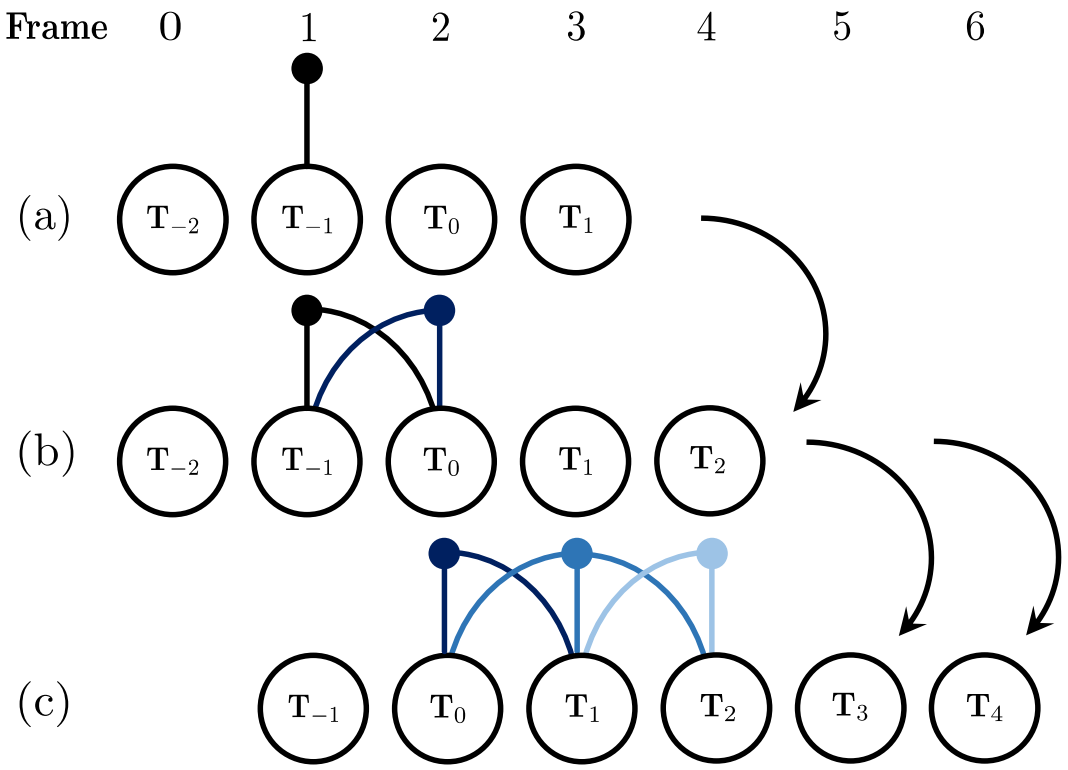


***Figura 4.1: Grafo simple asociado a spline BA**, en el que se considera una observación nueva por cada frame (● - ●) y una ventana temporal de 6 frames. (a) Se necesitan 4 puntos de control para comenzar. Con la primera observación (frame 1), se optimiza $\mathbf{T}_{-1}$ por tener mayor influencia. (b) Al añadir más observaciones, se optimizan más puntos de control. (c) Solo se optimizan los puntos de control situados en la ventana temporal.*

Con el objetivo de clarificar la relación entre los parámetros a optimizar y las observaciones, en las Figs. 4.1 y 4.2 se muestran los esquemas asociados a las optimizaciones *spline BA* y *local BA* respectivamente. A modo de simplificación, en ambas, se considera una ventana temporal de 6 imágenes y una única observación por frame. Formalmente

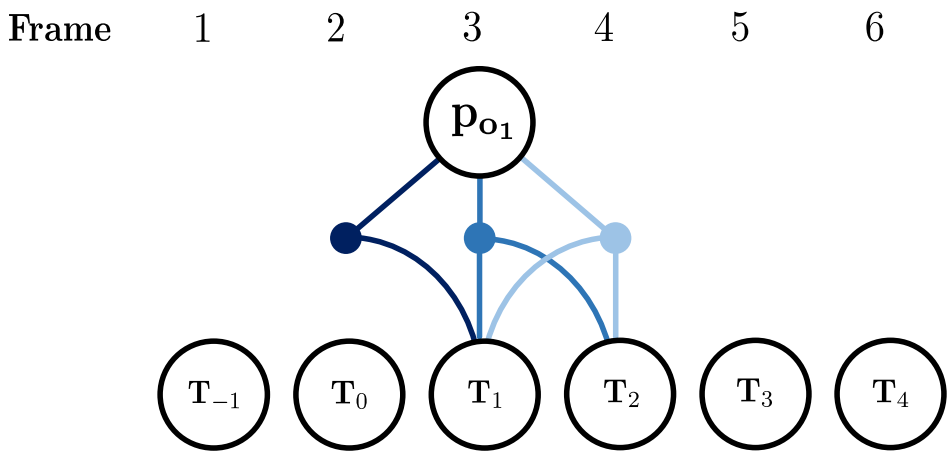


***Figura 4.2: Grafo simple asociado a local BA**, en el que se considera una observación nueva por cada frame (● - ●), una ventana temporal de 6 frames y un único punto del objeto a optimizad $\mathbf{p}_{o_1}$. El punto de control más antiguo es fijado para evitar grados de libertad en la optimización [Triggs et al., 1999], por ello no presenta relaciones con las observaciones.*

este tipo de gráficas se denominan como *grafos de factores*[1] [Dellaert et al., 2017].

En el caso de *spline BA*, ésta es utilizada en el inicio de la trayectoria y en cualquier momento en el que no se dispone de observaciones suficientes en $\mathcal{P}_{zo}$ como para optimizar la localización de los puntos del objeto. Para su ejecución, es necesario haber inicializado al menos 4 puntos control (Ec. 3.71).

Además, por cada conjunto de observaciones de un frame, solo se optimiza un nuevo punto de control, si no, la optimización presentaría *gauge freedom* (grados de libertad) al estar empleando más de una matriz de transformación para modelar una sola (la del objeto) [Triggs et al., 1999, Strasdat, 2012].

Es por esto que en la Fig. 4.1 solo existe un "eje" uniendo la observación inicial. Con el aumento de observaciones de distintos frames, es posible aumentar la cantidad de estas relaciones. En el caso de local BA, para evitar de nuevo grados de libertad en la optimización conjunta de $\mathcal{T}_{to}$ y $\mathcal{P}_{to}$, se fija el punto de control más antiguo, por lo que éste no presenta relaciones con las observaciones (Fig. 4.2).

4.1.1. Punto de vista probabilístico

Al problema de optimización anterior le podemos dar un enfoque probabilístico derivado de que las observaciones consideradas para optimizar, $\mathcal{P}_{zo}$, no son perfectas y presentan incertidumbre. Esto es debido a que éstas pueden presentar cierto ruido que ocasione, en nuestro caso, que una medición $\mathbf{p}_c$ no se corresponda exactamente con el punto $\mathbf{p}_o$ al que estamos asociándolo.

Por ello, no podemos pretender obtener con absoluta precisión el valor verdadero del conjunto de variables de estado $\mathcal{X}$ de interés, pero sí una estimación de las mismas

[1] Un estudio de los grafos de factores queda fuera del alcance de este TFM. En caso de que el lector esté interesado en los mismos, se recomienda el trabajo realizado en [Dellaert et al., 2017].

que se ajuste a un modelo estadístico elegido por nosotros [Dellaert et al., 2017]. Es decir, una estimación que maximize la probabilidad de obtenerse dado el conjunto de observaciones del que disponemos:

$$\mathcal{X}^{MAP} = \arg\max p(\mathcal{X} \mid \mathcal{P}_{zo}) \tag{4.4}$$

La estimación de $\mathcal{X}$ que maximiza dicha probabilidad se denomina como estimación de *Maximum A Posteriori* o MAP. Aplicando la regla de Bayes, obtenemos una expresión equivalente que facilita su derivación:

$$\mathcal{X}^{MAP} = \arg\max \frac{p(\mathcal{P}_{zo} \mid \mathcal{X})p(\mathcal{X})}{p(\mathcal{P}_{zo})}, \tag{4.5}$$

donde el término $p(\mathcal{P}_{zo})$, al no depender de $\mathcal{X}$, no lo podemos aprovechar para maximizar la Ec. 4.4 y por tanto lo podemos ignorar. Lo contrario ocurre con $p(\mathcal{P}_{zo} \mid \mathcal{X})$, conocido como *verosimilitud*, y que expresa la probabilidad de las observaciones dado el estado (también expresado como $\mathcal{L}(\mathcal{X} \mid \mathcal{P}_{zo})$), y con el término $p(\mathcal{X})$, conocido como *prior* e indica la probabilidad del estado dado un modelo o asunción del mismo.

En este trabajo, no consideramos ningún *prior* acerca de $\mathcal{X}$, o dicho de otra forma, asumimos que todos los estados presentan igual probabilidad, por lo que $p(\mathcal{X})$ es una constante que no influye en la maximización. Este acercamiento es común en la literatura [Mur-Artal et al., 2015, Mueggler et al., 2018, Triggs et al., 1999]. Por ello. *en nuestro caso*, la estimación MAP coincide con la de máxima verosimilitud (MLE):

$$\mathcal{X}^{MAP} = \mathcal{X}^{MLE} = \arg\max p(\mathcal{P}_{zo} \mid \mathcal{X}) = \arg\max \mathcal{L}(\mathcal{X} \mid \mathcal{P}_{zo}) \tag{4.6}$$

Otras asunciones extendidas en la literatura [Dellaert et al., 2017, Triggs et al., 1999], consisten en asumir independencia en las observaciones, y que cada observación $\mathbf{p}_c$ está perturbada por un ruido Gaussiano de media nula, $\mathcal{N}(\mathbf{0}, \Sigma_{\mathbf{p}_c})$, con $\Sigma_{\mathbf{p}_c}$ su matriz de covarianza. De esta forma, en la Ec. 4.2, $\mathbf{r}_{\mathbf{p}_c} \sim \mathcal{N}(\mathbf{0}, \Sigma_{\mathbf{p}_c})$, y además se cumple:

$$\mathcal{X}^{MLE} = \arg\max \prod_{\mathbf{p}_c \in \mathcal{P}_{zo}} p(\mathbf{p}_c \mid \mathcal{X}) \tag{4.7}$$

$$= \arg\max \prod_{\mathbf{p}_c \in \mathcal{P}_{zo}} \frac{1}{(2\pi)^{3/2}|\Sigma_{\mathbf{p}_c}|^{1/2}} \exp\left(-\frac{1}{2}\mathbf{r}_{\mathbf{p}_c}^T \Sigma_{\mathbf{p}_c}^{-1} \mathbf{r}_{\mathbf{p}_c}\right) \tag{4.8}$$

$$= \arg\max \log \prod_{\mathbf{p}_c \in \mathcal{P}_{zo}} \frac{1}{(2\pi)^{3/2}|\Sigma_{\mathbf{p}_c}|^{1/2}} \exp\left(-\frac{1}{2}\mathbf{r}_{\mathbf{p}_c}^T \Sigma_{\mathbf{p}_c}^{-1} \mathbf{r}_{\mathbf{p}_c}\right) \tag{4.9}$$

$$= \arg\max \sum_{\mathbf{p}_c \in \mathcal{P}_{zo}} \log\left[\frac{1}{(2\pi)^{3/2}|\Sigma_{\mathbf{p}_c}|^{1/2}} \exp\left(-\frac{1}{2}\mathbf{r}_{\mathbf{p}_c}^T \Sigma_{\mathbf{p}_c}^{-1} \mathbf{r}_{\mathbf{p}_c}\right)\right] \tag{4.10}$$

$$= \arg\max \sum_{\mathbf{p}_c \in \mathcal{P}_{zo}} \left(K_{\mathbf{p}_c} - \frac{1}{2}\mathbf{r}_{\mathbf{p}_c}^T \Sigma_{\mathbf{p}_c}^{-1} \mathbf{r}_{\mathbf{p}_c}\right), \quad K_{\mathbf{p}_c} = \log\left(\frac{1}{(2\pi)^{3/2}|\Sigma_{\mathbf{p}_c}|^{1/2}}\right) \tag{4.11}$$

$$= \arg\min \frac{1}{2} \sum_{\mathbf{p}_c \in \mathcal{P}_{zo}} \mathbf{r}_{\mathbf{p}_c}^T \Sigma_{\mathbf{p}_c}^{-1} \mathbf{r}_{\mathbf{p}_c} \tag{4.12}$$

$$= \arg\min \frac{1}{2} \sum_{\mathbf{p}_c \in \mathcal{P}_{zo}} \|\mathbf{r}_{\mathbf{p}_c}\|^2_{\Sigma_{\mathbf{p}_c}^{-1}} \tag{4.13}$$

Por lo que hemos llegado a la conclusión de que minimizar el error cuadrático en la estimación de la localización 3D de los puntos, es equivalente a maximizar la verosimilitud de la distribución de las observaciones si asumimos que el ruido al que están sometidas sigue una distribución $\mathcal{N}(\mathbf{0}, \Sigma_{\mathbf{p}_c})$.

4.2. Optimización mediante Gauss-Newton en SE(3)

Para llevar a cabo la minimización de $E(\mathcal{X})$, en este trabajo se utiliza el método de optimización de **mínimos cuadrados no lineales de Gauss-Newton**. Más concretamente, se emplea su versión *robustificada* mediante la función robusta de Huber [Triggs et al., 1999, Huber, 2004], expresada mediante $\rho(\cdot)$ en la Ec. 4.3.

En la explicación de esta parte, se asume que $\mathcal{X}$ está expresado en su forma *vectorizada*, es decir, con todos sus elementos apilados en un único vector $\mathbf{x}$. Además para facilitar la explicación, inicialmente se considera que a $E(\mathbf{x})$ no se le aplica $\rho(\cdot)$. Su contribución se añade más adelante. Juntando estas consideraciones, la Ec. 4.3 se re-expresa de la siguiente forma:

$$E(\mathbf{x}) = \frac{1}{2} \sum_{\mathbf{p}_c \in \mathcal{P}_{zo}} \mathbf{r}_{\mathbf{p}_c}^T \Sigma_{\mathbf{p}_c}^{-1} \mathbf{r}_{\mathbf{p}_c} \tag{4.14}$$

Al utilizar este método de optimización, estamos asumiendo que el error $\mathbf{r}_{\mathbf{p}_c}$ se comporta *localmente* de manera lineal con respecto a $\mathbf{x}$, por lo que $E(\mathbf{x})$ (Ec. 4.14) lo hace de forma cuadrática. Por tanto, localmente, podemos aproximarla por su desarrollo de Taylor de segundo orden [Triggs et al., 1999, Kümmerle et al., 2011]:

$$E(\mathbf{x} + \delta\mathbf{x}) \approx E(\mathbf{x}) + \frac{\partial E(\mathbf{x})}{\partial \mathbf{x}} \delta\mathbf{x} + \frac{1}{2} \delta\mathbf{x}^T \frac{\partial^2 E(\mathbf{x})}{\partial \mathbf{x}^2} \delta\mathbf{x} \tag{4.15}$$

$$= E(\mathbf{x}) + \mathbf{g}\delta\mathbf{x} \qquad + \frac{1}{2} \delta\mathbf{x}^T \mathbf{H} \delta\mathbf{x} \tag{4.16}$$

Partiendo de la Ec. 4.14, se derivan su *gradiente* $\mathbf{g}$ y *matriz Hessiana* $\mathbf{H}$:

$$\mathbf{g} = \frac{\partial E(\mathbf{x})}{\partial \mathbf{x}} = \sum_{\mathbf{p}_c \in \mathcal{P}_{zo}} \mathbf{r}_{\mathbf{p}_c}^T \Sigma_{\mathbf{p}_c}^{-1} \frac{\partial \mathbf{r}_{\mathbf{p}_c}}{\partial \mathbf{x}} = \sum_{\mathbf{p}_c \in \mathcal{P}_{zo}} \mathbf{r}_{\mathbf{p}_c}^T \Sigma_{\mathbf{p}_c}^{-1} \mathbf{J}_{\mathbf{p}_c}, \tag{4.17}$$

$$\mathbf{H} = \frac{\partial^2 E(\mathbf{x})}{\partial \mathbf{x}^2} \approx \sum_{\mathbf{p}_c \in \mathcal{P}_{zo}} \mathbf{J}_{\mathbf{p}_c}^T \Sigma_{\mathbf{p}_c}^{-1} \mathbf{J}_{\mathbf{p}_c} \tag{4.18}$$

Donde $\mathbf{J}_{\mathbf{p}_c} = \partial \mathbf{r}_{\mathbf{p}_c} / \partial \mathbf{x}$ se conoce como la **matriz Jacobiana** de $\mathbf{r}_{\mathbf{p}_c}$ evaluada en $\mathbf{x}$. La matriz Hessiana es una aproximación (propia del método de Gauss-Newton), ya que se están ignorando los términos de segundo orden $\partial^2 \mathbf{r}_{\mathbf{p}_c} / \partial \mathbf{x}^2$ al considerar un comportamiento *localmente* lineal.

Como nuestro objetivo es minimizar $E(\mathbf{x} + \delta\mathbf{x})$ a través del incremento de las variables de estado $\delta\mathbf{x}$, diferenciamos la Ec. 4.16 respecto a $\delta\mathbf{x}$, e igualamos a 0 para obtener los puntos críticos de *esta aproximación*:

$$\mathbf{g} + \mathbf{H}\delta\mathbf{x} = \mathbf{0} \quad \Rightarrow \quad \delta\mathbf{x} = -\mathbf{H}^{-1}\mathbf{g} \tag{4.19}$$

Repetir de manera iterativa la actualización del vector de estado según la Ec. 4.19 da lugar al método de optimización de Gauss-Newton. En nuestra implementación, en vez de invertir directamente $\mathbf{H}$, y aprovechando que es definida positiva, resolvemos el sistema de ecuaciones mediante su descomposición de Cholesky[2] [Golub and Van Loan,].

Algo a destacar, es que, salvo en situaciones donde las matrices Jacobianas no sean de rango completo, o la aproximación de $\mathbf{H}$ no sea buena (por ejemplo cuando valor actual de $\mathbf{x}$ es próximo al de un punto de silla) [Triggs et al., 1999, Sola, 2017a], $\delta\mathbf{x}$ es una dirección descendente de la función $E(\mathbf{x})$, es decir, una en la que se decrementa su valor, lo que resulta una característica atractiva de este método.

La justificación reside en que la aproximación de $\mathbf{H}$ de la Ec. 4.18 hace que ésta sea una matriz *definida positiva*, por lo que $\mathbf{H}^{-1}$ también lo es[3]. De esta forma se cumple que $\mathbf{g}^T\mathbf{H}^{-1}\mathbf{g} > \mathbf{0}$, implicando que $-\mathbf{H}^{-1}\mathbf{g}$ presenta un ángulo inferior a 90º con respecto al negativo del vector gradiente $-\mathbf{g}$ (dirección *local* en la que $E(\mathbf{x})$ disminuye más rápidamente con respecto a $\mathbf{x}$), justificando así el descenso del valor de la función siempre que la Ec. 4.16 sea una aproximación válida.

La justificación de por qué $\mathbf{H}$ es positiva definida, se deriva de la definición de las matrices $\Sigma_{\mathbf{p}_c}$ de la Ec. 4.18. Como se ha comentado en §4.1.1, cada una de ellas se corresponde con la matriz de covarianza asociada a cada observación $\mathbf{p}_c$, las cuales son semi-definidas positivas por definición, y que, por construcción (se eligen a priori) se asegura que sean positivas definidas [Triggs et al., 1999, Sola, 2017a]. Ahora bien, la Ec. 4.18 de forma matricial se puede expresar de forma equivalente según la Ec. 4.20.

$$\mathbf{H} \approx \mathbf{J}^T\mathbf{\Sigma}^{-1}\mathbf{J}, \quad \mathbf{J} = \begin{bmatrix} \mathbf{J}_{\mathbf{p}_{c_1}} \\ \mathbf{J}_{\mathbf{p}_{c_2}} \\ \vdots \end{bmatrix}, \quad \mathbf{\Sigma}^{-1} = \text{diag}(\Sigma^{-1}_{\mathbf{p}_{c_1}}, \Sigma^{-1}_{\mathbf{p}_{c_2}}, \ldots), \tag{4.20}$$

donde $\mathbf{\Sigma}^{-1}$ es definida positiva al ser una matriz diagonal por bloques donde cada bloque $\Sigma^{-1}_{\mathbf{p}_{c_i}}$ es una matriz definida positiva. De esta forma:

$$\mathbf{u}^T\mathbf{H}\mathbf{u} = \mathbf{u}^T\mathbf{J}^T\mathbf{\Sigma}^{-1}\mathbf{J}\mathbf{u} = (\mathbf{J}\mathbf{u})^T\mathbf{\Sigma}^{-1}(\mathbf{J}\mathbf{u}) = \mathbf{y}^T\mathbf{\Sigma}^{-1}\mathbf{y} > \mathbf{0}, \tag{4.21}$$

Para cualquier par de vectores $\mathbf{u} \in \mathbb{R}^n, \mathbf{y} \in \mathbb{R}^m$, con $\mathbf{J} \in \mathbb{R}^{m\times n}$, y como además es simétrica: $\mathbf{H}^T = (\mathbf{J}^T\mathbf{\Sigma}\mathbf{J})^T = \mathbf{J}^T\mathbf{\Sigma}^T\mathbf{J} = \mathbf{J}^T\mathbf{\Sigma}\mathbf{J} = \mathbf{H}$, se concluye que esta aproximación de la matriz Hessiana resulta en una matriz definida positiva.

