.

Universidad Zaragoza
1542

TFG en Ingeniería Electrónica y Automática

Sistema de adquisición en alta calidad y calibración para endoscopia

Autor

Enrique Bauzá Mingueza

Director

José María Martínez Montiel

Escuela de Ingeniería y Arquitectura
2020

Causas de la modificación a la propuesta inicial

Como se ha podido observar en la introducción, el trabajo realizado finalmente no se corresponde en su totalidad al propuesto en un inicio. El principal motivo de ellos es que el desarrollo del sistema de adquisición de endoscopias en alta calidad lo ha llevado a cabo el personal investigador del Departamento de Ingeniería de Sistemas ya que este software puede ser de gran utilidad para proyectos de gran relevancia y, por lo tanto, su magnitud era mayor que la de un TFG.

Además, también se da la situación de que, la calibración de una cámara y la utilización de esos parámetros para un procesado posterior de las imágenes, están estrechamente ligadas. En mi caso ese procesado se trata del SLAM y conforme avanzaba en el trabajo de calibración surgió la posibilidad de sustituir la parte del desarrollo del sistema de adquisición por la de la aplicación de SLAM a endoscopias, lo que también tiene un gran interés desde el punto de vista de la investigación, ya que no es un campo muy trabajado en relación a la aplicación del SLAM a otro tipo de secuencias.

Índice

Capítulo 1

Introducción

El SLAM es una técnica mediante la cual se construye un mapa de un entorno desconocido al mismo tiempo que se va calculando la localización geométrica del agente mapeador dentro del mismo. El SLAM es ampliamente utilizado en áreas como navegación, robótica, realidad aumentada o realidad virtual.

Con SLAM no nos referimos a un algoritmo en concreto sino a la técnica en general. De esta manera hay diferentes tipos de SLAM dependiendo, por ejemplo, del tipo de sensor que utilicemos, desde un sensor de ultrasonidos para medir distancias hasta una cámara, o del algoritmo que se implemente. El caso que nos ocupa en este trabajo es el del SLAM visual.

En el vSLAM (Visual Simultaneous Localization and Mapping) el sensor que se utiliza es una cámara. Por lo tanto la información para realizar la reconstrucción de la escena y el cálculo de la posición de la cámara se extrae de las imágenes tomadas por ésta. A grandes rasgos, los algoritmos de vSLAM funcionan de la siguiente manera: detectan puntos de interés, *KeyPoints*, en las imágenes, estos puntos son aquellos suficientemente diferentes (en color y forma) a los de su entorno, y asignan una descripción a esos puntos, de manera que un mismo punto de la escena tendrá una descripción

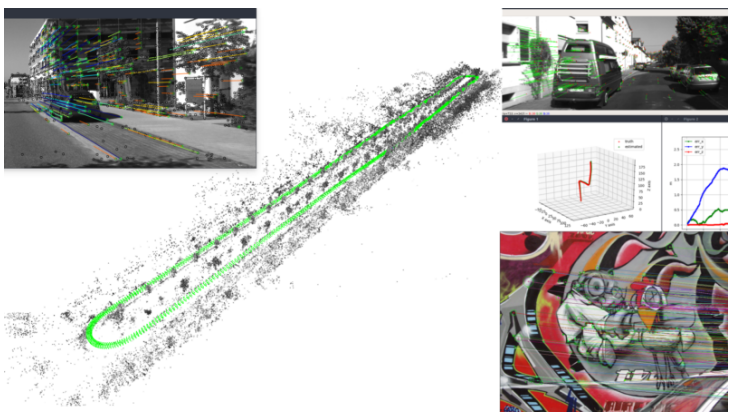

Figura 1.1: Ejemplo de funcionamiento de vSLAM.

prácticamente igual en todas las imágenes en las que aparezca. A continuación se hace un emparejamiento entre puntos de diferentes imágenes que tengan unos descriptores muy similares. De esta manera lo que se intenta es localizar un mismo punto de la escena en varias imágenes. A partir de estos emparejamientos se puede calcular la geometría de la escena y la posición de la cámara en ella. Esto se va realizando en tiempo real en la mayoría de aplicaciones, o de manera secuencial, siguiendo el orden de grabación de las imágenes, si se procesa una secuencia grabada anteriormente como es nuestro caso. De esta manera conforme se van procesando nuevas imágenes se va añadiendo más información al mapa y se va actualizando la trayectoria que ha seguido la cámara.

Aquí podemos ver un ejemplo de funcionamiento de vSLAM:

`https://youtu.be/sr9H3ZsZCzc?t=12`.

Sin embargo, a la hora de emplear el vSLAM en endoscopias, surgen varios problemas derivados de la naturaleza de la escena. Los mismos algoritmos de detección de puntos, que en otro tipo de escenas tienen un funcionamiento realmente bueno, no se comportan de igual manera en este tipo de imágenes médicas, lo que implica que haya menos información para el desarrollo del proceso. Esto es debido a la homogeneidad en cuanto a color y forma de las imágenes, que es mucho mayor en escenas médicas que la que puede tener una imagen del interior de un edificio o de un entorno urbano.

Además las superficies del interior del cuerpo humano no son rígidas, lo que hace que el cálculo geométrico basado en localizar el mismo punto en dos imágenes, asumiendo que la posición de ese punto es fija y solo se ha desplazado la cámara, se complique, ya que ese punto puede haber cambiado su posición 3D de una imagen a otra.

Otro problema que surge es que el tipo de óptica (*Fish-Eye*) de los endoscopios, diseñada para abarcar un gran ángulo de vista, hace que las imágenes tomadas por estos estén significativamente distorsionadas, por lo que es necesario una desdistorsión de las mismas antes de poder procesarlas.

Por último, debido a la naturaleza de las secuencias grabadas con un endoscopio, es muy probable que durante el procesado de la secuencia se pierda la localización de la cámara y por lo tanto se detenga la actualización del mapa. Esto se debe a que en algunas situaciones, la escena observada cambia mucho entre 2 imágenes relativamente próximas de la secuencia, lo que implica que el proceso de actualización secuencial del mapa y la trayectoria de la cámara se detenga, ya que en esas situaciones no será posible realizar emparejamientos de puntos entre esas 2 imágenes porque, básicamente, no estarán viendo los mismo puntos. Estas situaciones se pueden dar cuando el endoscopio realiza un giro, no necesariamente de muchos grados, dentro del intestino y, debido a que las distancias entre endoscopio y superficies son relativamente pequeñas, la escena

observada cambia significativamente. También puede ocurrir que, debido a la forma del intestino, el cual tiene varios giros pronunciados en su recorrido, haya un cambio radical de la escena observada entre imágenes próximas.

Para entender mejor esta problemática aquí se puede ver una colonoscopia:
`https://youtu.be/eCPySqbIk8U?t=97`.

En este trabajo se ha abordado la problemática de la pérdida de localización de la cámara y actualización del mapa utilizando ORBSLAM-Atlas [1]. La peculiaridad de este software de vSLAM es que es capaz de crear diversos mapas del entorno en caso de que se pierda la localización, lo que se ajusta perfectamente a nuestro problema. De esta manera se ha buscado mapear el intestino en partes, ya que no tiene sentido plantearlo de otra manera.

La solución del problema de la detección de puntos está más limitada ya que no se ha variado el algoritmo de detección utilizado, sino que se ha optado por un ajuste de los parámetros para adaptar su funcionamiento a nuestro caso. Este ajuste consiste básicamente en ampliar los rangos de detección de puntos ya que estos parámetros están pensados para escenas mucho más heterogéneas, en las que podemos ser mucho más restrictivos en cuanto a lo que consideramos como un punto diferente a su entorno.

El problema de la distorsión de las imágenes se ha resuelto mediante la calibración del endoscopio. Esta calibración nos permitirá conocer los parámetros de la cámara que describen la forma en que la escena se proyecta en la imagen y, partir de ellos, podremos desdistorsionar las imágenes para poder emplearlas en el vSLAM.

Capítulo 2

Bundle Adjustment (BA) en la reconstrucción 3D

Para entender el funcionamiento de los algoritmos de reconstrucción 3D a partir de imágenes es imprescindible introducir el concepto de correspondencia.

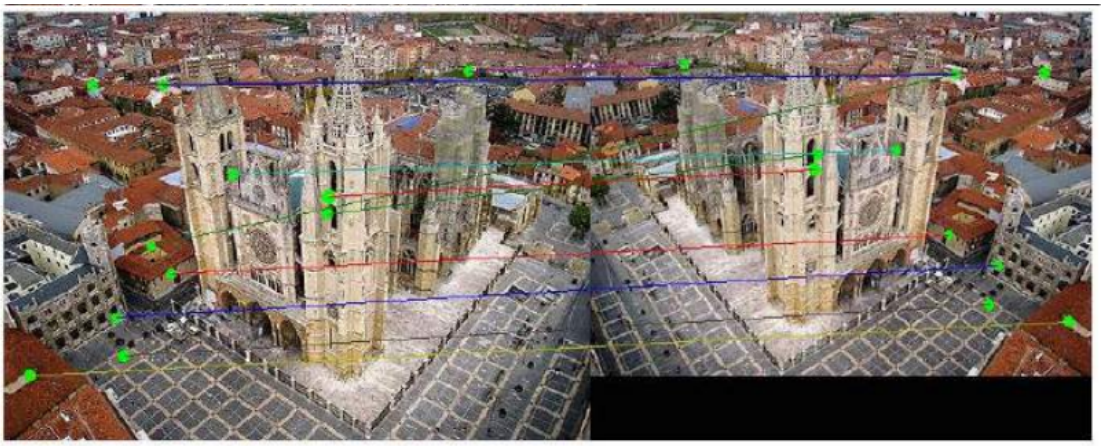

Figura 2.1: Correspondencias entre 2 imágenes representadas por las líneas de colores. [2]

Una ***correspondencia*** es un emparejamiento entre KeyPoints de 2 imágenes cuyos descriptores sean muy similares. El descriptor es la forma en la que algoritmo de detección de puntos de interés codifica las características del punto detectado. Con el cálculo de correspondencias lo que buscamos es conocer en qué imágenes aparece un mismo punto de la escena. Esto se puede ver en la figura 2.1.

Los métodos de reconstrucción 3D se basan en encontrar correspondencias entre imágenes, seleccionar grupos de imágenes con un gran número de correspondencias y, a partir de estas imágenes, triangular la posición de los puntos 3D y estimar la posición de la cámara para cada imagen. Sin embargo, para mejorar la precisión de esta reconstrucción 3D, posteriormente se realiza un ajuste no lineal. Para propiciar la convergencia de este ajuste no lineal utilizamos la solución obtenida de la triangulación

como solución inicial del ajuste.

Este ajuste es conocido como ***Bundle Adjustment*** o ajuste de haces. Este ajuste no lineal nos permite calcular al mismo tiempo:

- Los parámetros extrínsecos, que dan cuenta de la posición de la cámara.
- La posición 3D de los puntos de la escena.
- Y los parámetros del modelo que utilizamos para la cámara, en caso de que no sean conocidos.

Este ajuste no lineal consiste en minimizar el *error de reproyección*, que es la distancia entre la imagen de un punto de la escena y la proyección del mismo punto de la escena en el plano de imagen mediante el modelo:

$$\underset{\mathbf{p},\mathbf{X}_i^j,\theta_{ext}^j}{\arg\min} \sum_{i}^{N} \sum_{j}^{M} \left\| \mathbf{u}_i^j - \widehat{\mathbf{u}}_i^j(\mathbf{X}_i^j, \mathbf{p}, \theta_{ext}^j) \right\|^2 \tag{2.1}$$

Donde $\mathbf{p} = (k_1, k_2, k_3, k_4, f_u, f_v, c_u, c_v)$ es un vector que contiene los parámetros del modelo (presentación del modelo en sección 4) y θ_{ext} es el vector de parámetros extrínsecos, que contiene 3 componentes de rotación y 3 de traslación y nos aporta la información sobre la posición de la cámara. $\mathbf{N}$ es el número de puntos que reproyectamos y $\mathbf{M}$ es el número de imágenes que tomamos para el ajuste, $\mathbf{X}_i^j$ son las coordenadas 3D trianguladas del punto que reproyectamos, $\mathbf{u}_i^j$ son las coordenadas en la imagen de ese punto y $\widehat{\mathbf{u}}_i^j$ son las coordenadas en la imagen del punto $\mathbf{X}_i^j$ proyectado según nuestro modelo con los parámetros $\mathbf{p}$.

El error de reproyección se mide en píxeles ya que es una diferencia entre posiciones en el plano de la imagen.

En lo que respecta a nuestro trabajo el Bundle Adjustment es de gran importancia ya que se utiliza constantemente.

ORBSLAM-Atlas (descripción mas detallada en sección 3) utiliza el BA de dos formas diferentes:

- En el *Tracking*: En este caso tanto la calibración de la cámara (parámetros del modelo), como la posición de los puntos son conocidas y se fijan para el ajuste, solo se optimiza la posición de la cámara (parámetros extrínsecos) para cada frame. Los puntos utilizados son los puntos del mapa que identificamos en el frame que esté siendo procesado. Como solución inicial de la posición, se predice

la posición de la cámara a partir del frame anterior mediante un modelo de velocidad constante.