[2]En concreto, usamos la implementación de `scipy`. Solucionar el sistema de ecuaciones con este tipo de descomposición requiere un menor coste computacional que directamente invertir $\mathbf{H}$ [Golub and Van Loan,].

[3]Una matriz real, $\mathbf{H}_{n\times n}$, es definida positiva si es simétrica y todos sus valores propios son positivos. $\mathbf{H}^{-1}$ es simétrica ya que para cualquier matriz invertible: $(\mathbf{H}^{-1})^T = (\mathbf{H}^T)^{-1}$, y sabiendo que $\mathbf{H}^T = \mathbf{H} \Rightarrow (\mathbf{H}^{-1})^T = (\mathbf{H}^T)^{-1} = \mathbf{H}^{-1}$. Además todos sus valores propios son positivos también dado que son los inversos (recíprocos) de $\mathbf{H}$: $\mathbf{H}\mathbf{v} = \lambda\mathbf{v} \Rightarrow \frac{1}{\lambda}\mathbf{v} = \mathbf{H}^{-1}\mathbf{v}$. Por lo tanto $\mathbf{H}^{-1}$ es definida positiva.

4.2.1. Robustificación

Hasta ahora, el problema de optimización que se ha visto, trata de minimizar los errores cuadráticos asociados a cada observación $\|\mathbf{r}_{\mathbf{p}_c}\|^2_{\mathbf{\Sigma}_{\mathbf{p}_c}}$. Sin embargo, esto supone asumir que no existen observaciones erróneas o *espúreas*, es decir, que todas pertenecen al modelo que se está tratando de estimar.

Sin embargo, esto en la realidad no tiene por qué ser así (por ej. puede producirse un emparejamiento erróneo entre píxeles de imágenes diferentes). En situaciones en las que está asunción no es válida (hay **datos espúreos** o *outliers*), la optimización mediante mínimos cuadrados se vuelve susceptible a dar resultados insatisfactorios (ver ejemplo intuitivo de la Fig. 4.3b). Una forma de enfrentar este problema consiste en hacer uso de *funciones robustas* [Concha and Civera, 2015]. En este TFM se hace uso de la **función robusta de Huber** [Huber, 2004].

De esta forma, el problema de optimización de la Ec. 4.14 se modifica, minimizando en su lugar el error $E(\mathbf{x})$ de la Ec. 4.22.

$$E(\mathbf{x}) = \frac{1}{2}\sum_{\mathbf{p}_c\in\mathcal{P}_{zo}} \rho(\underbrace{\mathbf{r}^T_{\mathbf{p}_c}\Sigma^{-1}_{\mathbf{p}_c}\mathbf{r}_{\mathbf{p}_c}}_{L_{\mathbf{p}_c}}), \qquad \rho(L_{\mathbf{p}_c}) = \begin{cases} L_{\mathbf{p}_c} & \text{si } L_{\mathbf{p}_c} < k^2 \\ 2k\sqrt{L_{\mathbf{p}_c}} - k^2 & \text{si } L_{\mathbf{p}_c} \geq k^2 \end{cases} \tag{4.22}$$

Es decir, $\rho(\cdot)$ es cuadrática (con respecto al error) para valores pequeños de $\|\mathbf{r}_{\mathbf{p}_c}\|^2_{\mathbf{\Sigma}_{\mathbf{p}_c}}$ (por debajo de $k \in \mathbb{R}$) y lineal para valores elevados, limitando así la influencia de aquellas observaciones con un error asociado elevado y que presumiblemente pertenecen a observaciones espúreas (dada una inicialización lo suficientemente buena) [Eade, 2013]. El efecto intuitivo de esta función se muestra también en la Fig. 4.3.

Afortunadamente, la modificación anterior no afecta de forma significativa a las derivaciones iniciales. A través de la regla de la cadena, y las derivadas de $\rho(\cdot)$, se obtienen los nuevos gradiente y matriz Hessiana:

$$\mathbf{g} = \frac{\partial E(\mathbf{x})}{\partial \mathbf{x}} = \frac{1}{2}\sum_{\mathbf{p}_c\in\mathcal{P}_{zo}} \rho'_{\mathbf{p}_c}\frac{\partial L_{\mathbf{p}_c}}{\partial \mathbf{x}} \tag{4.23}$$

$$\overset{4,17}{=} \sum_{\mathbf{p}_c\in\mathcal{P}_{zo}} \rho'_{\mathbf{p}_c}\mathbf{r}^T_{\mathbf{p}_c}\Sigma^{-1}_{\mathbf{p}_c}\mathbf{J}_{\mathbf{p}_c}, \tag{4.24}$$

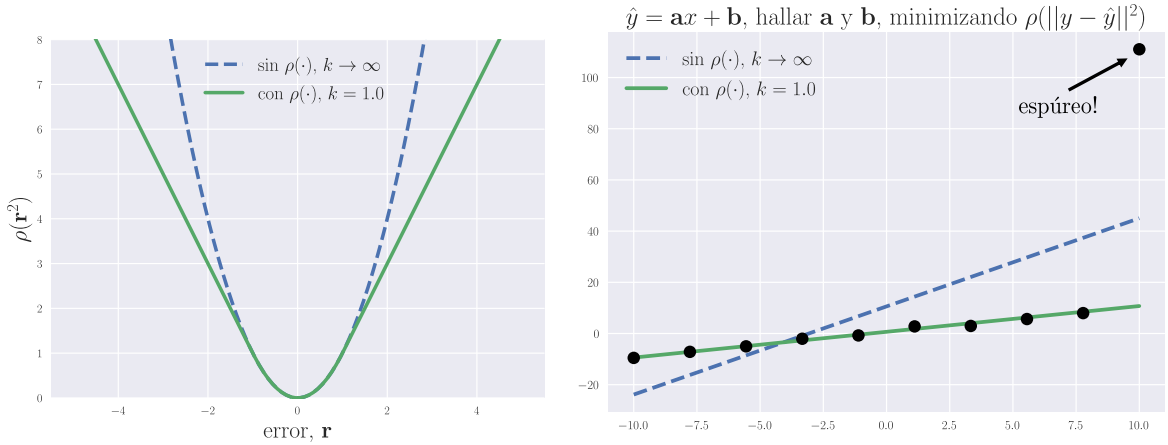
$$\mathbf{H} = \frac{\partial^2 E(\mathbf{x})}{\partial \mathbf{x}^2} = \frac{1}{2}\sum_{\mathbf{p}_c\in\mathcal{P}_{zo}} \rho'_{\mathbf{p}_c}\frac{\partial^2 L_{\mathbf{p}_c}}{\partial \mathbf{x}^2} + \rho''_{\mathbf{p}_c}\left(\frac{\partial L_{\mathbf{p}_c}}{\partial \mathbf{x}}\right)^2 \tag{4.25}$$

$$\overset{4,18}{\approx} \sum_{\mathbf{p}_c\in\mathcal{P}_{zo}} \mathbf{J}^T_{\mathbf{p}_c}\left(\rho'\Sigma^{-1}_{\mathbf{p}_c} + 2\rho''_{\mathbf{p}_c}\Sigma^{-1}_{\mathbf{p}_c}\mathbf{r}_{\mathbf{p}_c}\mathbf{r}^T_{\mathbf{p}_c}\Sigma^{-T}_{\mathbf{p}_c}\right)\mathbf{J}_{\mathbf{p}_c}, \tag{4.26}$$

con:

$$\rho'_{\mathbf{p}_c} = \frac{\partial \rho(L_{\mathbf{p}_c})}{\partial L_{\mathbf{p}_c}} = \begin{cases} 1 & \text{si } L_{\mathbf{p}_c} < k^2 \\ \frac{k}{L^{0,5}_{\mathbf{p}_c}} & \text{si } L_{\mathbf{p}_c} \geq k^2 \end{cases}, \quad \rho''_{\mathbf{p}_c} = \frac{\partial^2 \rho(L_{\mathbf{p}_c})}{\partial L^2_{\mathbf{p}_c}} = \begin{cases} 0 & \text{si } L_{\mathbf{p}_c} < k^2 \\ -\frac{k}{2L^{1,5}_{\mathbf{p}_c}} & \text{si } L_{\mathbf{p}_c} \geq k^2 \end{cases} \tag{4.27}$$

De esta forma, la robustificación tiene un doble efecto [Triggs et al., 1999]: 1) ponderar $\mathbf{g}$ y $\mathbf{H}$ con ρ', disminuyendo así la influencia de las observaciones espúreas en el cálculo



(a) (b)

Figura 4.3: ***(a)*** *Función cuadrática (azul) con respecto al error, r, y función robusta de Huber con $k = 1$ (verde). Se aprecia el comportamiento lineal de esta última conforme r aumenta.* ***(b)*** *Estimación, en presencia de un dato espúreo, de los parámetros de una línea tal que se ajuste a los puntos •, con y sin robustecimiento. Gracias a aplicar $\rho(\cdot)$, con una k adecuada, el dato espúreo es ignorado.*

de $\delta\mathbf{x}$, y 2) corregir la curvatura de la matriz Hessiana aproximada mediante el término asociado a ρ''.

Esta última contribución puede llegar a ser negativa, ya que si $\rho' + 2\rho''\|\mathbf{r}_{\mathbf{p}_c}\|^2 \leq 0$, la aproximación de la matriz Hessiana deja de ser definida positiva [Zach, 2014]. En nuestro caso, al usar la función de Huber, dicha cantidad es nula. Es por ello, que siguiendo la tendencia de implementaciones populares [Agarwal and Mierle, 2012, Kümmerle et al., 2011][4], se decidió implementar únicamente el primer efecto. Esta aplicación de las funciones robustas también se conoce como *mínimos cuadrados iterativamente reponderados* (**IRLS**) [Kerl et al., 2013].

4.2.2. Parametrización

Al tener como objetivo optimizar los puntos de control $\mathbf{T}_i$ y los puntos del objeto $\mathbf{p}_o$, puede parecer lógico que los elementos contenidos en el vector de estado $\mathcal{X}$, sean directamente $\mathbf{T}_i$ y $\mathbf{p}_o$. Esto, en cuanto a $\mathbf{p}_o$ es la opción más prudente ya que pertenecen a un espacio vectorial Euclídeo ($\mathbf{p}_o \in \mathbb{R}^3$), por lo que admiten actualizaciones del tipo $\mathbf{p}_o + \delta\mathbf{p}_o$.

Sin embargo, ésta no es una opción válida para los puntos de control. Actualizaciones del tipo $\mathbf{T}_i + \delta\mathbf{T}_i$ pueden provocar que éstos dejen de pertenecer a $SE(3)$, ya que este espacio no presenta clausura con la operación suma. Afortunadamente, tal y como se vio en §3.1, para una cinemática constante $\boldsymbol{\tau}^\wedge$ (expresada en el sistema de referencia mundo), la evolución temporal de una matriz de transformación $\mathbf{T}_0$ se puede expresar

[4]En nuestro caso, esas implementaciones no fueron empleadas ya que el lenguaje de programación que elegimos fue Python, y el de las librerías es C++.

como:

$$\mathbf{T}(t) \overset{3,37}{=} \exp(\underbrace{\boldsymbol{\tau}^{\wedge} t}_{\boldsymbol{\xi}^{\wedge}})\mathbf{T}_0, \tag{4.28}$$

por lo que una variación infinitesimal con respecto a este incremento (o *perturbación*) $\boldsymbol{\xi}$, de un punto de control $\mathbf{T}_i$ la podemos expresar como:

$$\frac{\partial}{\partial \boldsymbol{\xi}_i} \mathrm{Exp}(\boldsymbol{\xi}_i)\mathbf{T}_i \Big|_{\boldsymbol{\xi}_i = \mathbf{0}} \tag{4.29}$$

Resultando en una de las parametrizaciones más extendidas de la literatura reciente [Blanco, 2010, Strasdat, 2012, Kümmerle et al., 2011]. Es decir, en nuestro vector de estado $\mathcal{X}$, no incluimos la matriz de transformación como tal, sino que incluimos el elemento del álgebra de Lie asociado a la perturbación: $\boldsymbol{\xi}_i$[5]. Por ello, tras determinar mediante el algoritmo de Gauss-Newton, qué valor de $\boldsymbol{\xi}_i$ minimiza $E(\mathcal{X})$, podemos actualizar el punto de control mediante $\exp(\boldsymbol{\xi}_i)\mathbf{T}_i$, sin riesgo a que deje de pertenecer a $SE(3)$.

Esta parametrización de la matriz de transformación presenta ventajas sobre varias alternativas [Kümmerle et al., 2011, Lynch and Park, 2017]. Por un lado, es una *parametrización mínima* (sin restricciones) $\rightarrow \boldsymbol{\xi} \in \mathbb{R}^6$, que condensa los 6 grados de libertad de un movimiento rígido, y por otro, es una representación que, bajo magnitudes pequeñas de $\boldsymbol{\xi}$, no presenta singularidades (a diferencia de los ángulos de Euler, por ejemplo).

4.3. Matrices Jacobianas

Lo único restante para tener definido completamente el bloque de optimización (Fig. 2.1), es explicar cómo obtener las **matrices Jacobianas** introducidas en la Ec. 4.17, y que representan la diferenciación de cada error $\mathbf{r}_{\mathbf{p}_c}(\mathbf{x})$ con respecto a las variables de estado $\mathbf{x}$.

Por comodidad y buscando mayor claridad, en esta sección se relaja la notación asociada a un error $\mathbf{r}_{\mathbf{p}_c}$:

$$\mathbf{r} \triangleq \mathbf{r}_{\mathbf{p}_c}, \tag{4.30}$$

es decir, se deja indicar explícitamente que está asociado a una observación $\mathbf{p}_c$. De forma general, denominándola como una función $\mathbf{r}(\mathbf{x}) : \mathbb{R}^n \rightarrow \mathbb{R}^m$, su matriz Jacobiana, $\mathbf{J}_{\mathbf{r}}(\mathbf{x})$, viene dada por:

$$\mathbf{J}_{\mathbf{r}}(\mathbf{x}) = \begin{bmatrix} \frac{\partial \mathbf{r}_1}{\partial \mathbf{x}_1} & \cdots & \frac{\partial \mathbf{r}_1}{\partial \mathbf{x}_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial \mathbf{r}_m}{\partial \mathbf{x}_1} & \cdots & \frac{\partial \mathbf{r}_m}{\partial \mathbf{x}_n} \end{bmatrix} \tag{4.31}$$

[5]En este caso, hemos utilizado el plano tangente a la identidad para definir la perturbación. Podríamos igualmente haberla definido en el plano tangente a $\mathbf{T}$ como: $\mathbf{T}\exp(\boldsymbol{\xi}_j)$. Es más, ambas perturbaciones están relacionadas por la matriz adjunta de $\mathbf{T}$: $\boldsymbol{\xi}_i = \mathrm{Ad}_{\mathbf{T}}\boldsymbol{\xi}_j$.

Estas matrices Jacobianas, en problemas de *tiempo discreto*, han sido publicadas y ampliamente tratadas [Strasdat, 2012, Blanco, 2010, Forster et al., 2016]. Sin embargo, bajo nuestro conocimiento, en estimaciones de *tiempo continuo* como las que se tratan en este TFM (B-Splines cumulativos), aún no han sido cubiertas para el caso particular de estimación de una trayectoria en $SE(3)$.

Recientemente, con el trabajo de [Sommer et al., 2020], se derivaron las matrices Jacobianas para matrices de rotación estimadas con B-Splines. Sin embargo, su derivación en $SE(3)$ no fue tratada. Por ello, esta sección supone una de las **principales contribuciones de este trabajo**. En §6.2, se realizan experimentos con dos versiones distintas de cálculo de estas matrices, y se compara su eficiencia con los métodos de cálculo (diferenciación automática y numérica) que emplean la mayoría de trabajos que han hecho uso de B-Splines. Mostrando así las ventajas del método propuesto.

Particularizando a nuestro caso la Ec. 4.31, como $\mathbf{r} \in \mathbb{R}^3 \Rightarrow m = 3$. En cuanto al vector de estado, su tamaño n viene determinado por el de la ventana temporal que se esté considerando y de si se está empleando *Spline BA* (optimización de puntos de control) o *Local BA* (optimización de puntos de control y puntos del objeto).

De esta forma, existen variables de estado que no guardan relación con un error $\mathbf{r}$, por lo que su derivada asociada es nula. En esta sección solo se hace referencia a los elementos no nulos de $\mathbf{J_r}(\mathbf{x})$, ya que conociéndolos solo se necesita una reordenación de los mismos para ajustarlos a las columnas de $\mathbf{J_r}(\mathbf{x})$.

Incluyendo las perturbaciones $\boldsymbol{\xi} \in \mathbb{R}^6$ introducidas en §4.2.2 y la notación que se emplea en esta sección, un error $\mathbf{r}$ en un instante t se expresa como:

$$
\begin{aligned}
\mathbf{r}(t) &= \mathbf{p}_c - \text{proj}\left(\mathbf{T}_{cw}\mathbf{T}_{wo}(t)\tilde{\mathbf{p}}_o\right) = \mathbf{p}_c - \text{proj}\left(\mathbf{T}_{co}(t)\tilde{\mathbf{p}}_o\right), && \mathbb{R}^3, && (4.32)\\
\mathbf{T}_{wo}(t) &= \text{Exp}(\mathbf{a}_0)\mathbf{T}_0\mathbf{A}_1(t)\mathbf{A}_2(t)\mathbf{A}_3(t), && SE(3), && (4.33)\\
\mathbf{A}_j(t) &= \text{Exp}(\mathbf{a}_j(t)), && SE(3), && (4.34)\\
\mathbf{a}_j(t) &= \tilde{B}_j(t)\boldsymbol{\Omega}_j, && \mathbb{R}^6, && (4.35)\\
\mathbf{a}_0 &= \boldsymbol{\xi}_0|_{\boldsymbol{\xi}_0=\mathbf{0}}, && \mathbb{R}^6 && (4.36)\\
\boldsymbol{\Omega}_j &= \text{Log}\left((\text{Exp}(\boldsymbol{\xi}_{j-1})\mathbf{T}_{j-1})^{-1}\text{Exp}(\boldsymbol{\xi}_j)\mathbf{T}_j)\right)\big|_{\boldsymbol{\xi}_{j-1}=\boldsymbol{\xi}_j=\mathbf{0}}, && \mathbb{R}^6, && (4.37)
\end{aligned}
$$

con $j \in \{1, 2, 3\}$.

4.3.1. Derivación independiente del error

Como primera forma de obtener la matriz jacobiana, en esta sección se propone una **versión general**, independiente de la función de error elegida, facilitando de esta forma su aplicación a otros problemas. Esto es así, ya que no se aprovecha ninguna característica particular de $\mathbf{r}$, al contrario de la derivación propuesta en §4.3.2.

En concreto, las diferenciaciones propuestas, derivadas de aplicar la regla de la

cadena (multivariable), son:

$$\text{Para } \boldsymbol{\xi}_j, \quad j \in \{0,1,2\} \quad : \quad \frac{\partial \mathbf{r}}{\partial \boldsymbol{\xi}_j} = \frac{\partial \mathbf{r}}{\partial \mathbf{T}_{wo}} \left(\frac{\partial \mathbf{T}_{wo}}{\partial \mathbf{a}_j} \frac{\partial \mathbf{a}_j}{\partial \boldsymbol{\xi}_j} + \frac{\partial \mathbf{T}_{wo}}{\partial \mathbf{a}_{j+1}} \frac{\partial \mathbf{a}_{j+1}}{\partial \boldsymbol{\xi}_j} \right) \tag{4.38}$$

$$\text{Para } \boldsymbol{\xi}_3 \quad : \quad \frac{\partial \mathbf{r}}{\partial \boldsymbol{\xi}_3} = \frac{\partial \mathbf{r}}{\partial \mathbf{T}_{wo}} \frac{\partial \mathbf{T}_{wo}}{\partial \mathbf{a}_3} \frac{\partial \mathbf{a}_3}{\partial \boldsymbol{\xi}_3} \tag{4.39}$$

Se aprecia la independencia con la función de error dado que se está diferenciando $\mathbf{T}_{wo}(t)$ de manera directa con respecto a las perturbaciones $\boldsymbol{\xi}$.