– En el *Local Mapping*: Cada vez que se inserta un nuevo KeyFrame (3.1.2) se realiza un BA con ese KeyFrame y todos los anteriores que observen los mismos puntos del mapa. Aquí la calibración es, de nuevo, conocida y el ajuste optimiza simultáneamente la posición de los puntos y las poses de las cámaras. En este caso, al ir añadiendo pequeñas partes al mapa con cada KeyFrame, la solución inicial para la convergencia la tenemos en los KeyFrames y posición de los puntos del mapa calculados previamente.

El BA también es de gran utilidad en la calibración de cámaras ya que, a partir de una solución inicial calculada mediante métodos lineales que propicie la convergencia del ajuste, podemos calcular los parámetros del modelo que se quiera calibrar (sección 4.4).

Capítulo 3

Aplicación de SLAM a endoscopias

En el presente capítulo, y en lo que resta de memoria, nos referiremos al vSLAM simplemente como SLAM porque en este trabajo no se ha utilizado ninguna otra modalidad de SLAM.

3.1. ORBSLAM-Atlas y ORBSLAM

3.1.1. Descripción del sistema

ORBSLAM-Atlas es un software diseñado a partir de ORBSLAM [3], cuyas principales contribuciones a éste último son la implementación de todo el sistema multimapa, tanto a nivel de representación gráfica para el usuario como a nivel algorítmico, y un cambio de criterio para considerar que la localización de la cámara se ha perdido.

Por lo tanto tiene sentido explicar algunas de sus características para poder entender el trabajo realizado. Para un conocimiento más detallado de su funcionamiento consultar su respectiva documentación.

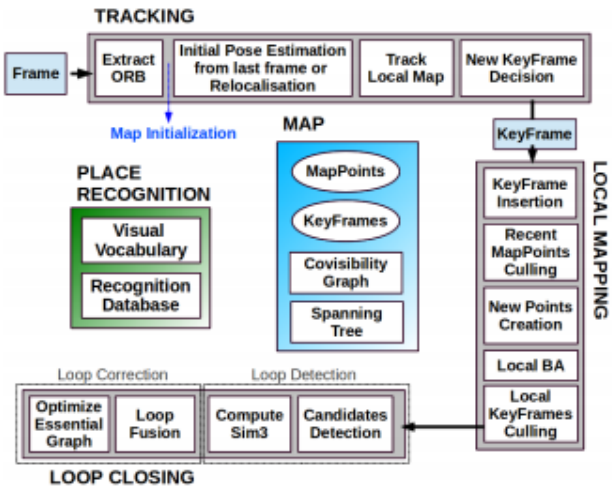

Figura 3.1: Sistema ORBSLAM

ORBSLAM es un sistema de SLAM monocular que realiza una reconstrucción en forma de mapa de puntos de la escena. Este mapa, a parte de los puntos 3D de la escena, también está formado por una serie de frames específicos llamados KeyFrames. Cada uno de estos KeyFrames contiene información de la posición de la cámara cuando grabó ese frame, la calibración de esa cámara y de todos los KeyPoints que se observan en ese frame.

Para calcular la posición de estos puntos del mapa y la posición de la cámara al mismo tiempo se utiliza Bundle Adjustment. Sin embargo debido al alto coste computacional del Bundle Adjustment no podemos aplicarlo a todos los frames de la secuencia, ya que haría imposible la operatividad del programa en aplicaciones en tiempo real. Por ello este BA se realiza solo con algunos frames que cumplen unas condiciones, estos son los denominados ***KeyFrames***.

Como podemos ver en la figura 3.1 el programa cuenta con 3 hilos de ejecución concurrentes. El primero de ellos, el *Tracking*, tiene 2 funciones, estimar la posición de la cámara según la información del frame actual, y evaluar si ese frame reúne las condiciones para ser un KeyFrame. El segundo, *Local Mapping*, tiene la función de actualizar el mapa en base a los KeyFrames mediante triangulación y BA, y de refinar tanto los puntos del mapa como los KeyFrames de la secuencia, eliminando aquellos que sean redundantes o poco precisos. Por último hay un tercer hilo encargado del *Loop Closing*, es decir, detectar si la escena que se está viendo en ese momento ya ha sido vista antes.

3.1.2. Inserción de KeyFrames

Los criterios de decisión para insertar un nuevo KeyFrame son los siguientes:

- Deben haber pasado un mínimo de frames igual al ratio de grabación de la cámara, FPS, desde la inserción del último KeyFrame.

- El frame actual se siguen menos del 90 % de los puntos que en el KeyFrame de referencia. Este KeyFrame de referencia es aquel, de todos lo KeyFrames, que tiene más puntos en común con el frame actual.

- En el frame actual se siguen, al menos, 50 puntos.

Aquí, con *seguir* un punto, nos referimos a que en el frame actual se ha detectado un KeyPoint cuyo descriptor es suficientemente similar al de alguno de los puntos que forman nuestro mapa 3D y, por lo tanto, se puede decir que se está viendo ese mismo punto.

Las 2 primeras condiciones buscan que no se inserten demasiados KeyFrames por dos razones, para evitar elevar innecesariamente el coste computacional y porque cuando se realiza el BA para calcular la geometría de la escena, ésta solo queda bien condicionada si el *paralaje* entre los KeyFrames es suficientemente grande. Este paralaje es el ángulo que forman los rayos que unen el mismo punto de la escena con 2 cámaras en las que se proyecta o, en nuestro caso, la misma cámara en 2 instantes diferentes. El paralaje es significativo cuando hay suficiente traslación entre las cámaras, por lo tanto, entre 2 KeyFrames próximos, el paralaje va a ser demasiado bajo.

En la segunda condición lo que se busca expresamente es que la escena que estemos viendo haya cambiado lo suficiente como para no estar acumulando una gran cantidad de KeyFrames de una misma parte del entorno y mapeándola en exceso.

La última condición va en el sentido de que el frame que insertemos tenga un número de KeyPoints significativo como para incluirlo como KeyFrame, es decir, que aporte un mínimo de información al mapa.