Derivación de $\partial \mathbf{r}/\partial \mathbf{T}_{wo}$ El primer término, $\partial \mathbf{r}/\partial \mathbf{T}_{wo}$, supone la diferenciación de un vector $\mathbf{r}$ con respecto a una matriz $\mathbf{T}_{wo}$. Esto, directamente no se puede representar en una matriz de 2 dimensiones como en la Ec. 4.31. Sería necesario un tensor. En su lugar, para facilitar los cálculos, en estas situaciones hacemos uso de la versión vectorizada, vec($\mathbf{T}$), de la matriz:

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}^{c1} & \mathbf{R}^{c2} & \mathbf{R}^{c3} & \mathbf{t} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \Rightarrow \quad \text{vec}(\mathbf{T}) = \begin{bmatrix} \mathbf{R}^{c1} \\ \mathbf{R}^{c2} \\ \mathbf{R}^{c3} \\ \mathbf{t} \end{bmatrix} \in \mathbb{R}^{12} \tag{4.40}$$

La última fila se ignora por ser términos constantes. Arrastrarlos supondría operaciones innecesarias. Es esta versión vectorizada la empleada en las Ecs. 4.38 y 4.39. Por otro lado, $\mathbf{R}^{ci}$ y $\mathbf{R}^{ri}$ hacen referencia a la columna y fila i de $\mathbf{R}$ respectivamente, y $\mathbf{R}^{ij}$ al elemento ij (en vectores solo indicamos un índice).

Para derivar este término podemos recurrir a la regla de la cadena:

$$\frac{\partial \mathbf{r}}{\partial \mathbf{T}_{wo}} = \frac{\partial \mathbf{r}}{\partial \mathbf{T}_{co}} \frac{\partial \mathbf{T}_{co}}{\partial \mathbf{T}_{wo}}, \tag{4.41}$$

a partir de los resultados detallados en las Ecs. A.4 y A.8 obtenemos:

$$\frac{\partial \mathbf{r}}{\partial \mathbf{T}_{co}} = -\begin{bmatrix} \mathbf{p}_o^T & 1 \end{bmatrix} \otimes \mathbf{I}_{3\times 3}, \qquad \frac{\partial \mathbf{T}_{co}}{\partial \mathbf{T}_{wo}} = \mathbf{I}_{4\times 4} \otimes \mathbf{R}_{cw}, \tag{4.42}$$

Donde $\otimes$ representa el producto de Kronecker [Van Loan, 2000], el cual condensa la siguiente operación:

$$\mathbf{B} \otimes \mathbf{C} = \begin{bmatrix} \mathbf{B}^{11}\mathbf{C} & \cdots & \mathbf{B}^{1n}\mathbf{C} \\ \vdots & \ddots & \vdots \\ \mathbf{B}^{m1}\mathbf{C} & \cdots & \mathbf{B}^{mn}\mathbf{C} \end{bmatrix} \tag{4.43}$$

Para dos matrices $\mathbf{B}_{m\times n}, \mathbf{C}_{o\times p}$ cualesquiera. Por lo que $\mathbf{B} \otimes \mathbf{C}$ da como resultado una matriz de tamaño $(m \cdot o) \times (n \cdot p)$. Por ello, concatenando ambas diferenciaciones, finalmente obtenemos:

$$\frac{\partial \mathbf{r}}{\partial \mathbf{T}_{wo}} = -\begin{bmatrix} \mathbf{p}_o^T & 1 \end{bmatrix} \otimes \mathbf{R}_{cw} \tag{4.44}$$

Derivación de $\partial \mathbf{T}_{wo}/\partial \mathbf{a}_j$ A partir de las definiciones de las Ecs. 4.32-4.37, y de la propiedad de la Ec. 3.58, podemos re-escribir esta diferenciación como:

$$\frac{\partial \mathbf{T}_{wo}}{\partial \mathbf{a}_j} = \frac{\partial}{\partial \mathbf{a}_j}\mathbf{P}_j \text{Exp}(\mathbf{a}_j)\mathbf{N}_j \tag{4.45}$$

$$= \frac{\partial}{\partial \boldsymbol{\tau}_j}\mathbf{P}_j \text{Exp}(\mathbf{a}_j + \boldsymbol{\tau}_j)\mathbf{N}_j\Big|_{\boldsymbol{\tau}_j=\mathbf{0}} \tag{4.46}$$

$$\overset{3,58}{=} \frac{\partial}{\partial \boldsymbol{\tau}_j}\mathbf{P}_j \underbrace{\text{Exp}(\mathbf{J}_l(\mathbf{a}_j)\boldsymbol{\tau}_j)}_{\mathbf{C}(\boldsymbol{\tau}_j)} \underbrace{\text{Exp}(\mathbf{a}_j)\mathbf{N}_j}_{\mathbf{N}'_j}\Big|_{\boldsymbol{\tau}_j=\mathbf{0}} \tag{4.47}$$

$$= \frac{\partial \mathbf{P}_j\mathbf{C}(\boldsymbol{\tau}_j)\mathbf{N}'_j}{\partial \mathbf{C}(\boldsymbol{\tau}_j)\mathbf{N}'_j}\Big|_{\boldsymbol{\tau}_j=\mathbf{0}} \frac{\partial \mathbf{C}(\boldsymbol{\tau}_j)\mathbf{N}'_j}{\partial \mathbf{C}(\boldsymbol{\tau}_j)}\Big|_{\boldsymbol{\tau}_j=\mathbf{0}} \frac{\partial \text{Exp}(\mathbf{J}_l(\mathbf{a}_j)\boldsymbol{\tau}_j)}{\partial \mathbf{J}_l(\mathbf{a}_j)\boldsymbol{\tau}_j}\Big|_{\boldsymbol{\tau}_j=\mathbf{0}} \frac{\partial \mathbf{J}_l(\mathbf{a}_j)\boldsymbol{\tau}_j}{\partial \boldsymbol{\tau}_j}\Big|_{\boldsymbol{\tau}_j=\mathbf{0}} \tag{4.48}$$

donde:

$$j=0 \quad \rightarrow \quad \mathbf{P}_0 = \mathbf{I}_{4\times4}, \qquad \mathbf{N}'_0 = \mathbf{T}_{wo} \tag{4.49}$$

$$j=1 \quad \rightarrow \quad \mathbf{P}_1 = \mathbf{T}_0, \qquad \mathbf{N}'_1 = \mathbf{A}_1\mathbf{A}_2\mathbf{A}_3 \tag{4.50}$$

$$j=2 \quad \rightarrow \quad \mathbf{P}_2 = \mathbf{T}_0\mathbf{A}_1, \qquad \mathbf{N}'_2 = \mathbf{A}_2\mathbf{A}_3 \tag{4.51}$$

$$j=3 \quad \rightarrow \quad \mathbf{P}_3 = \mathbf{T}_0\mathbf{A}_1\mathbf{A}_2, \quad \mathbf{N}'_3 = \mathbf{A}_3 \tag{4.52}$$

El último término de la Ec. 4.48 tiene la solución más directa:

$$\frac{\partial \mathbf{J}_l(\mathbf{a}_j)\boldsymbol{\tau}_j}{\partial \boldsymbol{\tau}_j}\Big|_{\boldsymbol{\tau}_j=\mathbf{0}} = \mathbf{J}_l(\mathbf{a}_j), \tag{4.53}$$

el primer y segundo término se corresponden con la diferenciación de una matriz vectorizada con respecto a otra. A partir de los resultados A.8 y A.10 conocemos su solución:

$$\frac{\partial \mathbf{P}_j\mathbf{C}(\boldsymbol{\tau}_j)\mathbf{N}'_j}{\partial \mathbf{C}(\boldsymbol{\tau}_j)\mathbf{N}'_j}\Big|_{\boldsymbol{\tau}_j=\mathbf{0}} = \frac{\partial \mathbf{P}_j\mathbf{C}(\boldsymbol{\tau}_j)\mathbf{N}'_j}{\partial \mathbf{C}(\boldsymbol{\tau}_j)\mathbf{N}'_j}\Big|_{\mathbf{C}(\boldsymbol{\tau}_j)\mathbf{N}'_j=\mathbf{N}'_j} = \mathbf{I}_{4\times4} \otimes \mathbf{R}_{\mathbf{P}_j} \tag{4.54}$$

$$\frac{\partial \mathbf{C}(\boldsymbol{\tau}_j)\mathbf{N}'_j}{\partial \mathbf{C}(\boldsymbol{\tau}_j)}\Big|_{\boldsymbol{\tau}_j=\mathbf{0}} = \frac{\partial \mathbf{C}(\boldsymbol{\tau}_j)\mathbf{N}'_j}{\partial \mathbf{C}(\boldsymbol{\tau}_j)}\Big|_{\mathbf{C}(\boldsymbol{\tau}_j)=\mathbf{I}_{4\times4}} = \mathbf{N}_j^T \otimes \mathbf{I}_{3\times3} \tag{4.55}$$

Donde $\mathbf{R}_{\mathbf{P}_j}$ hace referencia a la matriz de rotación de de $\mathbf{P}_j$. Juntando ambas derivadas:

$$\frac{\partial \mathbf{P}_j\mathbf{C}(\boldsymbol{\tau}_j)\mathbf{N}'_j}{\partial \mathbf{C}(\boldsymbol{\tau}_j)}\Big|_{\boldsymbol{\tau}_j=\mathbf{0}} = \mathbf{N}_j^T \otimes \mathbf{R}_{\mathbf{P}_j} \tag{4.56}$$

Por último, el tercer término de la Ec. 4.48, lo podemos calcular diferenciando la Ec. 3.46 con respecto a $\boldsymbol{\tau}$. Evaluándola en $\boldsymbol{\tau} = \mathbf{0}$ y expresándola de forma vectorizada, se obtiene (derivación detallada en §A):

$$\frac{\partial \text{Exp}(\mathbf{J}_l(\mathbf{a}_j)\boldsymbol{\tau}_j)}{\partial \mathbf{J}_l(\mathbf{a}_j)\boldsymbol{\tau}_j}\Big|_{\boldsymbol{\tau}_j=\mathbf{0}} = \frac{\partial \text{Exp}(\mathbf{J}_l(\mathbf{a}_j)\boldsymbol{\tau}_j)}{\partial \mathbf{J}_l(\mathbf{a}_j)\boldsymbol{\tau}_j}\Big|_{\mathbf{J}_l(\mathbf{a}_j)\boldsymbol{\tau}_j=\mathbf{0}} = \begin{bmatrix} \mathbf{0}_{3\times3} & -\mathbf{G}_1 \\ \mathbf{0}_{3\times3} & -\mathbf{G}_2 \\ \mathbf{0}_{3\times3} & -\mathbf{G}_3 \\ \mathbf{I}_{3\times3} & -\mathbf{0}_{3\times3} \end{bmatrix} \tag{4.57}$$

Donde los términos $\mathbf{G}_i$, $i \in \{1,2,3\}$, representan los *generadores* de SO(3) [Strasdat, 2012] (vectores base del plano tangente a la identidad de SO(3)), los cuales vienen dados por:

$$\mathbf{G}_1 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}, \qquad \mathbf{G}_2 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix}, \qquad \mathbf{G}_3 = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{4.58}$$

A esta conclusión también se puede llegar a través de los generadores de $SE(3)$, ya que éstos son los vectores tangentes (en la identidad) de un camino (o *path*) perteneciente a $SE(3)$ que pasa por el elemento identidad [Strasdat, 2012]. Expresándolos de manera vectorizada se llega igualmente a la Ec. 4.57.

Por lo que, el resultado final viene dado por:

$$\frac{\partial \mathbf{T}_{wo}}{\partial \mathbf{a}_j} = (\mathbf{N}_j^T \otimes \mathbf{R}_{\mathbf{P}_j}) \begin{bmatrix} \mathbf{0}_{3\times3} & -\mathbf{G}_1 \\ \mathbf{0}_{3\times3} & -\mathbf{G}_2 \\ \mathbf{0}_{3\times3} & -\mathbf{G}_3 \\ \mathbf{I}_{3\times3} & -\mathbf{0}_{3\times3} \end{bmatrix} \mathbf{J}_l(\mathbf{a}_j) = \begin{bmatrix} \mathbf{0}_{3\times3} & -\mathbf{R}_{\mathbf{P}_j}(\mathbf{R}_{\mathbf{N}_j}^{c1})^\wedge \\ \mathbf{0}_{3\times3} & -\mathbf{R}_{\mathbf{P}_j}(\mathbf{R}_{\mathbf{N}_j}^{c2})^\wedge \\ \mathbf{0}_{3\times3} & -\mathbf{R}_{\mathbf{P}_j}(\mathbf{R}_{\mathbf{N}_j}^{c3})^\wedge \\ \mathbf{R}_{\mathbf{P}_j} & -\mathbf{R}_{\mathbf{P}_j}\mathbf{t}_{\mathbf{N}_j}^\wedge \end{bmatrix} \mathbf{J}_l(\mathbf{a}_j) \tag{4.59}$$

Destacar que si $j = 0$, simplificaciones importantes ocurren: $\mathbf{J}_l(\mathbf{a}_j) = \mathbf{I}_{6\times6}$, $\mathbf{P}_j = \mathbf{I}_{4\times4}$.

Derivación de $\partial \mathbf{a}_j / \partial \boldsymbol{\xi}_j$ y de $\partial \mathbf{a}_{j+1} / \partial \boldsymbol{\xi}_j$ Para estas derivaciones es posible realizar una simplificación previa. En concreto, una parte de los términos de la Ec. 4.37 se puede re-expresar de la siguiente forma:

$$(\mathrm{Exp}(\boldsymbol{\xi}_{j-1})\mathbf{T}_{j-1})^{-1} = \mathbf{T}_{j-1}^{-1}\mathrm{Exp}(\boldsymbol{\xi}_{j-1})^{-1} = \mathbf{T}_{j-1}^{-1}\mathrm{Exp}(-\boldsymbol{\xi}_{j-1}) \tag{4.60}$$

Por lo que la Ec. 4.37 se puede re-escribir de acuerdo con lo anterior:

$$\boldsymbol{\Omega}_j = \mathrm{Log}\left(\mathbf{T}_{j-1}^{-1}\mathrm{Exp}(-\boldsymbol{\xi}_{j-1})\mathrm{Exp}(\boldsymbol{\xi}_j)\mathbf{T}_j\right)\big|_{\boldsymbol{\xi}_{j-1}=\boldsymbol{\xi}_j=\mathbf{0}} \tag{4.61}$$

Lo que implica las siguientes relaciones:

$$\frac{\partial \boldsymbol{\Omega}_j}{\partial \boldsymbol{\xi}_{j-1}} = -\frac{\partial \boldsymbol{\Omega}_j}{\partial \boldsymbol{\xi}_j}, \qquad \frac{\partial \mathbf{a}_j}{\partial \boldsymbol{\xi}_{j-1}} = -\frac{\partial \mathbf{a}_j}{\partial \boldsymbol{\xi}_j}, \tag{4.62}$$

Gracias a esto, únicamente necesitamos calcular $(\partial \mathbf{a}_j / \partial \boldsymbol{\xi}_j)|_{\boldsymbol{\xi}_j=\mathbf{0}}$. A partir de esta consideración y de las propiedades de la matriz Adjunta y Jacobiano izquierdo vistas en las Ecs. 3.50 y 3.60 respectivamente, cada término $(\partial \mathbf{a}_j / \partial \boldsymbol{\xi}_j)|_{\boldsymbol{\xi}_j=\mathbf{0}}$ para $j \in \{1,2,3\}$,

puede calcularse de la siguiente forma:

$$\left.\frac{\partial \mathbf{a}_j}{\partial \boldsymbol{\xi}_j}\right|_{\boldsymbol{\xi}_j=\mathbf{0}} = \left.\frac{\partial}{\partial \boldsymbol{\xi}_j}\tilde{B}_j(t)\mathrm{Log}\left(\mathbf{T}_{j-1}^{-1}\mathrm{Exp}(\boldsymbol{\xi}_j)\mathbf{T}_j\right)\right|_{\boldsymbol{\xi}_j=\mathbf{0}} \tag{4.63}$$

$$\overset{3,50}{=} \left.\frac{\partial}{\partial \boldsymbol{\xi}_j}\tilde{B}_j(t)\mathrm{Log}\left(\mathrm{Exp}(\mathrm{Ad}_{\mathbf{T}_{j-1}^{-1}}\boldsymbol{\xi}_j)\mathbf{T}_{j-1}^{-1}\mathbf{T}_j\right)\right|_{\boldsymbol{\xi}_j=\mathbf{0}} \tag{4.64}$$

$$\overset{3,60}{=} \left.\frac{\partial}{\partial \boldsymbol{\xi}_j}\tilde{B}_j(t)\left(\mathrm{Log}\left(\mathbf{T}_{j-1}^{-1}\mathbf{T}_j\right) + \mathbf{J}_l^{-1}\left(\mathrm{Log}(\mathbf{T}_{j-1}^{-1}\mathbf{T}_j)\right)\mathrm{Ad}_{\mathbf{T}_{j-1}^{-1}}\boldsymbol{\xi}_j\right)\right|_{\boldsymbol{\xi}_j=\mathbf{0}} \tag{4.65}$$

$$= \left.\frac{\partial}{\partial \boldsymbol{\xi}_j}\tilde{B}_j(t)\ \mathbf{J}_l^{-1}\left(\mathrm{Log}(\mathbf{T}_{j-1}^{-1}\mathbf{T}_j)\right)\mathrm{Ad}_{\mathbf{T}_{j-1}^{-1}}\boldsymbol{\xi}_j\right|_{\boldsymbol{\xi}_j=\mathbf{0}} \tag{4.66}$$

$$= \tilde{B}_j(t)\ \mathbf{J}_l^{-1}\left(\mathrm{Log}(\mathbf{T}_{j-1}^{-1}\mathbf{T}_j)\right)\mathrm{Ad}_{\mathbf{T}_{j-1}^{-1}} \tag{4.67}$$

Solamente falta derivar $(\partial \mathbf{a}_0/\partial \boldsymbol{\xi}_0)|_{\boldsymbol{\xi}_0=\mathbf{0}}$. Afortunadamente esta derivación es trivial, pues habíamos definido $\mathbf{a}_0 = \boldsymbol{\xi}_0|_{\boldsymbol{\xi}_0=\mathbf{0}}$ (Ec. 4.36) $\Rightarrow (\partial \mathbf{a}_0/\partial \boldsymbol{\xi}_0)|_{\boldsymbol{\xi}_0=\mathbf{0}} = \mathbf{I}_{3\times 3}$. Recopilando estos resultados:

$$\left.\frac{\partial \mathbf{a}_j}{\partial \boldsymbol{\xi}_j}\right|_{\boldsymbol{\xi}_j=\mathbf{0}} = \tilde{B}_j(t)\ \mathbf{J}_l^{-1}\left(\mathrm{Log}(\mathbf{T}_{j-1}^{-1}\mathbf{T}_j)\right)\mathrm{Ad}_{\mathbf{T}_{j-1}^{-1}}, \quad j \in \{1,2,3\} \tag{4.68}$$

$$\left.\frac{\partial \mathbf{a}_0}{\partial \boldsymbol{\xi}_0}\right|_{\boldsymbol{\xi}_0=\mathbf{0}} = \mathbf{I}_{3\times 3} \tag{4.69}$$

Completando así la derivación de todos los términos de las Ecs. 4.38 y 4.39.

4.3.2. Derivación aprovechando la definición del error

Así mismo también se propone el siguiente cálculo de la matriz Jacobiana que a primera vista puede parecer más directo:

$$\text{Para } \boldsymbol{\xi}_j, \quad j \in \{0,1,2\} \quad : \quad \frac{\partial \mathbf{r}}{\partial \boldsymbol{\xi}_j} = \frac{\partial \mathbf{r}}{\partial \mathbf{a}_j}\frac{\partial \mathbf{a}_j}{\partial \boldsymbol{\xi}_j} + \frac{\partial \mathbf{r}}{\partial \mathbf{a}_{j+1}}\frac{\partial \mathbf{a}_{j+1}}{\partial \boldsymbol{\xi}_j} \tag{4.70}$$

$$\text{Para } \boldsymbol{\xi}_3 \quad : \quad \frac{\partial \mathbf{r}}{\partial \boldsymbol{\xi}_3} = \frac{\partial \mathbf{r}}{\partial \mathbf{a}_3}\frac{\partial \mathbf{a}_3}{\partial \boldsymbol{\xi}_3} \tag{4.71}$$

La diferencia con la propuesta anterior reside en que los términos $\partial \mathbf{r}/\partial \mathbf{a}_j$, dependen del punto $\mathbf{p}_o$ correspondiente. Algo que solo sucedía antes con el término $\partial \mathbf{r}/\partial \mathbf{T}_{wo}$.

Esto tiene consecuencias de cara a la implementación de la matriz Jacobiana *para múltiples* puntos $\mathbf{p}_o$. Para agilizar su cómputo, es conveniente *vectorizar* las operaciones, evitando así un cálculo particular para cada punto $\mathbf{p}_o$ (evitando *bucles for*). En la primera versión por tanto, se vectoriza $\partial \mathbf{r}/\partial \mathbf{T}_{wo}$, mientras que en la segunda se vectoriza $\partial \mathbf{r}/\partial \mathbf{a}_j$, lo que puede resultar en eficiencias diferentes, analizadas en §6.2.

El único término de las Ecs. 4.70 y 4.71 que resulta nuevo, es el de $\partial \mathbf{r}/\partial \mathbf{a}_j$. El resto ya han sido calculados en §4.3.1. Su cálculo se puede descomponer aplicando la regla

de la cadena:

$$\frac{\partial \mathbf{r}}{\partial \mathbf{a}_j} = -\frac{\partial}{\partial \mathbf{a}_j}\text{proj}\left(\mathbf{P}_j \text{Exp}(\mathbf{a}_j)\tilde{\mathbf{p}}_j\right) \tag{4.72}$$

$$= -\frac{\partial}{\partial \boldsymbol{\tau}_j}\text{proj}\left(\mathbf{P}_j \text{Exp}(\mathbf{a}_j + \boldsymbol{\tau}_j)\tilde{\mathbf{p}}_j\right)\Big|_{\boldsymbol{\tau}_j=\mathbf{0}} \tag{4.73}$$

$$\overset{3,58}{=} -\frac{\partial}{\partial \boldsymbol{\tau}_j}\text{proj}(\mathbf{P}_j \underbrace{\text{Exp}(\mathbf{J}_l(\mathbf{a}_j)\boldsymbol{\tau}_j)}_{\mathbf{C}(\boldsymbol{\tau}_j)} \underbrace{\text{Exp}(\mathbf{a}_j)\tilde{\mathbf{p}}_j}_{\tilde{\mathbf{p}}'_j})\Big|_{\boldsymbol{\tau}_j=\mathbf{0}} \tag{4.74}$$

$$= -\frac{\partial \text{proj}(\mathbf{P}_j\mathbf{C}(\boldsymbol{\tau}_j)\tilde{\mathbf{p}}'_j)}{\partial \text{proj}(\mathbf{C}(\boldsymbol{\tau}_j)\tilde{\mathbf{p}}'_j)}\Big|_{\boldsymbol{\tau}_j=\mathbf{0}} \frac{\partial \text{proj}(\mathbf{C}(\boldsymbol{\tau}_j)\tilde{\mathbf{p}}'_j)}{\partial \mathbf{C}(\boldsymbol{\tau}_j)}\Big|_{\boldsymbol{\tau}_j=\mathbf{0}} \frac{\partial \text{Exp}(\mathbf{J}_l(\mathbf{a}_j)\boldsymbol{\tau}_j)}{\partial \mathbf{J}_l(\mathbf{a}_j)\boldsymbol{\tau}_j}\Big|_{\boldsymbol{\tau}_j=\mathbf{0}} \frac{\partial \mathbf{J}_l(\mathbf{a}_j)\boldsymbol{\tau}_j}{\boldsymbol{\tau}_j}\Big|_{\boldsymbol{\tau}_j=\mathbf{0}} \tag{4.75}$$

donde:

$$j=0 \quad \rightarrow \quad \mathbf{P}_0 = \mathbf{T}_{cw}, \qquad \tilde{\mathbf{p}}'_0 = \mathbf{T}_{wo}\tilde{\mathbf{p}}_o \tag{4.76}$$

$$j=1 \quad \rightarrow \quad \mathbf{P}_1 = \mathbf{T}_{cw}\mathbf{T}_0, \qquad \tilde{\mathbf{p}}'_1 = \mathbf{A}_1\mathbf{A}_2\mathbf{A}_3\tilde{\mathbf{p}}_o \tag{4.77}$$

$$j=2 \quad \rightarrow \quad \mathbf{P}_2 = \mathbf{T}_{cw}\mathbf{T}_0\mathbf{A}_1, \qquad \tilde{\mathbf{p}}'_2 = \mathbf{A}_2\mathbf{A}_3\tilde{\mathbf{p}}_o \tag{4.78}$$

$$j=3 \quad \rightarrow \quad \mathbf{P}_3 = \mathbf{T}_{cw}\mathbf{T}_0\mathbf{A}_1\mathbf{A}_2, \qquad \tilde{\mathbf{p}}'_3 = \mathbf{A}_3\tilde{\mathbf{p}}_o \tag{4.79}$$

Los dos últimos términos de la Ec. 4.75 son exactamente iguales a los derivados en las Ecs. 4.57 y 4.53. Por otro lado, el segundo término se obtiene mediante el resultado A.4:

$$\frac{\partial \text{proj}(\mathbf{C}(\boldsymbol{\tau}_j)\tilde{\mathbf{p}}'_j)}{\partial \mathbf{C}(\boldsymbol{\tau}_j)}\Big|_{\boldsymbol{\tau}_j=\mathbf{0}} = \begin{bmatrix}\mathbf{p}'_j & 1\end{bmatrix} \otimes \mathbf{I}_{3\times 3} \tag{4.80}$$

Finalmente, el primer término de la Ec. 4.75, viene dado por:

$$\frac{\partial \text{proj}(\mathbf{P}_j\mathbf{C}(\boldsymbol{\tau}_j)\tilde{\mathbf{p}}'_j)}{\partial \text{proj}(\mathbf{C}(\boldsymbol{\tau}_j)\tilde{\mathbf{p}}'_j)}\Big|_{\boldsymbol{\tau}_j=\mathbf{0}} = \mathbf{R}_{\mathbf{P}_j} \tag{4.81}$$

Por lo que, juntando todo, llegamos al resultado final:

$$\frac{\partial \mathbf{r}}{\partial \mathbf{a}_j} = -\mathbf{R}_{\mathbf{P}_j}\left(\begin{bmatrix}\mathbf{p}'_j & 1\end{bmatrix} \otimes \mathbf{I}_{3\times 3}\right) \begin{bmatrix}\mathbf{0}_{3\times 3} & -\mathbf{G}_1 \\ \mathbf{0}_{3\times 3} & -\mathbf{G}_2 \\ \mathbf{0}_{3\times 3} & -\mathbf{G}_3 \\ \mathbf{I}_{3\times 3} & -\mathbf{0}_{3\times 3}\end{bmatrix} \mathbf{J}_l(\mathbf{a}_j) \tag{4.82}$$

Desarrollando los términos se llega a la siguiente versión más simplificada:

$$\boxed{\frac{\partial \mathbf{r}}{\partial \mathbf{a}_j} = -\begin{bmatrix}\mathbf{R}_{\mathbf{P}_j} & -\mathbf{R}_{\mathbf{P}_j}(\mathbf{p}'_j)^\wedge\end{bmatrix} \mathbf{J}_l(\mathbf{a}_j)} \tag{4.83}$$

Con este cálculo se da por concluida la derivación de ambas versiones de la matriz Jacobiana. Tal y como se ha comentado anteriormente, en §6.2 se realiza un análisis de su eficiencia de cómputo.

Capítulo 5

Seguimiento

En este capítulo se detalla el bloque del sistema encargado de realizar el seguimiento de los objetos y que permite suministrar las variables a optimizar explicadas en §4. Los datos necesarios son una secuencia RGB-D, las máscaras donde se han detectado objetos, sin asociamiento[1] (en nuestro caso obtenidas mediante SiamMask), y una estimación de la localización 3D de las cámara en cada imagen (en nuestro caso extraídas con COLMAP). En la Figura 5.1 se muestran las partes fundamentales.

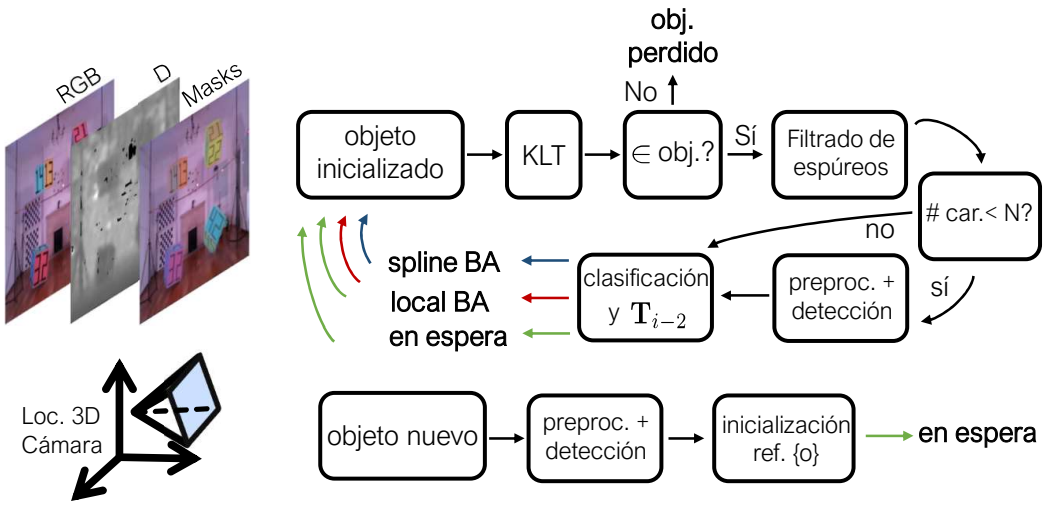


***Figura 5.1: Esquema general del bloque de seguimiento** y datos de entrada.*

En primer lugar, en caso de que los haya, se procesan los objetos ya inicializados (aquellos que han sido seguidos en imágenes previas), asociándolos a máscaras *válidas* de los datos de entrada. Una máscara se considera válida si tiene un tamaño lo suficientemente grande[2]. A continuación, si quedan máscaras sin asociar (libres), se inicializan nuevos objetos basados en éstas. Las técnicas empleadas se detallan a continuación.

[1] Es decir, sin relación con las detecciones de objetos en imágenes previas. Solo se utiliza la información de que en la zona de la imagen con máscara, es probable que se encuentre un objeto cualquiera.

[2] En nuestros experimentos: si cubre más de 1.000 píxeles.

5.1. Detección y seguimiento de características

Para inicializar un objeto, se extraen características Shi-Tomasi [Shi et al., 1994] en la zona cubierta por una de las máscaras libres. En la Figura 5.2 se muestra un ejemplo de extracción, donde los círculos se corresponden con las características. Éstas se corresponden principalmente con esquinas. En cada objeto, se detectan inicialmente N ($N = 100$ en nuestros experimentos) y se trata de mantener este número constante a lo largo de la secuencia. Para la detección, solo se consideran píxeles de los que se dispone de información de profundidad.

Si la posición en la imagen de una característica viene dada por $\mathbf{x} \in \mathbb{R}^2$, su localización 3D, $\mathbf{p}_c \in \mathbb{R}^3$, con respecto al sistema de referencia de la cámara, viene dada por $\mathbf{p}_c = D(\mathbf{x})\mathbf{K}^{-1}\tilde{\mathbf{x}}$, donde $D(\mathbf{x})$ indica su profundidad medida sobre el eje Z y proviene de las imágenes de profundidad de la entrada.

A partir de la localización 3D de todos los puntos detectados, se inicializa el sistema de referencia del objeto con respecto a la cámara, $\mathbf{T}_{co}$, situando su origen en el centro de masas de la nube de puntos y la orientación de sus ejes según sus direcciones principales [Géron, 2019]. Dicho sistema se expresa en la referencia mundo mediante: $\mathbf{T}_{wo} = \mathbf{T}_{wc}\mathbf{T}_{co}$, y los puntos en la referencia objeto mediante $\tilde{\mathbf{p}}_o = \mathbf{T}_{co}^{-1}\tilde{\mathbf{p}}_c$.

Figura 5.2: Detección y seguimiento mediante flujo óptico de características Shi-Tomasi mediante KLT. *Los círculos representan la posición actual de las características detectadas. Las líneas representan la trayectoria que han seguido en la imagen.*

El seguimiento de las características Shi-Tomasi en imágenes consecutivas se lleva

a cabo mediante la implementación piramidal del método iterativo de Lucas-Kanade [Lucas et al., 1981, Bouguet et al., 2001], conocido como **KLT**. En la Figura 5.2 este seguimiento se muestra mediante líneas que indican su trayectoria en la imagen. Una máscara se deja de considerar libre si un determinado porcentaje de características del objeto (50 % en nuestros experimentos) pertenece a ella. Si ninguna máscara cumple este criterio, el objeto se considera perdido.

Cada vez que una característica deja de ser seguida con éxito en la imagen i actual, se realiza una nueva detección en la misma. Para tener una estimación de $\mathbf{p}_o$ (localización 3D en referencia objeto), aplicamos flujo óptico hasta una imagen previa de la que se dispone una estimación de $\mathbf{T}_{wo}$. Si el número de frames en los que se ha seguido el objeto es superior a 3, entonces es necesario aplicar flujo óptico hasta la imagen $i-3$, en otro caso solamente hace falta llegar a la primera imagen.

La zona considerada para la detección, es aquella cubierta por la máscara asociada con información de profundidad. Además, se impone que no se detecten características en un radio de píxeles alrededor de las seguidas con éxito hasta el momento, evitando así características repetidas/ concentradas en una misma zona.

5.1.1. Pre-procesamiento y filtrado de espúreos

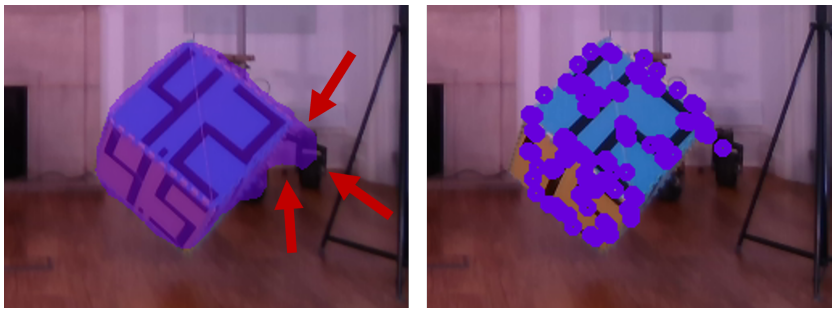
El método de seguimiento explicado anteriormente es susceptible a fallar conforme la secuencia avanza. Por ejemplo, dos situaciones problemáticas son las siguientes:

- *Cámara y objeto presentan giro relativo de cierta magnitud* → La trayectoria de las características que están siendo seguidas se aproximan al borde visible del objeto. Esto provoca una pérdida de información significativa, pues la característica, al dejar de estar rodeada de puntos del objeto, tiende a permanecer en dicho borde, ya que visualmente es la zona más parecida al presentar puntos comunes de la escena estática. Un ejemplo de esto se muestra en la Fig. 5.3.

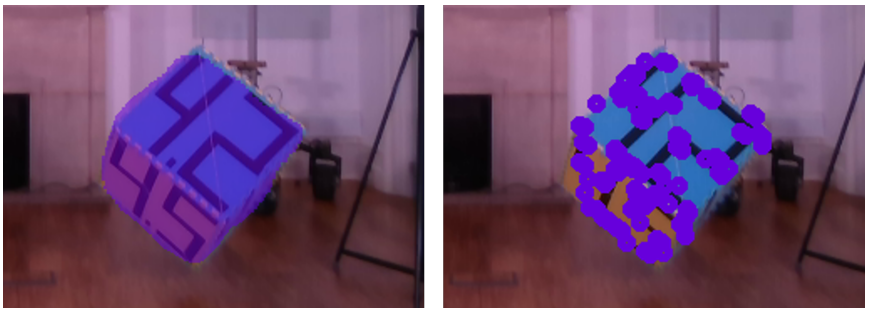
(a) *características iniciales.* (b) *características seguidas.*

***Figura 5.3: Seguimiento de características fallido**. Las características permanecen en el borde visible de los objetos de la derecha al haber experimentado éstos un giro relativo significativo con respecto a la cámara.*

- *Máscaras imperfectas* → Las máscaras de entrada no se ajustan perfectamente a la forma del objeto (ver ejemplos en la Fig. 2.4c), por lo que pueden cubrir zonas que se corresponden con otras entidades, como las de la escena estática. Nuestro sistema podría detectarlas, por tanto, en zonas con textura saliente ajena al objeto. Un ejemplo de esto se muestra en la Fig. 5.4a.



(a) *Detección de características espúreas debido a una máscara de entrada imperfecta.*



(b) *Al rechazar píxeles con valores de profundidad cuya MAD es significativamente alta, se consigue reducir la detección de características espúreas.*

Figura 5.4: Efectos del pre-procesar, o no, las máscaras de entrada.

Ante la presencia significativa de estos **datos espúreos**, necesitamos robustecer a nuestro sistema con el objetivo de evitar considerarlos. Para ello, no podemos recurrir a técnicas habituales como la comprobación de la geometría epipolar (ver Fig. 2.7), ya que los puntos de los objetos no son estáticos. En su lugar, como primera medida comprobamos si el flujo óptico es simétrico, calculándolo de una imagen a la siguiente y de vuelta. Si ambas localizaciones no coinciden (dentro de un margen de error de 0,1 píxeles) la característica se descarta.

Como segunda medida, antes de utilizar una máscara, ésta es pre-procesada: Se extrae la *profundidad* de todos los píxeles que cubre, y se estima su desviación estándar mediante la *desviación absoluta de la mediana* (MAD)[3]. Si denominamos a este conjunto de profundidades como $\mathcal{M} = \{D(\mathbf{x}_1), \ldots, D(\mathbf{x}_n)\}$, el estimador viene dado por:

$$\text{MAD} = \text{med}\left(|D(\mathbf{x}_i) - \text{med}(\mathcal{M})|\right), \quad \text{con } \mathbf{x}_i \in \mathcal{M} \tag{5.1}$$

donde $\text{med} : \mathbb{R}^n \mapsto \mathbb{R}$ es la función que calcula la mediana (valor que separa la mitad superior de los datos con la inferior) de un conjunto de n muestras, y $|\cdot|$ expresa el valor absoluto.

[3]MAD es considerado como el estimador más útil de la escala de una muestra, por aspectos como su alto punto de quiebre (50 %) [Leys et al., 2013, Huber, 2004].

Para ser restrictivos con los valores extremos, se asume que la distribución de la profundidad es Gaussiana. Con esto, el criterio para rechazar píxeles viene dado por:

$$\hat{\sigma}_{\mathcal{M}} \approx 1{,}4826 \text{ MAD}, \quad \rightarrow \quad \text{píxel } i \text{ rechazado si } |D(\mathbf{x}_i) - \text{med}(\mathcal{M})| > 2{,}5\ \hat{\sigma}_{\mathcal{M}} \tag{5.2}$$

Un ejemplo de pre-procesado de la máscara se muestra en la Fig. 5.4b. Apreciándose mejoría con respecto a la situación anterior. Una comparación general en el seguimiento tras incluir las anteriores técnicas se muestra en la Fig. 5.5.

(a) *Antes de incorporar las medidas* (b) *Tras incorporar las medidas*

Figura 5.5: Seguimiento de las características tras 200 imágenes de la secuencia. *En (a) se ha rodeado en rojo características detectadas en el fondo que en (b) no aparecen. Además, el problema de las características acumuladas en el borde visible del objeto se deja de apreciar en (b).*

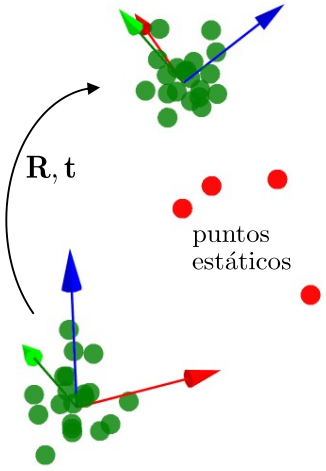
Como última medida para robustecer la propuesta, estimamos, mediante un método basado en MSAC[4] [Torr and Zisserman, 2000], un modelo de rotación, $\mathbf{R}$, y traslación, $\mathbf{t}$, que mejor explica el movimiento experimentado por la nube de puntos del objeto en dos frames consecutivos. Los puntos que no se ajusten a ese modelo son rechazados.

De esta forma, si denominamos a las coordenadas de un punto del objeto expresado en la referencia mundo en el frame i como $\mathbf{y}_j \in \mathbb{R}^3$ y en el $i-1$ como $\mathbf{x}_j \in \mathbb{R}^3$, entonces se selecciona el modelo que presenta un menor valor de C asociado:

$$C = \sum_j \phi\left(\|\mathbf{e}_j\|^2\right), \quad \text{con } \mathbf{e}_j = \mathbf{y}_j - (\mathbf{R}\mathbf{x}_j + \mathbf{t}), \tag{5.3}$$

$$\phi\left(\|\mathbf{e}_j\|^2\right) = \begin{cases} \|\mathbf{e}_j\|^2 & \text{si} \quad \|\mathbf{e}_j\|^2 < T^2, \\ T^2 & \text{si} \quad \|\mathbf{e}_j\|^2 \geq T^2 \end{cases} \tag{5.4}$$

Figura 5.6: *Resultado simulado de aplicar MSAC. los puntos estáticos son rechazados (en rojo), al no ajustarse al modelo $\mathbf{R}, \mathbf{t}$ estimado.*



[4]MSAC [Torr and Zisserman, 2000], es una mejora sobre RANSAC [Fischler and Bolles, 1981], propuesta para no solo penalizar modelos que a priori han sido corrompidos por datos espúreos, sino también estimar cómo de bien un modelo se ajusta a los datos.

donde, asumiendo que $\mathbf{e}_j \sim \mathcal{N}(\mathbf{0}, \sigma\mathbf{I}_3)$, fijamos $T^2 = 7{,}81\sigma^2$ para solo fallar en la predicción de un dato espúreo el 5 % de las veces [Hartley and Zisserman, 2004].