3.1.3. ORBSLAM-Atlas

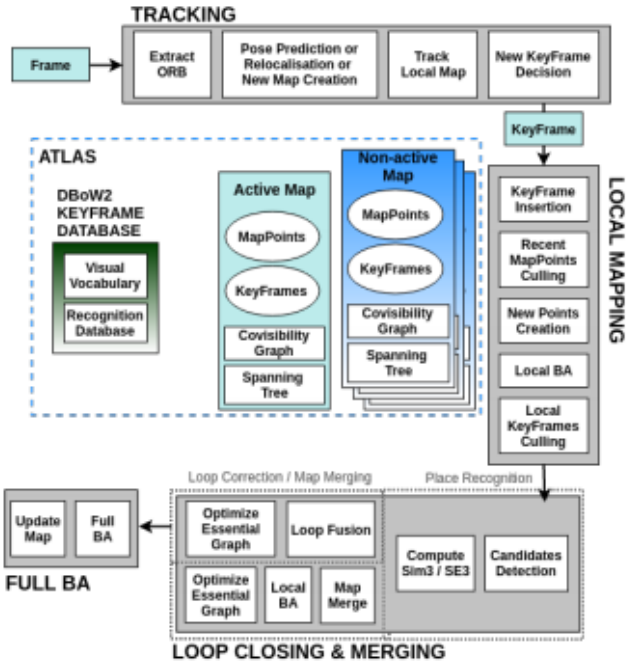

Figura 3.2: Sistema ORBSLAM-Atlas

ORBSLAM-Atlas sigue la misma lógica de creación de mapas que ORBSLAM pero implementa un sistema multimapa, de forma que, si en algún momento se pierde la localización de la cámara y no es posible relocalizarse en el mapa actual, ORBSLAM-Atlas inicializa un nuevo mapa. Además se añade una nueva funcionalidad en la parte de *Loop Closing*, se trata del *Merging*.

Todos los mapas se almacenan y se va comprobando durante la ejecución si el mapa que se está construyendo coincide con alguno de los ya creados hasta ese momento, en tal caso, el software fusiona ambos mapas recalculando la posición de la cámara en el mapa fusionado en base a los 2 originales. La comprobación de estar mapeando una zona ya mapeada se conoce como *Place Recognition*, y funciona de la siguiente manera:

- Candidates Detection: Primero se compara la bolsa de palabras (DBoW2) del KeyFrame en cuestión con las bolsas de palabras de otros KeyFrames. Si se encuentra suficiente similitud entre ellas los KeyFrames son propuestos como candidatos.

- Compute Sim3/SE3: Se intenta calcular una transformación rígida entre los puntos de los KeyFrames propuestos como candidatos.

Si ha sido posible calcular esa transformación rígida se procede a la fusión del mapa actual con el mapa al que pertenecen los KeyFrames candidatos (*Map Merging*). Si todos los KeyFrames candidatos corresponden al mismo mapa simplemente se hace una corrección de ese mapa en base a la nueva información (*Loop Correction*).

Al finalizar el procesado de la secuencia disponemos de todos los mapas que se han ido creando durante la ejecución.

En cuanto a la interfaz gráfica de ORBSLAM-Atlas, ésta está compuesta de 3 ventanas, una en la que se muestra el frame actual y los puntos del mapa que se ven en él, otra, el *MapViewer*, en la que se muestra el mapa 3D actual y que se va actualizando con los nuevos KeyFrames, y la tercera, el *AtlasViewer*, en la que aparecen las miniaturas de todos los mapas que se han generado hasta el momento.

3.1.4. ORB Extractor (Oriented FAST and Rotated BRIEF)

ORBSLAM utiliza el extractor de puntos ORB para extraer los puntos de interés de las imágenes. Este extractor tiene la ventaja de que su coste computacional es bastante más reducido que el de otros algoritmos de extracción populares como SIFT o SURF, lo que hace que sea idóneo para aplicaciones en tiempo real como es el SLAM. Además ORB consigue invarianza a rotación y a escala.

Para detectar los puntos de interés genera una pirámide de escalas compuesta por conjunto de imágenes a diferente resolución de la imagen original (figura 3.3). A continuación se aplica el algoritmo FAST de detección de puntos de interés a cada una de las imágenes de la pirámide. Este algoritmo compara la luminosidad de cada píxel de la imagen con la de sus píxeles circundantes y, en base a un umbral, determina si

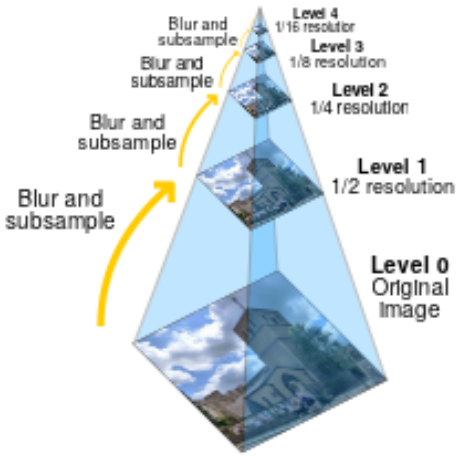

Figura 3.3: Pirámide de escalas. By Cmglee - Own work, CC BY-SA 3.0

ese píxel es un punto de interés. ORB también asigna una orientación a estos puntos FAST en base a la dirección de mayor cambio de luminosidad respecto al píxel central.

Para describir estos puntos utiliza el algoritmo BRIEF, que asigna un vector binario de 256 componentes a cada punto detectado anteriormente. En el caso de ORB, este descriptor también incluye información de la orientación asignada a cada punto FAST y no solo de la diferencia de la luminosidad con su entorno.

De esta manera, si un mismo punto de la escena aparece en 2 imágenes diferentes, con diferente escala y/u orientación, podremos emparejarlo sin problemas.

3.2. Modificaciones a ORBSLAM-Atlas

3.2.1. Aumento de la tasa de inserción de KeyFrames

La inserción de un nuevo KeyFrame es algo crítico para la construcción del mapa, ya que su geometría se reconstruye a partir de las correspondencias entre KeyFrames. Como ya se ha visto, las restricciones para la inserción de un KeyFrame buscan que estos no sean redundantes en cuanto a la información que aportan al mapa y que aporten un mínimo de información como para considerarlos KeyFrames.

Sin embargo, se debe tener en cuenta que en este trabajo se está aplicando ORBSLAM-Atlas a un tipo de escena que no es el habitual y, por lo tanto, tiene sentido replantearse algunas de las restricciones impuestas.

En el caso de una endoscopia el cambio de la escena observada por la cámara entre una imagen y otra es mucho mayor de lo que podría ser en una secuencia de, por ejemplo, el interior de un edificio. Esto se debe a la propia naturaleza de los movimientos que se realizan en una secuencia médica exploratoria y a que la distancia entre el endoscopio y las superficies grabadas es muy pequeña en relación a la escena.

Eso nos lleva a pensar que el número mínimo de frames para la inserción de un nuevo KeyFrame puede ser excesivo en el caso de las endoscopias. De un KeyFrame a otro, la escena observada puede ser tan diferente que no seamos capaces de mapearla, ya que para calcular la posición 3D de los puntos mediante BA necesitamos emparejamientos de esos puntos entre los KeyFrames y, si la escena observada cambia en exceso, será imposible realizar estos emparejamientos ya que, simplemente, no estaremos viendo los mismos puntos en esos KeyFrames.

Por ello uno de los cambios que se ha realizado en el software es reducir ese mínimo de frames para la inserción de un nuevo KeyFrame de FPS a FPS/6 tanto para la primera como para la segunda condición.