Cada modelo se estima mediante minimización de $\|\mathbf{e}_j\|^2$, utilizando para ello 3 parejas de puntos, $\{\mathbf{y}_j, \mathbf{x}_j\}$, seleccionadas aleatoriamente. Este es el mínimo número que permite una estimación de $\mathbf{R}$ y $\mathbf{t}$, la cual obtenemos mediante el método de alineamiento por mínimos cuadrados de [Umeyama, 1991]. Dado que la presencia de espúreos es esperable que no sea alta tras las medidas tomadas anteriormente, consideramos un total de ~ 30 modelos diferentes. Un resultado simple de aplicar este método se muestra en la Fig. 5.6.

5.2. Clasificación

Como última parte, se clasifica cada objeto para determinar qué acción realizar sobre él. Esta decisión se basa en el número de observaciones disponibles sobre las que realizar la optimización. Como éstas determinan el número de filas de la matriz Jacobiana explicada en §4.2, es necesario disponer de observaciones suficientes para evitar que la matriz Hessiana se vuelva singular y por tanto no se pueda realizar la optimización mediante el algoritmo de Gauss-Newton.

En concreto, cada observación $\mathbf{p}_c \in \mathbb{R}^3$ aporta 3 filas (al tener 3 dimensiones) a la matriz Jacobiana. Por otro lado, si solo se están optimizando puntos de control (*spline BA*), cada uno aporta 6 variables a optimizar. Si además se optimizan puntos del objeto (*local BA*), se añaden 3 variables más por cada uno. Así mismo, tal y como se comentó en §4.1, necesitamos que un objeto haya sido observado en al menos 4 frames consecutivos para llevar a cabo la optimización.

Por ello, en la ventana temporal explicada en §4.1, si denominamos al número de observaciones $\mathbf{p}_c$ como k, al número de puntos de control a optimizar como m, y al número de puntos del objeto a optimizar como n, la clasificación de un objeto cuando se ha seguido por un número de frames, f, mayor o igual que 4, se realiza de la siguiente forma:

$$\text{clasificación}_{f \geq 4} \begin{cases} \text{objeto } \textit{perdido} & \text{si} \quad 3k \leq 6n \\ \text{optimizar con } \textit{spline BA} & \text{si} \quad 6n < 3k \leq 6n + 3m \\ \text{optimizar con } \textit{local BA} & \text{si} \quad 3k > 6n + 3m \end{cases} \tag{5.5}$$

En cambio, si $f < 4$, el objeto entra en *espera* → Su optimización no empieza hasta la llegada de nuevas imágenes. Tras llevar a cabo la clasificación, salvo que el objeto haya sido clasificado como *perdido*, se inicializa el siguiente punto de control $\mathbf{T}_{i-2}$ según lo explicado en §4.1.

Capítulo 6

Experimentos y Resultados

En este capítulo se presentan los experimentos que ponen a prueba la propuesta y que sirven de motivación de la misma. En primer lugar se cuantifica cuantitativa y visualmente la importancia de tener en cuenta los objetos dinámicos en la estimación de la localización del sensor. A continuación, se evalúa el coste computacional del cálculo de las matrices Jacobianas de las alternativas empleadas en la literatura y de nuestra propuesta. Tras esto, se compara el error en la estimación de localización y velocidad de un objeto, según si su estimación es en tiempo discreto o en tiempo continuo (nuestra propuesta). Finalmente, evaluamos este trabajo en una base de datos pública y comparamos los resultados obtenidos con el estado del arte.

6.1. COLMAP en entorno dinámico

En este primer experimento evaluamos el impacto de considerar, o no, los objetos dinámicos en la estimación de la localización y orientación (pose) del sensor. En ambas situaciones, para estimarla empleamos COLMAP, un software del estado del arte en SfM, presentado más en detalle en §2.4.

La secuencia RGB-D empleada es `swinging_4_unconstrained` de la base de datos pública OMD (*Oxford Multimotion Dataset*) [Judd and Gammell, 2019]. La componente dinámica de esta secuencia, grabada en una habitación, la aportan 4 cajas que se balancean mediante un mecanismo de poleas (ejemplos de imágenes se encuentran en las Figs. 5.2 - 5.5). El movimiento de la cámara que las graba es libre (presenta rotación y traslación), y es calculado mediante un sistema de captura de movimiento (*Vicon*)[1].

El error en la estimación se calcula tomando el movimiento capturado por la Vicon como el verdadero. De esta forma comparamos las estimaciones dadas por COLMAP $\mathcal{E}_{\text{colmap}} = \{\mathbf{T}_1, \ldots, \mathbf{T}_n\}$ en cada instante, con las del instante más próximo de la Vicon,

[1]En concreto los autores hacen uso de un producto de la marca `Vicon`, contando con múltiples cámaras infra-rojas colocadas en puntos estratégicos de la habitación para detectar marcadores colocados sobre las cajas..

$\mathcal{E}_{\text{vicon}} = \{\mathbf{P}_1, \ldots, \mathbf{P}_n\}$, al tener este último una frecuencia de captura elevada (~ 7 veces más alta que el sensor RGB-D). En esta sección se asume que las poses $\in \mathcal{E}_{\text{colmap}}, \mathcal{E}_{\text{vicon}}$ transforman puntos de la referencia cámara a su referencia global.

Antes de cuantificar el error, es necesario alinear la trayectoria estimada con la verdadera. Esto es debido a que, a pesar de que ambas expresan la pose del sistema de referencia de la cámara (con origen en el centro óptico y ejes según la Fig. 2.2), éstas están expresadas en sistemas de referencia globales diferentes. Por un lado, la trayectoria estimada está expresada con respecto al sistema de referencia de la cámara en la primera imagen, mientras que la verdadera está expresada en nuestro caso en el sistema de referencia de la Vicon.

Este alineamiento de trayectorias es común realizarlo con el método de Umeyama [Umeyama, 1991], mediante el cual se minimiza la diferencia cuadrática de la componente *translacional*. Es decir, es la transformación $(s, \mathbf{R}, \mathbf{t})$ que minimiza la siguiente cantidad:

$$\frac{1}{n}\sum_{i=1}^{n} \left(\text{transl}(\mathbf{P}_i) - s\,\mathbf{R}\,\text{transl}(\mathbf{T}_i) + \mathbf{t}\right)^2 \tag{6.1}$$

Donde transl$(\cdot)$ lo utilizamos para expresar que solo tenemos en cuenta la componente translacional de la matriz de transformación. De aquí en adelante, en esta sección, consideramos que $\mathcal{E}_{\text{colmap}}$, $\mathcal{E}_{\text{vicon}}$ han sido alineadas según este método.

Para cuantificar el error en una trayectoria, en la literatura es común emplear 2 tipos de métricas diferentes. La primera de ellas es el Error Absoluto en la Pose (**APE**) [Sturm et al., 2012b, Zhang and Scaramuzza, 2018], la cual está basada en la pose relativa (error) entre la estimada y la verdadera. Para un instante i ésta viene dada por:

$$\mathbf{E}_i = \mathbf{P}_i^{-1}\,\mathbf{T}_i \tag{6.2}$$

Dicha pose relativa se calcula para todos los instantes y comúnmente [Sturm et al., 2012b], se calcula el error cuadrático medio (RMSE) de la parte translacional:

$$\text{APE} = \left(\frac{1}{n}\sum_{i=1}^{n} \|\text{transl}(\mathbf{E}_i)\|^2\right)^{0,5} \tag{6.3}$$

Al emplear los valores absolutos de las poses, APE sirve para evaluar la *consistencia global* de la trayectoria estimada. Para evaluar la *consistencia local* se emplea el Error Relativo en la Pose (**RPE**), el cual esta basado en la diferencia del incremento entre poses de las trayectorias. Para un instante i, esta diferencia viene dada por:

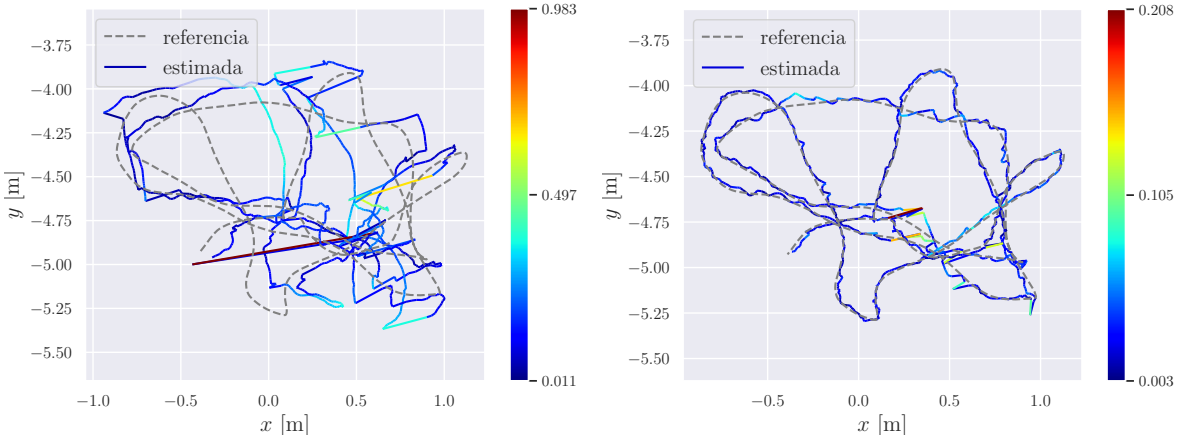
$$\mathbf{F}_i = (\mathbf{P}_i^{-1}\mathbf{P}_{i+\Delta})^{-1}\,(\mathbf{T}_i^{-1}\mathbf{T}_{i+\Delta}), \tag{6.4}$$

donde $\Delta \in \mathbb{N}^+$ expresa el número de imágenes de diferencia entre las que se calcula la pose relativa en cada trayectoria. En esta evaluación, usamos $\Delta = 1$. Igualmente, es común emplear el RMSE para cuantificar el error, no solo aplicándolo a la translación, sino también a la rotación:

$$\text{RPE}_{\text{transl}} = \left(\frac{1}{n}\sum_{i=1}^{n} \|\text{transl}(\mathbf{F}_i)\|^2\right)^{0,5}, \quad \text{RPE}_{\text{rot}} = \left(\frac{1}{n}\sum_{i=1}^{n} \|\text{Log}\left(\text{rot}(\mathbf{F}_i)\right)\|^2\right)^{0,5} \tag{6.5}$$

Sin tener en cuenta a los objetos dinámicos			*Teniendo en cuenta a los objetos dinámicos*		
APE [m]	**RPE [m/im.]**	**RPE [°/im.]**	**APE [m]**	**RPE [m/im.]**	**RPE [°/im.]**
0.202	0.063	0.775	0.033	0.026	0.693

Tabla 6.1: Valores de APE y RPE teniendo en cuenta o no a los objetos dinámicos *para estimar la pose del sensor. Se observa mejoría en todas las métricas si se impone que en la zona de la imagen en la que se encuentran los objetos dinámicos no se extraigan características salientes.*



(a) *Sin tener en cuenta los objetos dinámicos* (b) *Teniendo en cuenta los objetos dinámicos*

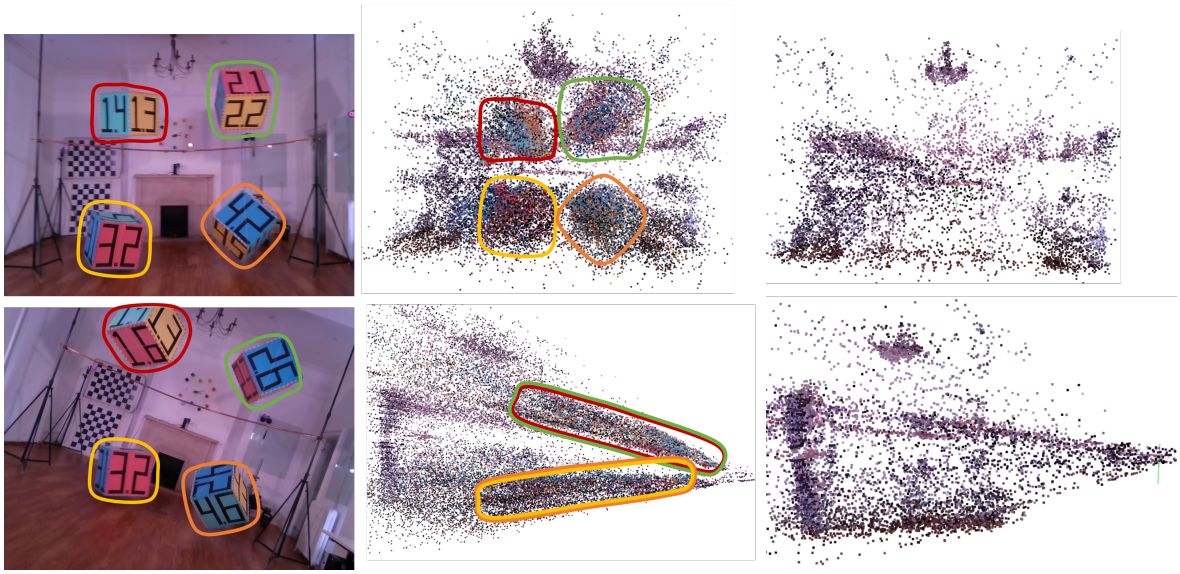
Figura 6.1: Comparación visual de trayectorias estimadas teniendo en cuenta o no a los objetos dinámicos*. Se observa como en (b) la trayectoria estimada es significativamente más próxima a la de referencia (verdadera). El color indica el valor de APE [m] en cada punto de la trayectoria según el código de colores de la derecha.*

En la tabla 6.1 se muestran los valores de las anteriores métricas obtenidos tras emplear 1.000 frames imágenes de la secuencia. El método que tiene en cuenta los objetos dinámicos, consiste en no extraer características salientes en las zonas de la imagen en la que éstos se encuentran (determinadas mediante SiamMask, §2.3). Además, en la Fig. 6.1 se realiza una comparación visual de las trayectorias estimadas con la trayectoria verdadera[2].

A partir de estas comparaciones, concluimos que el tener en cuenta a los objetos dinámicos marca una diferencia significativa en la precisión de la localización del sensor en aquellas secuencias donde la componente dinámica es significativa.

Finalmente, en la Fig. 6.2 se muestran los mapas de puntos estáticos creados con ambos métodos. A través de éstos se entiende la pérdida de precisión, ya que si no se tienen en cuenta a los objetos dinámicos puede que las características asociadas a éstos no pasen el filtrado de espúreos, quedando reflejados en el mapa a la vez que corrompiendo las estimaciones de la pose de la cámara al romper la asunción de estaticidad del mapa.

[2] Ambos resultados han sido obtenidos utilizando el paquete de Python comparación de trayectorias `evo`.



(a) (b) (c)

Figura 6.2: Comparación visual del mapa estático creado teniendo en cuenta, o no, los objetos dinámicos. *(a) 2 imágenes de la secuencia. (b) Mapa estático reconstruido sin tener en cuenta a los objetos dinámicos. Es posible apreciar como parte de estos aparecen en la reconstrucción final (especialmente en las zonas rodeadas). (c) Mapa estático reconstruido teniendo en cuenta los objetos dinámicos. Al no extraer características de la zona en la que se éstos se encuentran, aspectos de la reconstrucción mejoran: por ejemplo, se aprecian detalles más finos en el candelabro, o menos puntos situados en donde no debería haber nada.*

6.2. Tiempo de cómputo de las matrices Jacobianas

En este segundo experimento, realizamos un análisis comparativo del tiempo de cómputo necesario para calcular las matrices Jacobianas derivadas en §4.3 y lo comparamos con el que necesitan las técnicas habituales empleadas en la literatura. Para el cálculo de tiempos se emplea `perfplot`, un paquete de Python especializado en este tipo de análisis.

En la literatura encontramos dos métodos alternativos que evitan el cómputo analítico de las matrices Jacobianas. En primer lugar, el más utilizado es el de la **diferenciación automática** empleada por [Lovegrove et al., 2013, Patron-Perez et al., 2015, Mueggler et al., 2015, Kim et al., 2016b, Mueggler et al., 2018, Yang et al., 2021] entre otros. El otro método, consiste en hacer uso de **diferenciación numérica**, la cual se empleó en [Kerl et al., 2013].

A modo de breve explicación de ambos métodos, con la diferenciación numérica se obtiene una *aproximación* de la matriz Jacobiana (de dimensiones $m \times n$), pues cada elemento de la fila i y columna j es calculado según:

$$\mathbf{J}_{\mathbf{r}_i}(\mathbf{x}_j) = \frac{\partial \mathbf{r}_i(\mathbf{x})}{\partial \mathbf{x}_j} \approx \frac{\mathbf{r}_i(\mathbf{x} + h\mathbf{e}_j) - \mathbf{r}_i(\mathbf{x} - h\mathbf{e}_j)}{2h} \tag{6.6}$$

Donde $\mathbf{e}_j$ es el j-ésimo vector unitario (de n dimensiones) que indica la variable de estado que se incrementa (y decrementa) por una magnitud pequeña $h > 0$. Este

método es considerado como *inestable* y *mal condicionado* por la introducción de errores de truncamiento y redondeo [Baydin et al., 2018].

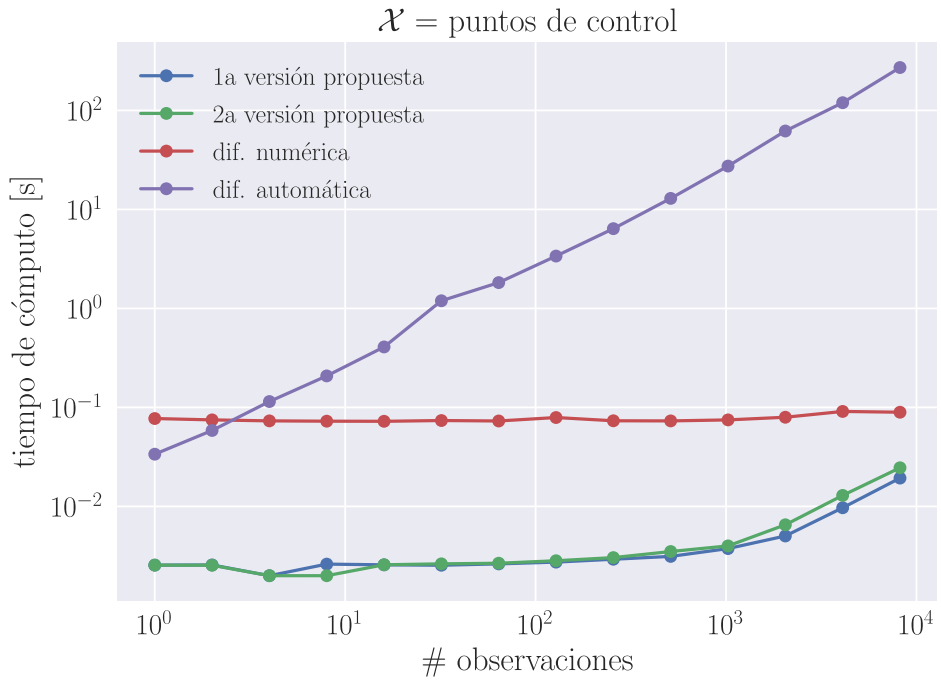


Figura 6.3: Análisis de tiempo de cómputo de una matriz Jacobiana. *Se aprecia como las versiones propuestas en §4.3, necesitan una cantidad de tiempo significativamente menor que las alternativas (diferenciación numérica y automática). En este análisis se consideran las derivadas con respecto a 3 puntos de control.*

Por otro lado, la diferenciación automática calcula las derivadas parciales de cada elemento o *nodo* de un grafo computacional que representa a la función de interés (en nuestro caso $\mathbf{r}_{\mathbf{p}_c}(\mathbf{x})$) con respecto a las entradas ($\boldsymbol{\xi}$). Aplicando la regla de la cadena, el valor de la derivada se propaga a través de los nodos que conforman el grafo. Un análisis detallado de la técnica se encuentra en [Baydin et al., 2018]. A diferencia del método anterior, la diferenciación automática ofrece resultados exactos. Para este análisis se usó la implementación de `Autograd` [Maclaurin et al., 2015].

Los resultados se muestran en la Figura 6.3. Se aprecia como el cálculo analítico de las matrices Jacobianas ofrece ventajas en tiempo de cómputo con respecto a las alternativas. En nuestro caso de interés, el número de observaciones se suele situar en $\sim 10^3$, suponiendo por tanto, de forma aproximada, 1 y 3 órdenes de magnitud menos de tiempo que la diferenciación numérica y automática, a la vez que ofreciendo resultados exactos (sujetos a la precisión numérica del ordenador). Así mismo se aprecia como la primera versión de las propuestas en §4.3 escala mejor conforme el número de observaciones aumenta.

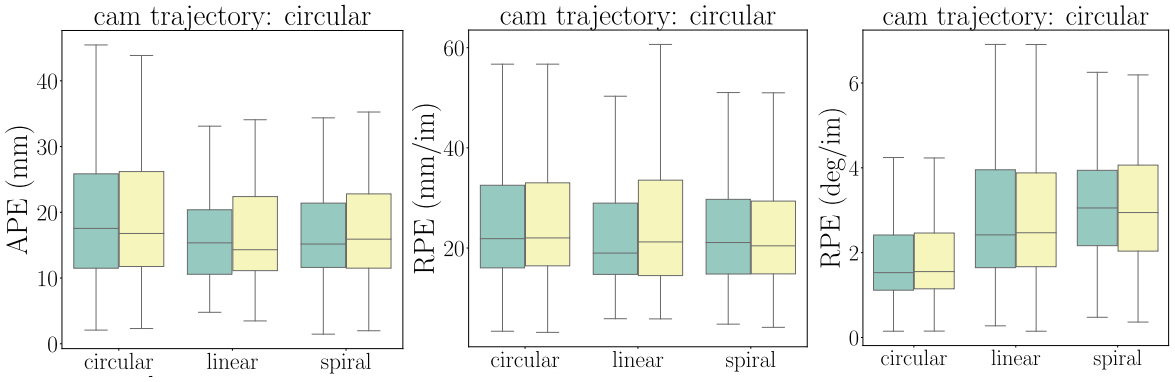
6.3. Estimación en tiempo continuo vs discreto

En este experimento, a partir de datos sintéticos, se compara la precisión de la estimación de la localización y orientación (pose) y velocidad de un objeto según si su

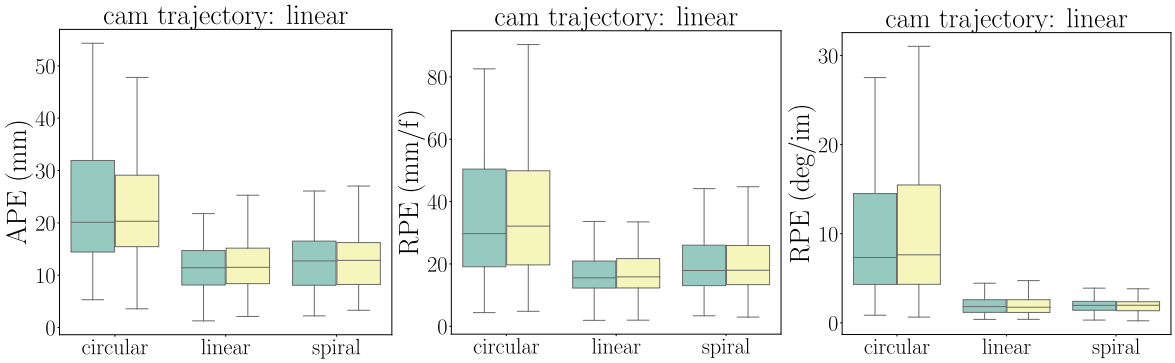
estimación se realiza en tiempo discreto, o si ajustamos su trayectoria a una curva en tiempo continuo con B-Splines cumulativos (nuestra propuesta).

Para la comparación de la precisión en la estimación de las poses del objeto, se crearon tres tipos de trayectorias que involucran diferentes grados de libertad: *lineal*, *circular* y en *espiral* (ver Fig 6.5). En la trayectoria circular el objeto experimenta una rotación alrededor su eje $\hat{x}_o$ y otra alrededor del eje $\hat{z}_w$ global (completando 2 giros). En las trayectorias lineal y en espiral, el objeto experimenta rotación en sus tres ejes. El sensor RGB-D simulado sigue una trayectoria lineal o circular.

Las estimaciones en tiempo discreto de las poses del objeto se obtienen igualmente a través de la minimización de la ecuación 4.1, solo que en este caso se optimiza directamente la pose $\mathbf{T}_{wo}$ del objeto (en lugar de optimizar los puntos de control de la trayectoria interpolada $\mathbf{T}_{wo}(t)$). Al igual que con la estimación en tiempo continuo, se optimizan los puntos del objeto cuando existen suficientes observaciones en la ventana temporal.



(a) *Sensor con trayectoria circular.*



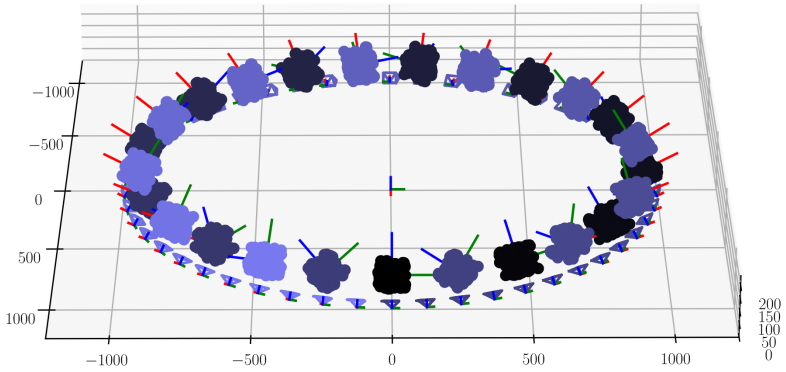
(b) *Sensor con trayectoria lineal.*

***Figura 6.4: Diagramas de caja con los valores de APE y RPE. Verde**: estimación en tiempo continuo. **Amarillo**: estimación en tiempo discreto. El eje horizontal indica el tipo de trayectoria seguida por el objeto.*

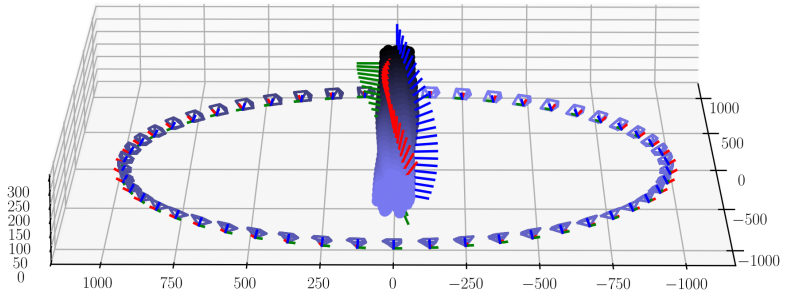
Cada secuencia artificial creada es de 100 imágenes. Para simular unos datos reales, se introduce un ruido Gaussiano de varianza $\sigma^2_{\text{px}} = 1$ píxel^2 en las coordenadas en la imagen de las características salientes y otro ruido Gaussiano con $\sigma^2_{\text{d}} = 10$ mm^2 en los valores de profundidad de los puntos observados por el sensor RGB-D.

En la Figura 6.4 se muestran los diagramas de caja de los valores obtenidos de

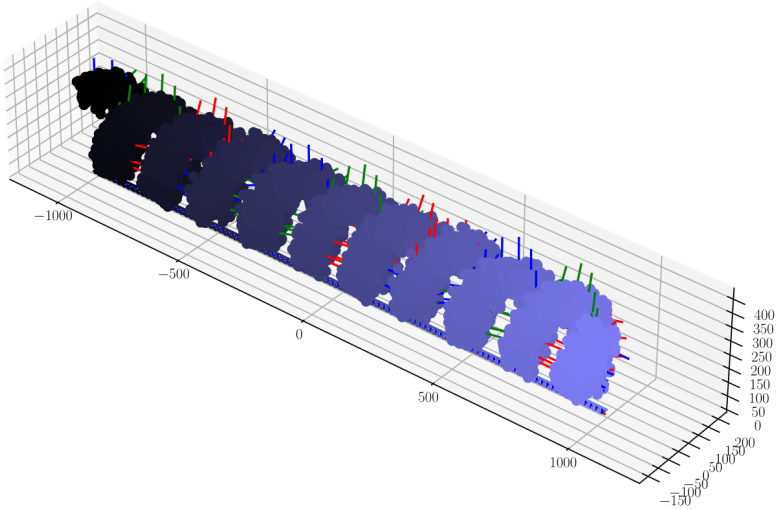
APE y RPE en cada imagen (cada término de los sumatorios de las definiciones dadas en §6.1), según la trayectoria del sensor y del objeto. En ellos se observa como la consistencia global y local son similares, es decir, a pesar de estar imponiendo un modelo de trayectoria capaz de ofrecer estimaciones de velocidad y aceleración (y pose) continuas para cualquier instante temporal, ésta converge a una solución con error en posición y orientación similar.



(a) *Cámara y objeto: ambos trayectoria circular.*



(b) *Cámara: trayectoria circular. Objeto: trayectoria lineal.*



(c) *Cámara: trayectoria lineal. Objeto: trayectoria espiral.*

***Figura 6.5: Ejemplos de trayectorias evaluadas en este experimento**. Solo se representa la nube de puntos asociada al objeto. Su color indica el instante temporal asociado al mismo: un color más oscuro indica más antigüedad. Solo se representan parte de las cámaras/objetos para evitar que las cámaras y los ejes del objeto se mezclen.*

La comparación de la precisión de las estimaciones de **velocidad** (lineal y angular) en tiempo discreto vs continuo, se realizó a través de múltiples trayectorias en las que el objeto gira alrededor del eje $\hat{z}_w$ global (describiendo un círculo), a la vez que experimenta un giro sobre su propio eje $\hat{x}_o$, *dependiendo de dos grados de libertad*: θ_{tranls}, θ_{rot}.

Ambos parámetros determinan magnitudes angulares entre 2 instantes consecutivos de los que se conoce la pose del objeto: El primero, θ_{transl}, determina el incremento sobre $\hat{z}_w$, mientras que θ_{rot} determina el incremento en $\hat{x}_o$ (ver Figura 6.6). La motivación de esta elección reside en evaluar la robustez de las estimaciones tanto en movimientos suaves ($\theta_{\text{transl}}, \theta_{\text{rot}}$ bajos) como bruscos ($\theta_{\text{transl}}, \theta_{\text{rot}}$ altos).

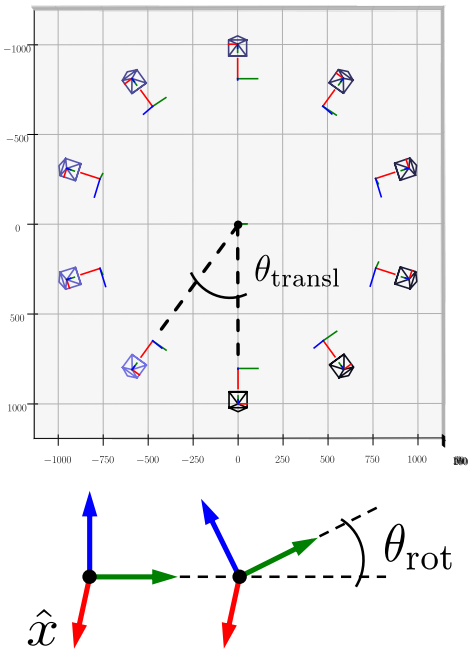


Figura 6.6: Definición de los grados de libertad de las trayectorias generadas. *θ_{transl} y θ_{rot}, representan respectivamente: el incremento angular medido sobre el eje $\hat{z}_w$ global y el medido sobre el eje $\hat{z}_o$ del objeto, entre dos poses consecutivas.*

Al solo estar interesados en evaluar las estimaciones de velocidad, la localización y orientación de los sistemas de coordenadas del objeto se consideran conocidos: En el caso discreto coinciden con los verdaderos, y en el caso continuo se define cada punto de control asociado a un instante i, $\mathbf{T}_i$, con la pose del objeto en el instante $i+2$[3].

El cálculo de las velocidades asociadas la curva B-Spline cumulativa se realiza a través de las Ecs. 3.75 y 3.22 para cualquier instante temporal t. En el caso discreto, asumimos que la velocidad es constante entre dos poses consecutivas, pudiéndose calcular de dos formas (ambas son evaluadas):

- *Considerar acoplamiento en la traslación y rotación*: En este caso, tal y como se vio en las Ecs. 3.39 y 3.26, sabemos que el incremento entre dos poses consecutivas

[3]Siguiendo así la intuición comentada en §4.1 y visualizada en la Fig. 3.6 de que en el cálculo de la pose del objeto en instante i, el punto de control de mayor peso es $\mathbf{T}_{i-2}$.

$\mathbf{T}_{wo1}$, $\mathbf{T}_{wo2}$, viene dado por:

$$\mathbf{T}_{wo2} = \mathbf{T}_{wo1} \exp(\boldsymbol{\tau}_o^{\wedge} \Delta t) \quad \Rightarrow \quad \boldsymbol{\tau}_o^{\wedge} = \frac{1}{\Delta t} \log(\mathbf{T}_{wo1}^{-1} \mathbf{T}_{wo2}) \tag{6.7}$$

Donde $\boldsymbol{\tau}_o = [\mathbf{v}_o, \boldsymbol{\omega}_o]^T$ contiene la velocidad lineal $\mathbf{v}_o$ y angular $\boldsymbol{\omega}_o$, constantes y en la referencia del objeto, que éste experimenta durante el incremento temporal entre poses Δt (con $t_1 \leq t < t_2 = t_1 + \Delta t$)[4].

- *Considerar traslación y rotación desacopladas*: En este caso, la velocidad angular coincide con el cálculo anterior:

$$\boldsymbol{\omega}_o^{\wedge} = \frac{1}{\Delta t} \log(\mathbf{R}_{wo1}^T \mathbf{R}_{wo2}) \tag{6.8}$$

Mientras que para $\mathbf{v}_o$, se considera interpolación lineal:

$$\mathbf{v}_o = \frac{1}{\Delta t} \mathbf{R}_{wo_1}^T (\mathbf{t}_{wo2} - \mathbf{t}_{wo1}) \tag{6.9}$$

Este acercamiento es el seguido por trabajos del estado del arte como [Zhang et al., 2020, Bescos et al., 2021].

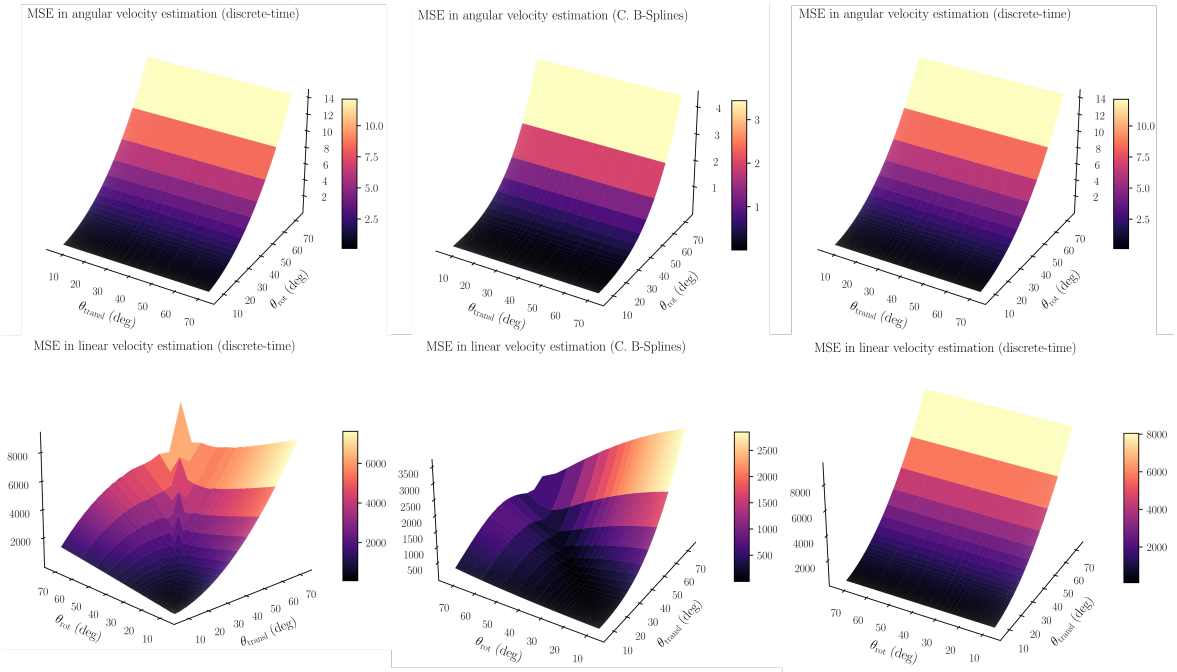


Figura 6.7: MSE en función de θ_{transl} y θ_{rot}. *Col. izquierda: Estimación en tiempo discreto con rotación y translación acopladas. Col. central: Estimación en tiempo continuo. Col. derecha: Estimación en tiempo discreto con rotación y translación desacopladas.*

Para evaluar las estimaciones, empleamos el error cuadrático medio (MSE), es decir, denominando a cada una de las n estimaciones en un instante i como $\mathbf{v}_{\text{est},i}$, $\boldsymbol{\omega}_{\text{est},i}$ y a los valores verdaderos (calculados analíticamente) como $\mathbf{v}_{\text{gt},i}$, $\boldsymbol{\omega}_{\text{gt},i}$, entonces:

$$\text{MSE}_{\mathbf{v}} = \frac{1}{n} \sum_{i=1}^{n} \|\mathbf{v}_{\text{est},i} - \mathbf{v}_{\text{gt},i}\|^2, \qquad \text{MSE}_{\boldsymbol{\omega}} = \frac{1}{n} \sum_{i=1}^{n} \|\boldsymbol{\omega}_{\text{est},i} - \boldsymbol{\omega}_{\text{gt},i}\|^2 \tag{6.10}$$

[4]Si se quiere expresar dichas velocidades en la referencia mundo, basta con aplicarles la transformación: $\mathbf{R}_{wo}\mathbf{v}_o$, $\mathbf{R}_{wo}\boldsymbol{\omega}_o$.

Los resultados para las diferentes estrategias se muestran en la Figura 6.7. Se aprecia como, a pesar de la aproximación realizada en la colocación de los puntos de control, la estimación en tiempo continuo ofrece errores más bajos en todo el rango de valores de θ_{rot} y θ_{transl} evaluados. Esto tiene sentido ya que con la estimación mediante B-Splines Cumulativos cúbicos no tenemos la restricción de asumir velocidades constantes entre poses consecutivas del objeto.

6.4. Evaluación en base de datos pública

En este último experimento evaluamos el sistema propuesto completo y nos comparamos con el estado del arte en la secuencia `swinging_4_unconstrained` de la base de datos OMD [Judd and Gammell, 2019], utilizada en §6.1 para destacar la necesidad de tener en cuenta a los objetos dinámicos en la estimación de la localización del sensor y creación del mapa estático. En este caso, nos centramos en el error de las **trayectorias estimadas de los objetos**.

Siguiendo el protocolo de los trabajos del estado del arte [Judd et al., 2018, Zhang et al., 2020, Huang et al., 2020, Bescos et al., 2021], empleamos los primeros 500 frames de la secuencia. En cuanto a las métricas para comparar las trayectorias estimadas con las verdaderas, empleamos nuevamente el APE y RPE introducidos en §6.1, y además los errores máximos de la componente translacional.

En este caso, el alineamiento necesario para las trayectorias difiere del explicado en §6.1. Esto es debido a que el origen del sistema de referencia escogido para cada objeto puede diferir arbitrariamente del utilizado en la trayectoria verdadera, al no emplearse ninguna convención como con los sistemas de referencia de una cámara (Fig. 2.2). Esta diferencia, de manera visual aparece en la Figura 6.8.

De esta forma, contamos con dos trayectorias de dos sistemas de referencia distintos $\{o_1, o_2\}$, expresadas en dos sistemas de referencia globales diferentes $\{w_1, w_2\}$. Para subsanar este inconveniente, y siguiendo las recomendaciones de los autores de la base de datos[5] llevamos a cabo dos transformaciones:

- **1er Alineamiento**: Imponemos que la primera pose de la trayectoria estimada y la verdadera coincidan. Es decir, si el conjunto de poses estimadas de un objeto vienen dadas por $\mathcal{E}_{\text{est}} = \{\mathbf{T}^1_{w_1o_1}, \ldots, \mathbf{T}^n_{w_1o_1}\}$, y el conjunto de poses verdaderas asociadas viene dada por $\mathcal{E}_{\text{gt}} = \{\mathbf{T}^1_{w_2o_2}, \ldots, \mathbf{T}^n_{w_2o_2}\}$, a cada pose, $\mathbf{T}^i_{w_1o_1} \in \mathcal{E}_{\text{est}}$ le aplicamos la siguiente transformación:

  $$\mathbf{T}^1_{w_2w_1}\mathbf{T}^i_{w_1o_1} \tag{6.11}$$

  donde $\mathbf{T}^1_{w_2w_1}$ es la pose de la cámara en la imagen $i = 1$ desde la referencia de la Vicon, ya que ésta se toma como sistema de referencia mundo de la trayectoria estimada.