Además, dado que el software tiene su propio algoritmo de eliminación de KeyFrames redundantes, no corremos el riesgo de excedernos en ese sentido. De hecho, la lógica que sigue el sistema de inserción de KeyFrames del software original es que su inserción, para el tipo de escenas habitual, sea generosa, ya que se tiene en cuenta ese posterior refinado de los KeyFrames totales.

3.2.2. Reducción del tamaño mínimo del mapa

En este caso el tamaño del mapa lo medimos en el número de KeyFrames que se han añadido a ese mapa antes de que se perdiera la localización.

En el software original un mapa tiene que tener un tamaño mínimo de 8 KeyFrames para no ser descartado. Esto se debe a que se considera que un mapa de menos de 8 KeyFrames no suficientemente relevante como para guardarlo. En las aplicaciones habituales de ORBSLAM-Atlas se consiguen mapas de unos pocos cientos de KeyFrames, por lo tanto, tiene sentido considerar que un mapa de menos de 8 KeyFrames pueda ser descartado.

Sin embargo, a la hora de procesar endoscopias con ORBSLAM-Atlas, el tamaño medio de lo mapas que obtenemos raramente supera los 20 KeyFrames, por lo tanto, en este caso, un mapa de 8 KeyFrames no es, en absoluto, irrelevante.

Por ello se ha eliminado esa condición de tamaño mínimo de los mapas de manera que todos los mapas generados son guardados.

3.2.3. El visualizador muestra la imagen a color

En el software original el frame actual que aparece en la ventana está en blanco y negro. Se ha modificado el visualizador de forma que ese frame aparezca en color. Este cambio es simplemente estético.

3.3. Preprocesado de las secuencias

Antes de procesar las secuencias con ORBSLAM-Atlas hemos tenido que preparar las imágenes en varios sentidos. Todas estas operaciones previas se han realizado a través de unos scripts relativamente simples pero necesarios, en los que básicamente se han utilizado librerías de OpenCV.

1. **Separación del vídeo en frames** Las endoscopias del dataset al que hemos tenido acceso no han sido adquiridas con el software EndoStore, el cual almacena directamente los frames de la secuencia, sino que se disponía del vídeo completo.

   Dado que ORBSLAM-Atlas procesa las secuencias por frames ha sido necesario separar estos vídeos en los frames que lo formaban.

2. **Reducción de la resolución de las imágenes** Durante el procesado de las secuencias con ORBSLAM-Atlas surgió el problema de que si la resolución de las imágenes era demasiado alta se producían fallos de ejecución debido al gran coste computacional de procesar todas las imágenes en alta resolución. Por ello se ha reducido la resolución de las imágenes originales con la modificación de los parámetros intrínsecos que eso conlleva, desarrollado en la sección 4.3

3. **Desdistorsión de los frames** Una parte imprescindible de este preprocesado ha sido la propia desdistorsión de las imágenes a partir de los parámetros de calibración. Estos frames desdistorsionados son los que se utilizaban como datos de entrada de ORBSLAM-Atlas.

4. **Generación de TimeStamp** En las aplicaciones robóticas en las que se ha usado habitualmente el SLAM las imágenes se procesan en tiempo real. Los sistemas de adquisición de imágenes en esos casos etiquetan todas las imágenes que se van grabando con un *TimeStamp*, una etiqueta temporal que utiliza el software de SLAM para procesar estas imágenes de manera secuencial.

   En nuestro caso no estamos trabajando en tiempo real y, tanto las imágenes que grabamos con el software EndoStore, como las que obtenemos de dividir los vídeos en frames, no llevan asociado este TimeStamp propio de sistemas en tiempo real. Por ello ha sido necesario generar este TimeStamp para todos los frames de las secuencias ya que, de otra manera, ORBSLAM-Atlas no es capaz de procesarlas.

Capítulo 4

Modelo de cámara de Kannala & Brandt

4.1. Proyección de la escena en la imagen

Los píxeles de una imagen nos aportan información sobre la dirección de los *rayos proyectantes*. Estos rayos proyectantes unen un punto real de la escena 3D, del cual tenemos su proyección en la imagen, con el centro óptico de la cámara que ha tomado la imagen (ver figura 4.1). Sin embargo, de esta manera, al proyectar una escena 3D en una imagen 2D perdemos la información sobre la profundidad de los puntos de la escena. Las coordenadas en píxeles de un punto de la imagen y la dirección de su rayo proyectante están relacionadas mediante un *modelo de cámara*.

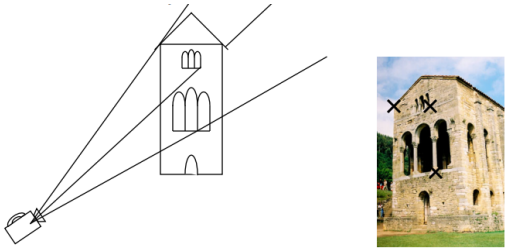

Figura 4.1: Representación de los rayos proyectantes

Un modelo de cámara consta de *parámetros intrínsecos*, que son los que definen las características físicas y geométricas de la cámara, y *coeficientes de distorsión*, que son los que modelan la desviación que experimentan los rayos proyectantes al atravesar la lente de la cámara debido a la no idealidad de ésta. La calibración de la cámara nos permite estimar los parámetros del modelo, lo que es realmente útil, ya que si los conocemos no solo conoceremos como se proyectan los puntos en la imagen, sino que podemos obtener información sobre la posición de los puntos en la escena a partir de sus coordenadas en la imagen.

Existen diversos tipos de cámaras según su geometría, características físicas de la

lente o su forma de proyectar los puntos de la imagen (ejemplos en figura 4.2). En nuestro caso particular, un endoscopio, tenemos una cámara de tipo *fish-eye*.

Las cámaras de tipo fish-eye se caracterizan por conseguir un gran ángulo de vista (incluso superior a 180º) utilizando un complejo sistema de lentes, no obstante este mismo sistema hace que la imagen proyectada sufra una gran distorsión.

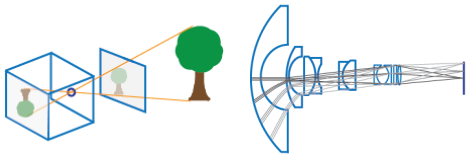

Figura 4.2: Representaciones de una cámara de tipo PinHole (izq) y Fish-eye (dcha). [4] © 1994-2020 The MathWorks, Inc.

4.2. Presentación del modelo de cámara de Kannala & Brandt

Uno de los múltiples modelos que describen la óptica de las cámaras Fish-Eye, y el que se ha utilizado en este trabajo, es el de Kannala & Brandt [5].