- **2o Alineamiento**: La anterior transformación sirve de alineamiento de las referencias globales. Falta alinear las referencias de los objetos. Para ello, llevamos a

[5] `https://github.com/robotic-esp/dataset_tools/issues/3`

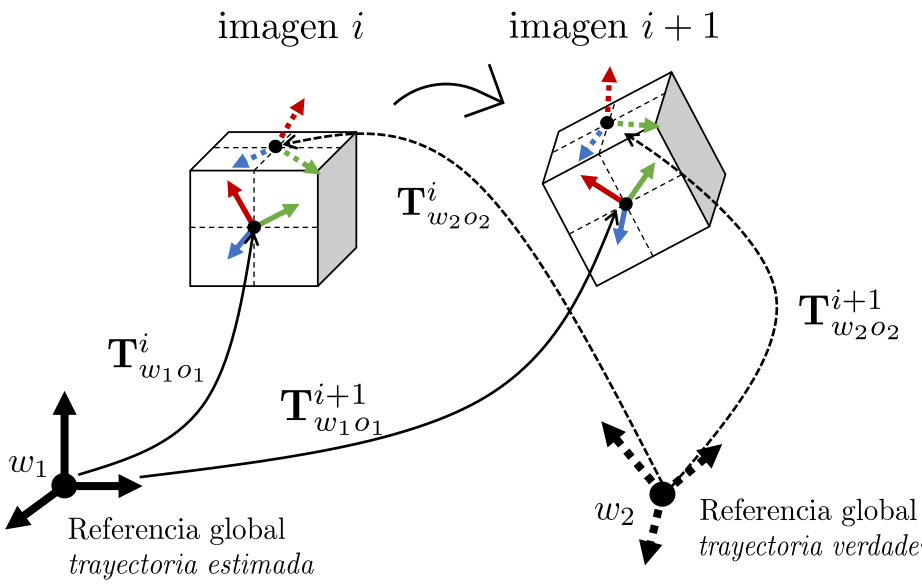


***Figura 6.8: Esquema de referencias de la trayectoria estimada - verdadera**. Debido a que no existe una convención para definir el sistema de referencia de un objeto, contamos con un par de trayectorias de dos sistemas de referencia distintos, tanto en localización como en orientación: $\{o_1, o_2\}$, expresadas desde dos sistemas de referencia globales diferentes $\{w_1, w_2\}$.*

cabo la siguiente optimización:

$$\mathbf{T}_{o_1o_2} = \arg\min_{\mathbf{T}_{o_1o_2}} \sum_{i=1}^{n} \left[\mathrm{Log}\left(\mathbf{T}^{i}_{w_2o_1}\mathbf{T}_{o_1o_2}(\mathbf{T}^{i}_{w_2o_2})^{-1}\right)\right]^2 \tag{6.12}$$

Por lo que para alinear finalmente cada pose, $\mathbf{T}^{i}_{w_2o_1}$, aplicamos la siguiente transformación:

$$\mathbf{T}^{i}_{w_2o_1}\mathbf{T}_{o_1o_2} \tag{6.13}$$

Con las poses de la trayectoria estimada alineadas con la trayectoria verdadera calculamos las métricas comentadas anteriormente. En las Tablas 6.3 y 6.2 se muestran los resultados obtenidos por nuestra propuesta, así como los reportados por los trabajos del estado del arte en sus respectivas publicaciones.

Sistema	Caja 1	Caja 2	Caja 3	Caja 4
[Judd et al., 2018]	0,36	0,64	0,45	5,94
[Judd et al., 2018] (act.)	0,44	**0,27**	0,99	0,39
[Huang et al., 2020]	**0,24**	0,45	**0,24**	4,69
Nuestro con Local BA	0,39	0,30	0,77	0,47
Nuestro sin Local BA	0,29	0,38	0,27	**0,32**

***Tabla 6.2: Comparación con MVO [Judd et al., 2018] y ClusterVO [Huang et al., 2020]** de los errores de translación máximos en la trayectoria [m]. Los mejores resultados son marcados en negrita. De MVO se reportan dos versiones, la aceptada en la revista RA-L, y una versión actualizada en la que los autores reportan ciertas mejoras. Se observa como nuestro sistema ofrece resultados similares, siendo el mejor para una de las cajas.*

De nuestra propuesta se reportan los resultados de emplear Spline BA + Local BA (según la clasificación 5.5) y de emplear tan solo Spline BA. Esta última opción, es atractiva por su menor coste computacional (menos variables a optimizar), teniendo como desventaja el no poder corregir el error en la localización de los puntos. Sin embargo, esta opción presenta además menor error en algunas de las trayectorias, es decir, optimizar la localización de los puntos del objeto puede conllevar un mayor error en la estimación de la trayectoria. Por esta razón, sería interesante investigar más medidas para la detección de datos espúreos.

Sistema	Caja 1			Caja 2			Caja 3			Caja 4		
	APE (m)	RPE (m/f)	RPE (º/f)	APE (m)	RPE (m/f)	RPE (º/f)	APE (m)	RPE (m/f)	RPE (º/f)	APE (m)	RPE (m/f)	RPE (º/f)
[Zhang et al., 2020]	-	**0,030**	**1,01**	-	**0,023**	**1,36**	-	**0,029**	**1,64**	-	**0,026**	**1,75**
[Bescos et al., 2021]	0,41	-	-	0,37	-	-	1,09	-	-	0,28	-	-
Nuestro con Local BA	0,16	0,061	6,01	**0,18**	0,038	5,00	0,37	0,041	4,59	0,38	0,040	3,19
Nuestro sin Local BA	**0,12**	0,051	6,29	0,19	0,043	5,90	**0,12**	0,051	6,05	**0,21**	0,043	4,25

Tabla 6.3: Comparación con VDO-SLAM [Zhang et al., 2020] y DynaSLAM II [Bescos et al., 2021]. *Los mejores resultados son marcados en negrita. Nuestra propuesta presenta mejores resultados en cuanto a APE, mientras que VDO-SLAM presenta mejores valores de RPE.*

Capítulo 7

Conclusiones

En este TFM se ha presentado un sistema para estimar el movimiento de los objetos dinámicos presentes en una escena a través de un sensor RGB-D y en tiempo continuo. En concreto, se ha propuesto ajustar las trayectorias seguidas por los objetos a curvas B-Spline Cumulativas cúbicas, un tipo de curva que entre otras propiedades, presenta la ventaja de ofrecer estimaciones continuas de posición y orientación, velocidad y aceleración para cualquier instante temporal.

Bajo nuestro conocimiento, tras repasar los trabajos relacionados, este acercamiento no había sido explorado a través de sensores visuales antes en la literatura, diferenciándonos así de los trabajos anteriores, los cuales realizan estimaciones en tiempo discreto (solo para los instantes de los que se disponen de datos sensoriales).

Especial atención se ha prestado a la derivación analítica de las matrices Jacobianas necesarias para la optimización. Hasta la fecha, los trabajos relacionados de la literatura emplean técnicas computacionalmente costosas (diferenciación automática) y/o aproximadas (diferenciación numérica). De esta forma, esperamos que esta contribución pueda servir de ayuda para futuros trabajos que usen este tipo de curvas.

Nuestro sistema se ha evaluado de diferentes formas. En primer lugar, se ha mostrado de manera cuantitativa y cualitativa la necesidad de tener en cuenta a los objetos dinámicos en aquellas escenas donde su presencia es dominante.

A continuación, se han comparado las estimaciones en tiempo discreto y continuo: Por un lado, a pesar de estar imponiendo un modelo de trayectoria, se ha comprobado que para distintos tipos de movimientos de cámara y objetos, esto no supone una reducción en la precisión de la estimación de la orientación y posición. Por otro, al no estar asumiendo un modelo de velocidad constante entre imágenes consecutivas, esto ha conllevado a unos mejores resultados en cuanto a estimación de la velocidad lineal y angular experimentadas.

Para finalizar, se evaluado nuestro trabajo en una base de datos pública, diseñada específicamente para evaluar este tipo de sistemas, obteniendo resultados similares al estado del arte, y en algunos casos, ofreciendo mejores resultados, validando así la propuesta.

7.1. Trabajo futuro

Existen varias direcciones de trabajo futuro de las que nuestro trabajo se podría beneficiar. En primer lugar, se podría incluir un método para volver a reconocer un objeto después de que éste se haya dado como perdido. Con una adecuada colocación de los puntos de control esto permitiría interpolar la trayectoria que hasta ese momento había sido perdida. Para ello, podrían emplearse descriptores visuales de los objetos, por ejemplo a través de la creación de bolsas de palabras [Gálvez-López and Tardos, 2012], o a través de técnicas de aprendizaje profundo como [Wojke et al., 2017].

Otro paso interesante a realizar sería el de optimizar no solo la localización de los puntos de control y de los puntos del objeto, sino también las matrices de transformación de las cámaras y los puntos del mapa estático. Contando con un modelado adecuado del ruido en las observaciones, esto podría beneficiar al conjunto global de los parámetros al poder incrementar el número de términos relacionados en la matriz de información ($\mathbf{H}$). Otro detalle de cara a la optimización, sería el de llevar a cabo la marginalización de las variables de estado que dejan de optimizarse en la ventana temporal, según la propuesta de [Leutenegger et al., 2015].

Así mismo, para obtener mayores prestaciones de cara a la ejecución, podría plantearse cambiar el lenguaje de programación a C^{++}. Esto permitiría además emplear paquetes específicos de programación relacionados con la optimización de grafos de factores, como [Dellaert, 2012, Kümmerle et al., 2011].

Bibliografía

[Agarwal et al., 2011] Agarwal, S., Furukawa, Y., Snavely, N., Simon, I., Curless, B., Seitz, S. M., and Szeliski, R. (2011). Building rome in a day. *Communications of the ACM*, 54(10):105–112.

[Agarwal and Mierle, 2012] Agarwal, S. and Mierle, K. (2012). Ceres solver: Tutorial & reference. *Google Inc*, 2(72):8.

[Amayo et al., 2018] Amayo, P., Piniés, P., Paz, L. M., and Newman, P. (2018). Geometric multi-model fitting with a convex relaxation algorithm. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8138–8146.

[Arandjelović and Zisserman, 2012] Arandjelović, R. and Zisserman, A. (2012). Three things everyone should know to improve object retrieval. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2911–2918. IEEE.

[Avidan and Shashua, 2000] Avidan, S. and Shashua, A. (2000). Trajectory triangulation: 3d reconstruction of moving points from a monocular image sequence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(4):348–357.

[Barfoot, 2017] Barfoot, T. D. (2017). *State estimation for robotics*. Cambridge University Press.

[Baydin et al., 2018] Baydin, A. G., Pearlmutter, B. A., Radul, A. A., and Siskind, J. M. (2018). Automatic differentiation in machine learning: a survey. *Journal of machine learning research*, 18.

[Bertinetto et al., 2016] Bertinetto, L., Valmadre, J., Henriques, J. F., Vedaldi, A., and Torr, P. H. (2016). Fully-convolutional siamese networks for object tracking. In *European conference on computer vision*, pages 850–865. Springer.

[Bescos et al., 2021] Bescos, B., Campos, C., Tardós, J. D., and Neira, J. (2021). Dynaslam ii: Tightly-coupled multi-object tracking and slam. *IEEE Robotics and Automation Letters*, 6(3):5191–5198.

[Bescos et al., 2018] Bescos, B., Fácil, J. M., Civera, J., and Neira, J. (2018). Dynaslam: Tracking, mapping, and inpainting in dynamic scenes. *IEEE Robotics and Automation Letters*, 3(4):4076–4083.

[Bibby and Reid, 2007] Bibby, C. and Reid, I. (2007). Simultaneous localisation and mapping in dynamic environments (slamide) with reversible data association. In *Proceedings of Robotics: Science and Systems*, volume 66, page 81.

[Bibby and Reid, 2010] Bibby, C. and Reid, I. (2010). A hybrid slam representation for dynamic marine environments. In *2010 IEEE International Conference on Robotics and Automation*, pages 257–264. IEEE.

[Blanco, 2010] Blanco, J.-L. (2010). A tutorial on se (3) transformation parameterizations and on-manifold optimization. *University of Malaga, Tech. Rep*, 3:6.

[Bolme et al., 2010] Bolme, D. S., Beveridge, J. R., Draper, B. A., and Lui, Y. M. (2010). Visual object tracking using adaptive correlation filters. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 2544–2550. IEEE.

[Bouguet et al., 2001] Bouguet, J.-Y. et al. (2001). Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm. *Intel corporation*, 5(1-10):4.

[Cadena et al., 2016] Cadena, C., Carlone, L., Carrillo, H., Latif, Y., Scaramuzza, D., Neira, J., Reid, I., and Leonard, J. J. (2016). Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on robotics*, 32(6):1309–1332.

[Campos et al., 2021] Campos, C., Elvira, R., Rodríguez, J. J. G., Montiel, J. M., and Tardós, J. D. (2021). Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam. *IEEE Transactions on Robotics*.

[Concha and Civera, 2015] Concha, A. and Civera, J. (2015). An evaluation of robust cost functions for rgb direct mapping. In *2015 European Conference on Mobile Robots (ECMR)*, pages 1–8. IEEE.

[Concha et al., 2016] Concha, A., Loianno, G., Kumar, V., and Civera, J. (2016). Visual-inertial direct slam. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 1331–1338. IEEE.

[Corke, 2011] Corke, P. I. (2011). *Robotics, vision and control: fundamental algorithms in MATLAB*, volume 73. Springer.

[Cox, 1972] Cox, M. G. (1972). The numerical evaluation of b-splines. *IMA Journal of Applied mathematics*, 10(2):134–149.

[De Boor, 1972] De Boor, C. (1972). On calculating with b-splines. *Journal of Approximation theory*, 6(1):50–62.

[Dellaert, 2012] Dellaert, F. (2012). Factor graphs and gtsam: A hands-on introduction. Technical report, Georgia Institute of Technology.

[Dellaert et al., 2017] Dellaert, F., Kaess, M., et al. (2017). Factor graphs for robot perception. *Foundations and Trends® in Robotics*, 6(1-2):1–139.

[Droeschel and Behnke, 2018] Droeschel, D. and Behnke, S. (2018). Efficient continuous-time slam for 3d lidar-based online mapping. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5000–5007. IEEE.

[Eade, 2013] Eade, E. (2013). Gauss-newton/levenberg-marquardt optimization. *Tech. Rep.*

[Engel et al., 2017] Engel, J., Koltun, V., and Cremers, D. (2017). Direct sparse odometry. *IEEE transactions on pattern analysis and machine intelligence*, 40(3):611–625.

[Fischler and Bolles, 1981] Fischler, M. A. and Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395.

[Forster et al., 2016] Forster, C., Carlone, L., Dellaert, F., and Scaramuzza, D. (2016). On-manifold preintegration for real-time visual–inertial odometry. *IEEE Transactions on Robotics*, 33(1):1–21.

[Gálvez-López and Tardos, 2012] Gálvez-López, D. and Tardos, J. D. (2012). Bags of binary words for fast place recognition in image sequences. *IEEE Transactions on Robotics*, 28(5):1188–1197.

[Gao et al., 2003] Gao, X.-S., Hou, X.-R., Tang, J., and Cheng, H.-F. (2003). Complete solution classification for the perspective-three-point problem. *IEEE transactions on pattern analysis and machine intelligence*, 25(8):930–943.

[Geiger et al., 2012] Geiger, A., Lenz, P., and Urtasun, R. (2012). Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3354–3361. IEEE.

[Géron, 2019] Géron, A. (2019). *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems.* O'Reilly Media.

[Golub and Van Loan,] Golub, G. H. and Van Loan, C. F. Matrix computations. johns hopkins studies in the mathematical sciences.

[Goodfellow et al., 2016] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning.* MIT press.

[Haarbach et al., 2018] Haarbach, A., Birdal, T., and Ilic, S. (2018). Survey of higher order rigid body motion interpolation methods for keyframe animation and continuous-time trajectory estimation. In *2018 International Conference on 3D Vision (3DV)*, pages 381–389. IEEE.

[Han and Kanade, 2004] Han, M. and Kanade, T. (2004). Reconstruction of a scene with multiple linearly moving objects. *International Journal of Computer Vision*, 59(3):285–300.

[Hartley and Zisserman, 2004] Hartley, R. I. and Zisserman, A. (2004). *Multiple View Geometry in Computer Vision.* Cambridge University Press, ISBN: 0521540518, second edition.

[He et al., 2017] He, K., Gkioxari, G., Dollár, P., and Girshick, R. (2017). Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969.

[Huang et al., 2020] Huang, J., Yang, S., Mu, T.-J., and Hu, S.-M. (2020). Clustervo: Clustering moving instances and estimating visual odometry for self and surroundings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2168–2177.

[Huber, 2004] Huber, P. J. (2004). *Robust statistics*, volume 523. John Wiley & Sons.

[Judd and Gammell, 2019] Judd, K. M. and Gammell, J. D. (2019). The oxford multimotion dataset: Multiple se (3) motions with ground truth. *IEEE Robotics and Automation Letters*, 4(2):800–807.

[Judd and Gammell, 2020] Judd, K. M. and Gammell, J. D. (2020). Occlusion-robust mvo: Multimotion estimation through occlusion via motion closure. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5855–5862. IEEE.

[Judd et al., 2018] Judd, K. M., Gammell, J. D., and Newman, P. (2018). Multimotion visual odometry (mvo): Simultaneous estimation of camera and third-party motions. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3949–3956. IEEE.

[Kaminski and Teicher, 2004] Kaminski, J. Y. and Teicher, M. (2004). A general framework for trajectory triangulation. *Journal of Mathematical Imaging and Vision*, 21(1):27–41.

[Kerl et al., 2015] Kerl, C., Stuckler, J., and Cremers, D. (2015). Dense continuous-time tracking and mapping with rolling shutter rgb-d cameras. In *Proceedings of the IEEE international conference on computer vision*, pages 2264–2272.

[Kerl et al., 2013] Kerl, C., Sturm, J., and Cremers, D. (2013). Robust odometry estimation for rgb-d cameras. In *2013 IEEE international conference on robotics and automation*, pages 3748–3754. IEEE.

[Kim et al., 2016a] Kim, H., Leutenegger, S., and Davison, A. J. (2016a). Real-time 3d reconstruction and 6-dof tracking with an event camera. In *European Conference on Computer Vision*, pages 349–364. Springer.

[Kim et al., 2016b] Kim, J.-H., Cadena, C., and Reid, I. (2016b). Direct semi-dense slam for rolling shutter cameras. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1308–1315. IEEE.

[Kim et al., 1995] Kim, M.-J., Kim, M.-S., and Shin, S. Y. (1995). A general construction scheme for unit quaternion curves with simple high order derivatives. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 369–376.

[Kümmerle et al., 2011] Kümmerle, R., Grisetti, G., Strasdat, H., Konolige, K., and Burgard, W. (2011). g 2 o: A general framework for graph optimization. In *2011 IEEE International Conference on Robotics and Automation*, pages 3607–3613. IEEE.

[Kundu et al., 2011] Kundu, A., Krishna, K. M., and Jawahar, C. (2011). Realtime multibody visual slam with a smoothly moving monocular camera. In *2011 International Conference on Computer Vision*, pages 2080–2087. IEEE.

[Leutenegger et al., 2015] Leutenegger, S., Lynen, S., Bosse, M., Siegwart, R., and Furgale, P. (2015). Keyframe-based visual–inertial odometry using nonlinear optimization. *The International Journal of Robotics Research*, 34(3):314–334.

[Leys et al., 2013] Leys, C., Ley, C., Klein, O., Bernard, P., and Licata, L. (2013). Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median. *Journal of experimental social psychology*, 49(4):764–766.

[Li et al., 2018] Li, P., Qin, T., et al. (2018). Stereo vision-based semantic 3d object and ego-motion tracking for autonomous driving. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 646–661.

[Lovegrove et al., 2013] Lovegrove, S., Patron-Perez, A., and Sibley, G. (2013). Spline fusion: A continuous-time representation for visual-inertial fusion with application to rolling shutter cameras. In *BMVC*, volume 2, page 8.

[Lowe, 2004a] Lowe, D. (2004a). Distinctive image features from scale-invariant keypoints/ijcv.-vol. 60 (2).

[Lowe, 2004b] Lowe, D. G. (2004b). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110.

[Lucas et al., 1981] Lucas, B. D., Kanade, T., et al. (1981). An iterative image registration technique with an application to stereo vision.

[Lynch and Park, 2017] Lynch, K. M. and Park, F. C. (2017). *Modern robotics*. Cambridge University Press.

[Maclaurin et al., 2015] Maclaurin, D., Duvenaud, D., and Adams, R. P. (2015). Autograd: Effortless gradients in numpy. In *ICML 2015 AutoML workshop*, volume 238, page 5.

[Mueggler et al., 2018] Mueggler, E., Gallego, G., Rebecq, H., and Scaramuzza, D. (2018). Continuous-time visual-inertial odometry for event cameras. *IEEE Transactions on Robotics*, 34(6):1425–1440.

[Mueggler et al., 2015] Mueggler, E., Gallego, G., and Scaramuzza, D. (2015). Continuous-time trajectory estimation for event-based vision sensors. Technical report.

[Mur-Artal et al., 2015] Mur-Artal, R., Montiel, J. M. M., and Tardos, J. D. (2015). Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5):1147–1163.

[Mur-Artal and Tardós, 2017] Mur-Artal, R. and Tardós, J. D. (2017). Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE transactions on robotics*, 33(5):1255–1262.

[Newcombe et al., 2011] Newcombe, R. A., Lovegrove, S. J., and Davison, A. J. (2011). Dtam: Dense tracking and mapping in real-time. In *2011 international conference on computer vision*, pages 2320–2327. IEEE.

[Nistér, 2004] Nistér, D. (2004). An efficient solution to the five-point relative pose problem. *IEEE transactions on pattern analysis and machine intelligence*, 26(6):756–770.

[Nister and Stewenius, 2006] Nister, D. and Stewenius, H. (2006). Scalable recognition with a vocabulary tree. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 2161–2168. Ieee.

[Park et al., 2010] Park, H. S., Shiratori, T., Matthews, I., and Sheikh, Y. (2010). 3d reconstruction of a moving point from a series of 2d projections. In *European conference on computer vision*, pages 158–171. Springer.