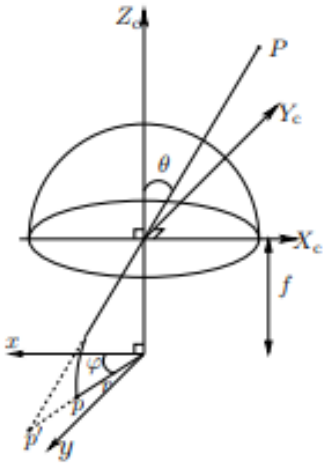

Figura 4.3: Modo de proyección de cámara Fisheye. La imagen del punto P es p mientras que en una cámara pinhole sería p'. [5] ©2006 IEEE

De hecho este modelo no se ajusta solo a cámaras de tipo Fish-Eye, sino que se propone un modelo de cámara genérico. El modelo de proyección propuesto es radialmente simétrico, sin embargo, también se hace una extensión considerando que, debido a imperfecciones geométricas o físicas de la lente o la cámara, la distorsión puede no ser radialmente simétrica. Las diferencias entre los resultados utilizando el modelo inicial o

modelo extendido son suficientemente pequeñas como para que se recomiende trabajar con el modelo reducido.

Este modelo describe las diferentes proyecciones de la siguiente forma:

$$r(\theta) = f(\theta + k_1\theta^3 + k_2\theta^5 + k_3\theta^7 + k_4\theta^9 + \ldots) \tag{4.1}$$

Donde θ es el ángulo que forma el rayo proyectante con el eje Z , perpendicular al plano de la imagen, *r(θ)* es la distancia desde el punto proyectado hasta el centro de la imagen y f es la distancia focal (ver figura 4.3).

Utilizando las 5 primeras potencias impares de θ se consiguen los suficientes grados de libertad para una buena aproximación de las diferentes curvas de proyección.

La transformación entre los rayos proyectantes y las coordenadas en la imagen se describe de la siguiente manera:

$$\begin{pmatrix} x \\ y \end{pmatrix} = r(\theta) \begin{pmatrix} \cos\varphi \\ \sin\varphi \end{pmatrix} = F(\Phi) \tag{4.2}$$

$\Phi = (\theta, \varphi)^T$ es la dirección del rayo proyectante (φ es el ángulo que forma con el eje X del plano de imagen y θ con el eje Z) (figura 4.3). Mientras que *r(θ)* contiene los 5 primeros términos y es monótonamente creciente en el intervalo $[0, \theta_{max}]$ donde θ_{max} es el ángulo de vista máximo de la cámara.

Y finalmente las coordenadas de la imagen han de transformarse de unidades de longitud a píxeles, esto se expresa en la siguiente ecuación:

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{bmatrix} m_u & 0 \\ 0 & m_v \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} c_u \\ c_v \end{pmatrix} \tag{4.3}$$

Donde $(u, v)^T$ son las coordenadas en el plano de la imgen en píxeles, m_u y m_v representan el número de píxeles por unidad de longitud en las direcciones horizontal y vertical respectivamente. Y $(u_0, v_0)^T$ son las coordenadas del centro de la imagen o *punto principal*.

Desarrollando la expresión de u y de v:

$$u = m_u f \left(\theta + k_1\theta^3 + k_2\theta^5 + k_3\theta^7 + k_4\theta^9\right)\cos(\varphi) + c_u \tag{4.4}$$

$$v = m_v f \left(\theta + k_1\theta^3 + k_2\theta^5 + k_3\theta^7 + k_4\theta^9\right)\sin(\varphi) + c_v \tag{4.5}$$

Si tenemos en cuenta la transformación de la distancia focal en unidades de longitud a píxeles y la no idealidad de las cámaras, que hace que la distancia focal en píxeles en dirección vertical y horizontal no coincida:

$$m_u f = f_u \quad (4.6)$$
$$m_v f = f_v \quad (4.7)$$

Por lo tanto sustituyendo (4.6) y (4.7) en (4.4) y (4.5) respectivamente:

$$u = f_u \left(\theta + k_1\theta^3 + k_2\theta^5 + k_3\theta^7 + k_4\theta^9\right)\cos(\varphi) + c_u \quad (4.8)$$
$$v = f_v \left(\theta + k_1\theta^3 + k_2\theta^5 + k_3\theta^7 + k_4\theta^9\right)\sin(\varphi) + c_v \quad (4.9)$$

Por lo que el modelo consta de 8 parámetros, los 4 *coeficientes de distorsión* (k_1, k_2, k_3, k_4) de la ecuación (4.1), que modelan la distorsión, y los 4 parámetros de la transformación de sistema de unidades, o *parámetros intrínsecos*, (f_u, f_v, c_u, c_v) de (4.8) y (4.9).

4.3. Transformación de los parámetros con el cambio de resolución

En ocasiones puede ser interesante trabajar con imágenes de una resolución diferente a la grabada originalmente por una cámara. Por ejemplo, nuestro caso, a la hora de la calibración nos interesa que la imagen tenga una alta resolución para que el software que se esté utilizando detecte los puntos con más precisión, pero para la aplicación de SLAM el trabajar con imágenes de alta resolución hace que el coste computacional sea demasiado alto y perjudique a la ejecución del programa.

Por ello es interesante conocer en qué afecta este cambio de resolución a los parámetros del modelo.

Desde el punto de vista de la distorsión, es decir, de como afecta la física de las lentes a la trayectoria de los rayos que las atraviesan, los parámetros no cambian ya que eso no depende del número de píxeles que utilicemos para proyectar la imagen, sino de la cámara con la que tomamos las imágenes y, en este caso, no estamos cambiando la cámara sino la resolución de las imágenes tomadas por ella.

Lo que realmente hacemos al reducir o aumentar la resolución es modificar los parámetros intrínsecos de la cámara, ya que al variar el número de píxeles, la transformación de unidades entre longitud y píxeles cambiará.

Los parámetros m_u y m_v de la ecuación 4.3 dan cuenta del número de píxeles que tenemos por unidad de longitud, y se calculan dividiendo el número de píxeles entre la longitud del plano de la imagen en cada una de las direcciones de los ejes. Las

dimensiones del plano de la imagen también son algo invariable ya que, al igual que las lentes, es una característica de la cámara, por lo tanto lo único que varía es el número de píxeles.

$$m_{u1} = N_x^1/x \tag{4.10}$$
$$m_{v1} = N_y^1/y \tag{4.11}$$
$$m_{u2} = N_x^2/x \tag{4.12}$$
$$m_{v2} = N_y^2/y \tag{4.13}$$
$$\tag{4.14}$$

Donde N_x^i y N_y^i representan el número de píxeles en cada una de las direcciones del plano de imagen de la imagen i y x e y representan la longitud del plano en ambas direcciones.