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

[Pire et al., 2017] Pire, T., Fischer, T., Castro, G., De Cristóforis, P., Civera, J., and Berlles, J. J. (2017). S-ptam: Stereo parallel tracking and mapping. *Robotics and Autonomous Systems*, 93:27–42.

[Qin, 2000] Qin, K. (2000). General matrix representations for b-splines. *The Visual Computer*, 16(3-4):177–186.

[Sabzevari and Scaramuzza, 2014] Sabzevari, R. and Scaramuzza, D. (2014). Monocular simultaneous multi-body motion segmentation and reconstruction from perspective views. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 23–30. IEEE.

[Saputra et al., 2018] Saputra, M. R. U., Markham, A., and Trigoni, N. (2018). Visual slam and structure from motion in dynamic environments: A survey. *ACM Computing Surveys (CSUR)*, 51(2):1–36.

[Schonberger and Frahm, 2016] Schonberger, J. L. and Frahm, J.-M. (2016). Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113.

[Schops et al., 2019] Schops, T., Sattler, T., and Pollefeys, M. (2019). Bad slam: Bundle adjusted direct rgb-d slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 134–144.

[Shi et al., 1994] Shi, J. et al. (1994). Good features to track. In *1994 Proceedings of IEEE conference on computer vision and pattern recognition*, pages 593–600. IEEE.

[Sola, 2017a] Sola, J. (2017a). Course on slam. Technical report, Technical Report IRI-TR-16-04, Institut de Robòtica i.

[Sola, 2017b] Sola, J. (2017b). Quaternion kinematics for the error-state kalman filter. *arXiv preprint arXiv:1711.02508*.

[Sola et al., 2018] Sola, J., Deray, J., and Atchuthan, D. (2018). A micro lie theory for state estimation in robotics. *arXiv preprint arXiv:1812.01537*.

[Sommer et al., 2020] Sommer, C., Usenko, V., Schubert, D., Demmel, N., and Cremers, D. (2020). Efficient derivative computation for cumulative b-splines on lie groups. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11148–11156.

[Strasdat, 2012] Strasdat, H. (2012). *Local accuracy and global consistency for efficient visual SLAM*. PhD thesis, Department of Computing, Imperial College London.

[Sturm et al., 2012a] Sturm, J., Engelhard, N., Endres, F., Burgard, W., and Cremers, D. (2012a). A benchmark for the evaluation of rgb-d slam systems. In *Proc. of the International Conference on Intelligent Robot Systems (IROS)*.

[Sturm et al., 2012b] Sturm, J., Engelhard, N., Endres, F., Burgard, W., and Cremers, D. (2012b). A benchmark for the evaluation of rgb-d slam systems. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 573–580. IEEE.

[Sturm and Triggs, 1996] Sturm, P. and Triggs, B. (1996). A factorization based algorithm for multi-image projective structure and motion. In *European conference on computer vision*, pages 709–720. Springer.

[Tomasi and Kanade, 1992] Tomasi, C. and Kanade, T. (1992). Shape and motion from image streams under orthography: a factorization method. *International journal of computer vision*, 9(2):137–154.

[Torr and Zisserman, 2000] Torr, P. H. and Zisserman, A. (2000). Mlesac: A new robust estimator with application to estimating image geometry. *Computer vision and image understanding*, 78(1):138–156.

[Triggs et al., 1999] Triggs, B., McLauchlan, P. F., Hartley, R. I., and Fitzgibbon, A. W. (1999). Bundle adjustment—a modern synthesis. In *International workshop on vision algorithms*, pages 298–372. Springer.

[Umeyama, 1991] Umeyama, S. (1991). Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 13(04):376–380.

[Van Loan, 2000] Van Loan, C. F. (2000). The ubiquitous kronecker product. *Journal of computational and applied mathematics*, 123(1-2):85–100.

[Wang et al., 2003] Wang, C.-C., Thorpe, C., and Thrun, S. (2003). Online simultaneous localization and mapping with detection and tracking of moving objects: Theory and results from a ground vehicle in crowded urban areas. In *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)*, volume 1, pages 842–849. IEEE.

[Wang et al., 2007] Wang, C.-C., Thorpe, C., Thrun, S., Hebert, M., and Durrant-Whyte, H. (2007). Simultaneous localization, mapping and moving object tracking. *The International Journal of Robotics Research*, 26(9):889–916.

[Wang et al., 2021] Wang, H., Wang, C., Chen, C.-L., and Xie, L. (2021). F-loam: Fast lidar odometry and mapping. *arXiv preprint arXiv:2107.00822*.

[Wang et al., 2019] Wang, Q., Zhang, L., Bertinetto, L., Hu, W., and Torr, P. H. (2019). Fast online object tracking and segmentation: A unifying approach. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1328–1338.

[Wojke et al., 2017] Wojke, N., Bewley, A., and Paulus, D. (2017). Simple online and realtime tracking with a deep association metric. In *2017 IEEE international conference on image processing (ICIP)*, pages 3645–3649. IEEE.

[Wu, 2013] Wu, C. (2013). Towards linear-time incremental structure from motion. In *2013 International Conference on 3D Vision-3DV 2013*, pages 127–134. IEEE.

[Wu et al., 2011] Wu, C. et al. (2011). Visualsfm: A visual structure from motion system.

[Yang et al., 2021] Yang, A. J., Cui, C., Bârsan, I. A., Urtasun, R., and Wang, S. (2021). Asynchronous multi-view slam. *arXiv preprint arXiv:2101.06562*.

[Yang and Scherer, 2019] Yang, S. and Scherer, S. (2019). Cubeslam: Monocular 3-d object slam. *IEEE Transactions on Robotics*, 35(4):925–938.

[Zach, 2014] Zach, C. (2014). Robust bundle adjustment revisited. In *European Conference on Computer Vision*, pages 772–787. Springer.

[Zappella et al., 2013] Zappella, L., Del Bue, A., Lladó, X., and Salvi, J. (2013). Joint estimation of segmentation and structure from motion. *Computer Vision and Image Understanding*, 117(2):113–129.

[Zhang et al., 2020] Zhang, J., Henein, M., Mahony, R., and Ila, V. (2020). Vdo-slam: a visual dynamic object-aware slam system. *arXiv preprint arXiv:2005.11052*.

[Zhang and Scaramuzza, 2018] Zhang, Z. and Scaramuzza, D. (2018). A tutorial on quantitative trajectory evaluation for visual (-inertial) odometry. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7244–7251. IEEE.

Lista de Figuras

Lista de Tablas

Anexos

Apéndice A

Anexo matrices Jacobianas

En este Anexo se presentan algunos desarrollos utilizados en la derivación de las matrices Jacobianas de §4.3. En primer lugar, un resultado que se empleó en la Ec. 4.44, fue la **diferenciación de un vector con respecto a una matriz de transformación vectorizada** tal que:

$$¿\frac{\partial \mathbf{p}_a}{\partial \mathbf{T}}?, \quad \text{sabiendo que:} \quad \tilde{\mathbf{p}}_a = \mathbf{T}\tilde{\mathbf{p}}_b \quad \Rightarrow \quad \mathbf{p}_a = \mathbf{R}\mathbf{p}_b + \mathbf{t} = \begin{bmatrix} \mathbf{R}^{r1}\mathbf{p}_b + \mathbf{t}^1 \\ \mathbf{R}^{r2}\mathbf{p}_b + \mathbf{t}^2 \\ \mathbf{R}^{r3}\mathbf{p}_b + \mathbf{t}^3 \end{bmatrix} \tag{A.1}$$

Donde recordemos que la notación $\mathbf{R}^{ci}, \mathbf{R}^{fi}, \mathbf{t}^j$ hacían referencia a la columna y fila i de $\mathbf{R}$ y al $j-$ésimo elemento de $\mathbf{t}$, respectivamente. Ahora, recordando que la versión vectorizada de una matriz de transformación viene dada por:

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}^{c1} & \mathbf{R}^{c2} & \mathbf{R}^{c3} & \mathbf{t} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \Rightarrow \quad \text{vec}(\mathbf{T}_{wo}) = \begin{bmatrix} \mathbf{R}^{c1} \\ \mathbf{R}^{c2} \\ \mathbf{R}^{c3} \\ \mathbf{t} \end{bmatrix} \in \mathbb{R}^{12} \tag{A.2}$$

Significa que diferenciando cada elemento del vector $\mathbf{p}_a$ con respecto a cada elemento de la versión vectorizada de $\mathbf{T}$, se obtiene:

$$\frac{\partial \mathbf{p}_1}{\partial \mathbf{T}} = \begin{bmatrix} \mathbf{p}_b^1 & 0 & 0 & \mathbf{p}_b^2 & 0 & 0 & \mathbf{p}_b^3 & 0 & 0 & 1 & 0 & 0 \\ 0 & \mathbf{p}_b^1 & 0 & 0 & \mathbf{p}_b^2 & 0 & 0 & \mathbf{p}_b^3 & 0 & 0 & 1 & 0 \\ 0 & 0 & \mathbf{p}_b^1 & 0 & 0 & \mathbf{p}_b^2 & 0 & 0 & \mathbf{p}_b^3 & 0 & 0 & 1 \end{bmatrix} \tag{A.3}$$

Que, afortunadamente se puede expresar de manera compacta a través del producto de Kronecker $\otimes$ [Van Loan, 2000]:

$$\frac{\partial \mathbf{p}_a}{\partial \mathbf{T}} = \begin{bmatrix} \mathbf{p}_b^T & 1 \end{bmatrix} \otimes \mathbf{I}_{3\times 3} \tag{A.4}$$

Otra de las herramientas usadas en Ecs. como 4.54 y 4.55, es la **diferenciación entre matrices de transformación vectorizadas** tales que:

$$¿\frac{\partial \mathbf{T}_a \mathbf{T}_b}{\partial \mathbf{T}_b}?, \qquad ¿\frac{\partial \mathbf{T}_a \mathbf{T}_b}{\partial \mathbf{T}_a}? \tag{A.5}$$

El primer término de la Ec. A.5, lo podemos derivar desplegando $\mathbf{T}_a\mathbf{T}_b$ en su versión vectorizada:

$$\frac{\partial \mathbf{T}_a\mathbf{T}_b}{\partial \mathbf{T}_b} = \frac{\partial}{\partial \mathbf{T}_b}\left(\begin{bmatrix}\mathbf{R}_a\mathbf{R}_b & \mathbf{R}_a\mathbf{t}_b+\mathbf{t}_a\\ \mathbf{0} & 1\end{bmatrix}\right) = \frac{\partial}{\partial \mathbf{T}_b}\begin{pmatrix}\begin{bmatrix}\mathbf{R}_a^{r1}\mathbf{R}_b^{c1}\\ \mathbf{R}_a^{r2}\mathbf{R}_b^{c1}\\ \mathbf{R}_a^{r3}\mathbf{R}_b^{c1}\\ \mathbf{R}_a^{r1}\mathbf{R}_b^{c2}\\ \vdots \\ \mathbf{R}_a^{r1}\mathbf{t}_b+\mathbf{t}_a^0\\ \mathbf{R}_a^{r2}\mathbf{t}_b+\mathbf{t}_a^1\\ \mathbf{R}_a^{r3}\mathbf{t}_b+\mathbf{t}_a^2\end{bmatrix}\end{pmatrix} \tag{A.6}$$

Centrándonos en un sub-bloque:

$$\frac{\partial}{\partial \mathbf{T}_b^{ci}}\left(\begin{bmatrix}\mathbf{R}_a^{r1}\mathbf{T}_b^{ci}\\ \mathbf{R}_a^{r2}\mathbf{T}_b^{ci}\\ \mathbf{R}_a^{r3}\mathbf{T}_b^{ci}\end{bmatrix}\right) = \mathbf{R}_a \tag{A.7}$$

Implica que organizando los elementos según la columna i correspondiente, obtenemos:

$$\frac{\partial \mathbf{T}_a\mathbf{T}_b}{\partial \mathbf{T}_b} = \begin{bmatrix}\mathbf{R}_a & \mathbf{0} & \mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{R}_a & \mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{0} & \mathbf{R}_a & \mathbf{0}\\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{R}_a\end{bmatrix} = \mathbf{I}_{4\times 4}\otimes\mathbf{R}_a \tag{A.8}$$

El segundo término de la Ec. A.5, lo podemos obtener basándonos igualmente en el desarrollo de la Ec. A.6. Por ejemplo, si derivamos manualmente las 3 primeras filas:

$$\frac{\partial}{\partial \mathbf{T}_a}\left(\begin{bmatrix}\mathbf{R}_a^{r1}\mathbf{R}_b^{c1}\\ \mathbf{R}_a^{r2}\mathbf{R}_b^{c1}\\ \mathbf{R}_a^{r3}\mathbf{R}_b^{c1}\end{bmatrix}\right) = \begin{bmatrix}\mathbf{T}_b^{11} & 0 & 0 & \mathbf{T}_b^{21} & 0 & 0 & \mathbf{T}_b^{31} & 0 & 0 & \mathbf{T}_b^{41} & 0 & 0\\ 0 & \mathbf{T}_b^{11} & 0 & 0 & \mathbf{T}_b^{21} & 0 & 0 & \mathbf{T}_b^{31} & 0 & 0 & \mathbf{T}_b^{41} & 0\\ 0 & 0 & \mathbf{T}_b^{11} & 0 & 0 & \mathbf{T}_b^{21} & 0 & 0 & \mathbf{T}_b^{31} & 0 & 0 & \mathbf{T}_b^{41}\end{bmatrix} \tag{A.9}$$

Siguiendo esta derivación manual, se obtiene finalmente:

$$\frac{\partial \mathbf{T}_a\mathbf{T}_b}{\partial \mathbf{T}_a} = \mathbf{T}_b^T\otimes\mathbf{I}_{3\times 3} \tag{A.10}$$

Finalmente, el resultado empleado en la Ec. 4.57 se justifica a continuación. Éste consiste en la siguiente **diferenciación del mapeo exponencial evaluado en la identidad**:

$$\left.\frac{\partial \mathrm{Exp}(\boldsymbol{\tau})}{\partial \boldsymbol{\tau}}\right|_{\boldsymbol{\tau}=\mathbf{0}} \tag{A.11}$$

Tal y como se vio en la Ec. 3.46, el mapeo exponencial viene dado por:

$$\exp(\boldsymbol{\tau}^\wedge) = \exp\left(\begin{bmatrix}\mathbf{v}\\ \boldsymbol{\omega}\end{bmatrix}^\wedge\right) = \begin{bmatrix}\exp(\boldsymbol{\omega}^\wedge) & \mathbf{V}\mathbf{v}\\ \mathbf{0} & 1\end{bmatrix}, \tag{A.12}$$

$$\text{donde } \begin{cases}\exp(\boldsymbol{\omega}^\wedge) & = \mathbf{I}_3 + \frac{\sin\theta}{\theta}\boldsymbol{\omega}^\wedge + \frac{1-\cos\theta}{\theta^2}(\boldsymbol{\omega}^\wedge)^2\\ \mathbf{V} & = \mathbf{I}_3 + \frac{1-\cos\theta}{\theta^2}\boldsymbol{\omega}^\wedge + \frac{\theta-\sin\theta}{\theta^3}(\boldsymbol{\omega}^\wedge)^2\end{cases}, \text{ con } \theta = \|\boldsymbol{\omega}\| \tag{A.13}$$

Las derivaciones correspondientes a $\mathbf{v} \in \mathbb{R}^3$ son directas:

$$\frac{\partial \exp(\boldsymbol{\omega}^\wedge)}{\partial \mathbf{v}} = \mathbf{0}_{12\times 3},\ \forall \mathbf{v} \in \mathbb{R}^3 \tag{A.14}$$

$$\frac{\partial \mathbf{V}\mathbf{v}}{\partial \mathbf{v}} = \mathbf{V},\ \forall \mathbf{v} \in \mathbb{R}^3 \Rightarrow \left.\frac{\partial \mathbf{V}\mathbf{v}}{\partial \mathbf{v}}\right|_{\boldsymbol{\tau}=\mathbf{0}} = \mathbf{I}_{3\times 3} \tag{A.15}$$

Por otro lado, en cuanto a $\boldsymbol{\omega} \in \mathbb{R}^3$, se observa lo siguiente:

$$\left.\frac{\partial \mathbf{V}\mathbf{v}}{\partial \boldsymbol{\omega}}\right|_{\boldsymbol{\tau}=\mathbf{0}} = \cancelto{\mathbf{0}}{\left.\frac{\partial \mathbf{V}\mathbf{v}}{\partial \mathbf{V}}\right|_{\boldsymbol{\tau}=\mathbf{0}}} \left.\frac{\partial \mathbf{V}}{\partial \boldsymbol{\omega}}\right|_{\boldsymbol{\tau}=\mathbf{0}} + \left.\frac{\partial \mathbf{V}\mathbf{v}}{\partial \mathbf{v}}\right|_{\boldsymbol{\tau}=\mathbf{0}} \cancelto{\mathbf{0}}{\left.\frac{\partial \mathbf{v}}{\partial \boldsymbol{\omega}}\right|_{\boldsymbol{\tau}=\mathbf{0}}} = \mathbf{0}_{3\times 3} \tag{A.16}$$

De manera directa $\partial \mathbf{v}/\partial \boldsymbol{\omega} = \mathbf{0}$. Para justificar por qué $\partial \mathbf{V}\mathbf{v}/\partial \mathbf{V} = \mathbf{0}$, nos podemos fijar en su resultado general:

$$\frac{\partial \mathbf{V}\mathbf{v}}{\partial \mathbf{V}} = \mathbf{v}^T \otimes \mathbf{I}_{3\times 3} \tag{A.17}$$

Por lo que si $\mathbf{v} = \mathbf{0} \Rightarrow \partial \mathbf{V}\mathbf{v}/\partial \mathbf{V} = \mathbf{0}$

Por último, en cuanto a $\partial \exp(\boldsymbol{\omega}^\wedge)/\partial \boldsymbol{\omega}$ evaluada en $\boldsymbol{\omega} = \mathbf{0}$, algunos resultados que permiten simplificar su derivación son:

$$\frac{\partial}{\partial \boldsymbol{\omega}} \mathrm{vec}(\mathbf{I}_3) = \mathbf{0}_{9\times 3} \tag{A.18}$$

$$\left.\frac{\partial}{\partial \omega_k} \frac{\sin\theta}{\theta}\right|_{\boldsymbol{\omega}=\mathbf{0}} = 0 \tag{A.19}$$

$$\left.\frac{\partial}{\partial \omega_k} \frac{1-\cos\theta}{\theta^2}\right|_{\boldsymbol{\omega}=\mathbf{0}} = 0 \tag{A.20}$$

$$\left.\frac{\partial}{\partial \omega_k} \boldsymbol{\omega}^\wedge\right|_{\boldsymbol{\omega}=\mathbf{0}} = \mathbf{G}_k \tag{A.21}$$

$$\left.\frac{\partial}{\partial \omega_k} (\boldsymbol{\omega}^\wedge)^2\right|_{\boldsymbol{\omega}=\mathbf{0}} = \boldsymbol{\omega}^\wedge \mathbf{G}_k + \mathbf{G}_k \boldsymbol{\omega}^\wedge\Big|_{\boldsymbol{\omega}=\mathbf{0}} = \mathbf{0}_{3\times 3} \tag{A.22}$$

$$\lim_{\theta\to 0} \frac{\sin\theta}{\theta} = 1 \tag{A.23}$$

Donde ω_k es el elemento k-ésimo de $\boldsymbol{\omega} = [\omega_1, \omega_2, \omega_3]^T$. Por tanto:

$$\left.\frac{\partial}{\partial \omega_1} \exp(\boldsymbol{\omega}^\wedge)\right|_{\boldsymbol{\omega}=\mathbf{0}} = \mathbf{G}_1, \quad \left.\frac{\partial}{\partial \omega_2} \exp(\boldsymbol{\omega}^\wedge)\right|_{\boldsymbol{\omega}=\mathbf{0}} = \mathbf{G}_2, \quad \left.\frac{\partial}{\partial \omega_3} \exp(\boldsymbol{\omega}^\wedge)\right|_{\boldsymbol{\omega}=\mathbf{0}} = \mathbf{G}_3, \tag{A.24}$$

es decir, en su versión vectorizada pasa a ser:

$$\left.\frac{\partial}{\partial \boldsymbol{\omega}} \exp(\boldsymbol{\omega}^\wedge)\right|_{\boldsymbol{\omega}=\mathbf{0}} = \begin{bmatrix} -\mathbf{G}_1 \\ -\mathbf{G}_2 \\ -\mathbf{G}_3 \end{bmatrix} \tag{A.25}$$

Por lo que finalmente, juntando todo lo anterior:

$$\left.\frac{\partial \mathrm{Exp}(\boldsymbol{\tau})}{\partial \boldsymbol{\tau}}\right|_{\boldsymbol{\tau}=\mathbf{0}} = \begin{bmatrix} \mathbf{0}_{3\times 3} & -\mathbf{G}_1 \\ \mathbf{0}_{3\times 3} & -\mathbf{G}_2 \\ \mathbf{0}_{3\times 3} & -\mathbf{G}_3 \\ \mathbf{I}_{3\times 3} & -\mathbf{0}_{3\times 3} \end{bmatrix} \tag{A.26}$$