Si consideramos un cambio de resolución entre la imagen 1 y 2 en el que el número de píxeles en la dirección del eje X cambia de la siguiente manera:

$$N_x^1 = aN_x^2 \tag{4.15}$$

Sustituyendo en la ecuación (4.10):

$$m_{u1} = aN_x^2/x \tag{4.16}$$

Si ahora sustituimos m_u de la ecuación 4.6 por 4.16 y 4.11 y tenemos en cuenta que la distancia focal, f, también es un parámetro físico de la cámara y por lo tanto es la misma para ambos casos:

$$f\frac{aN_x^2}{x} = f_{u1} \tag{4.17}$$
$$f\frac{N_x^2}{x} = f_{u2} \tag{4.18}$$

Haciendo el cociente entre 4.17 y 4.18 y reordenando:

$$f_{u2} = \frac{f_{u1}}{a} \tag{4.19}$$

El desarrollo para hallar la relación entre f_{v1} y f_{v2} es análogo al anterior por lo que podemos extrapolar el resultado de 4.19:

$$f_{v2} = \frac{f_{v1}}{b} \tag{4.20}$$

Donde b representa el cambio de resolución en el eje Y de la imagen.

Para las coordenadas del centro de la imagen, $(u_0, v_0)^T$, se busca que la distancia relativa entre el centro y los bordes de la imagen se mantenga, por lo tanto, estas coordenadas cambiarán en la misma proporción que el número de píxeles:

$$c_{u2} = \frac{c_{u1}}{a} \tag{4.21}$$

Análogamente:

$$c_{v2} = \frac{c_{v1}}{b} \tag{4.22}$$

4.4. Calibración con patrones 2D

Una vez conocido el modelo se debe elegir un método de calibración para estimar sus parámetros. Hay diversos métodos de calibración dependiendo, por ejemplo, del tipo de patrones utilizado o del algoritmo implementado.

En el presente trabajo hemos calibrado mediante un método en el que se utilizan patrones planos y rígidos, que nos aporta la ventaja de que el diseño y fabricación de los patrones es realmente sencilla y versátil. Este proceso de calibración consta de dos partes:

En la primera de ellas se calcula una solución inicial de los parámetros intrínsecos que facilitará que el proceso de optimización no lineal, BA, de la segunda parte converja.

Para esta primera parte el especialmente relevante el concepto de ***homografía***.

Dado un conjunto de puntos, $\mathbf{x}_i$, en $\mathbb{P}^2$ y un conjunto de puntos correspondientes a los primeros, $\mathbf{x}'_i$, también en $\mathbb{P}^2$, hay una transformación proyectiva entre $\mathbf{x}_i$ y $\mathbf{x}'_i$ conocida como ***homografía***.

$$\begin{pmatrix} x'_i \\ y'_i \\ 1 \end{pmatrix} = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} \tag{4.23}$$

$$\mathbf{x}'_i = \mathbf{H}\mathbf{x}_i \tag{4.24}$$

Para esta primera parte establecemos la referencia del sistema de coordenadas global en una de las esquinas del patrón para conseguir que todos los puntos del patrón estén en el mismo plano y su tercera componente, que da cuenta de la profundidad, sea nula. Además como el patrón es de dimensiones conocidas conoceremos las coordenadas 3D

de los puntos del patrón.

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} f_u & 0 & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \tag{4.25}$$

Donde la primera matriz, $\mathbf{K}$, es la matriz de parámetros intrínsecos y la segunda matriz, $[\mathbf{R}|\mathbf{t}]$, compuesta por una matriz de rotación 3x3 y vector de traslación, es la matriz de parámetros extrínsecos que lleva los puntos del sistema de referencia global al sistema de referencia de la cámara.

Si todos los puntos están en el plano $z = 0$, la ecuación (4.25) queda de la siguiente manera:

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} f_u & 0 & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & t_1 \\ r_{21} & r_{22} & t_2 \\ r_{31} & r_{32} & t_3 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{4.26}$$

De esta manera reducimos la transformación entre las coordenadas 3D de los puntos del patrón y las coordenadas 2D del plano de imagen a una transformación entre dos planos, es decir, una homografía. Por ello es especialmente importante que el patrón sea lo más plano posible, ya que, de otra manera, no se podría utilizar una homografía. Además se debe hacer una estimación inicial de los parámetros de distorsión que, aunque de una manera poco precisa, nos permita realizar una desdistorsión inicial de las imágenes del patrón que posibilite el cálculo de la homografía.

$$\mathbf{H} = \begin{pmatrix} f_u & 0 & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & t_1 \\ r_{21} & r_{22} & t_2 \\ r_{31} & r_{32} & t_3 \end{pmatrix} \tag{4.27}$$

$$\mathbf{u} = \mathbf{H}\mathbf{x} \tag{4.28}$$

Esta homografía contiene la información conjunta sobre los parámetros extrínsecos y los intrínsecos, de forma que, con las operaciones adecuadas, podremos conocer estos últimos mediante el cálculo de la homografía [6].

Al calcular esta homografía también estaremos estimando los parámetros extrínsecos, pero en el caso de este trabajo no tienen especial interés ya que nuestro sistema está formado por una sola cámara. La calibración de estos parámetros es de gran utilidad en sistemas multicámara, p.ej. drones o cámaras estéreo, donde nos interesa conocer la posición relativa entre las cámaras que lo forman.

Teóricamente para poder estimar los parámetros intrínsecos hace falta resolver 3 homografías, es decir, se necesitarían 3 imágenes del patrón desde un ángulo de vista suficientemente diferente. Además por cada imagen del patrón se necesitan al menos

4 correspondencias entre puntos de la imagen y la escena. En la práctica, a la hora de grabar una secuencia de calibración, es importante grabar el patrón desde varios ángulos para tener suficientes imágenes ya que sería difícil asegurar que las 3 que se tomasen fuesen las ideales. Además los software de calibración refinan los parámetros resolviendo más de 3 homografías. También es importante que en las imágenes del patrón que se tomen para la calibración éste ocupe toda la imagen, de manera que se puedan estimar los parámetros correctamente, teniendo también en cuenta la proyección en los bordes que es, de hecho, donde más afecta la distorsión.

La segunda parte consiste en un proceso de optimización mediante BA. En el caso de la calibración de una sola cámara lo que más interés tiene del ajuste son los parámetros del modelo.

Como solución inicial de $\mathbf{p}$ se toman los parámetros del modelo calculados en la primera parte de la calibración.

4.5. Diseño de los patrones

Para la calibración del endoscopio se ha utilizado el software de calibración Vicalib, que se puede encontrar en el siguiente repositorio de GitHub [7]. Este software utiliza patrones conocidos como *circle grid pattern*, en los que los puntos de control son círculos (imagen 4.4).

Dado que la accesibilidad al endoscopio del Hospital Clínico Lozano Blesa iba a ser reducida había que intentar cubrir todas las posibilidades a la hora del diseño para poder optimizar al máximo el tiempo de trabajo con el endoscopio.

Para el diseño de estos patrones se utilizó una herramienta del propio software Vicalib destinada para ello. En nuestro caso, dado que se iba a calibrar un endoscopio y que éste graba a distancias muy pequeñas de las superficies, los patrones debían ser bastante más pequeños de lo que se suele usar normalmente. Se diseñaron 4 patrones de tamaños diferentes comprendidos entre 2'75x4'78 cm y 7'54x13'2 cm. Para asegurar su rigidez los patrones se imprimieron en papel grueso y posteriormente se plastificó un ejemplar de cada modelo, esto fue debido a que se barajó la posibilidad de que la

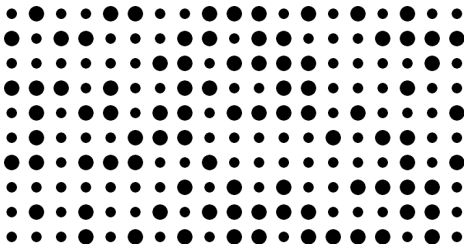

Figura 4.4: Circle grid pattern.

plastificación reflejara la luz que emite el endoscopio cuando está en funcionamiento y estos reflejos tapasen parte de los puntos del patrón, por ello era conveniente tener un ejemplar plastificado y otro sin plastificar.

4.6. Grabación de la secuencia de calibración

En la documentación de Vicalib no aparecen indicaciones específicas para la correcta grabación de una secuencia. Por ello, para la grabación de una secuencia de calibración, se siguieron las siguientes pautas:

- Durante la grabación se debe rotar la cámara respecto de sus 3 ejes para conseguir imágenes con un ángulo de vista suficientemente diferente.
- Se debe intentar ajustar, en la medida de lo posible, la imagen grabada al tamaño del patrón, de manera que ocupemos la mayor parte posible de la imagen viendo la mayor parte posible del patrón.
- El número de imágenes (duración de la secuencia) debe ser suficiente.

Capítulo 5

Validación experimental

A continuación se exponen la experimentación y resultados tanto de la parte de calibración del endoscopio como del procesado de las secuencias con ORBSLAM-Atlas.

5.1. Calibración

Primero se exponen los resultados de la parte de calibración en la que entran tanto los experimentos para encontrar el método óptimo de grabación como los resultados de la calibración del endoscopio mismo.

5.1.1. Calibración de cámara Fish-Eye

Antes de grabar la secuencia de calibración con el endoscopio se realizaron unas pruebas de calibración con otra cámara de tipo fish-eye disponible en el laboratorio de la cual conocemos los parámetros.

Para verificar la bondad del método de grabación propuesto en 4.6 se procesó la secuencia con Vicalib y se compararon los parámetros estimados con los que aporta el fabricante de la cámara. Siendo estos del mismo orden se procedió a desdistorsionar las imágenes de la secuencia, de esta forma se puede comprobar a simple vista si la desdistorsión ha sido correcta y, por lo tanto, los parámetros estaban bien estimados. Para la desdistorsión de la secuencia se utilizaron las librerías para cámaras Fisheye de OpenCV.

Los parámetros del modelo son los siguientes:

	f_u (px)	f_v (px)	c_u (px)	c_v (px)	k_1	k_2	k_3	k_4
Fabricante	220.2	220.26	363.33	242.67	0.0346	0.00621	0.000796	-0.00128
Vicalib	220.11	219.99	363.17	243.15	0.0416	0.00737	-0.000927	-0.00103

Tabla 5.1: Parámetros de la cámara del laboratorio

Y en las figuras 5.1 y 5.2 podemos observar un par de imágenes de la secuencia de

calibración y otro del laboratorio. En ambos casos aparece la imagen original distorisionada que toma la cámara y otra tras desdistorsionar en OpenCV con los parámetros anteriores:

Figura 5.1: Imágenes del laboratorio: distorsionada (izq) y tras corregir la distorsión (dcha)

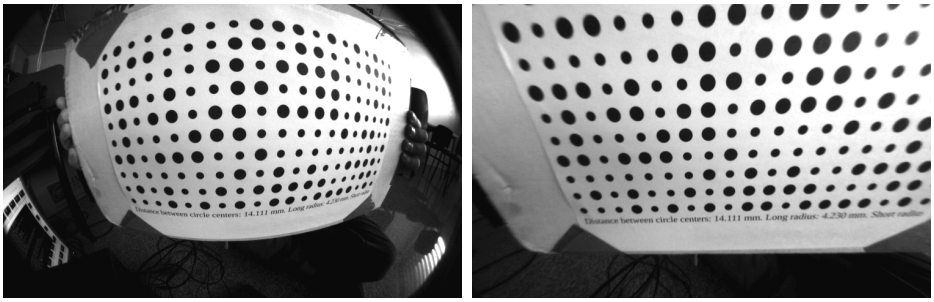

Figura 5.2: Imágenes del patrón: distorsionada (izq) y tras corregir la distorsión (dcha)

5.1.2. Calibración del endoscopio

Una vez fijado el método de grabación y con los patrones diseñados y fabricados se pudo proceder a la grabación de la secuencia de calibración del endoscopio. Para la grabación de la secuencia se reprodujo el método del laboratorio. El patrón utilizado fue uno de los plastificados, ya que se observó que el reflejo producido por el endoscopio no parecía demasiado significativo (como podemos ver en las imágenes de 5.4), y el tamaño escogido fue 5'61x9'82 cm. Las secuencias se almacenaron para su posterior procesamiento en el laboratorio. Para la grabación de la secuencia se utilizó el sistema EndoStore [8].

Además se grabaron varias secuencias de la sala de endoscopias para tener material con el que trabajar posteriormente para, por ejemplo, comprobar la calibración del endoscopio mediante la desdistorsión de estas secuencias o emplear técnicas de SLAM con ellas. Todas las secuencias grabadas tienen una resolución de 720x540.

A continuación se presentan los resultados de calibración del endoscopio. Primero los parámetros obtenidos con Vicalib y en la figuras 5.3 y 5.4 una imagen tomada con

el endoscopio de la sala de endoscopias y su correspondiente imagen desdistorsionada y una imagen del patrón utilizado para la calibración del endoscopio y su correspondiente imagen desdistorsionada.

Los parámetros del modelo son los siguientes:

f_u (px)	f_v (px)	c_u (px)	c_v (px)	k_1	k_2	k_3	k_4
396.37	396.05	374.67	265.28	-0.131	-0.0115	0.0131	-0.00571

Tabla 5.2: Parámetros del endoscopio

Figura 5.3: Imágenes del endoscopio: distorsionada (izq) y tras corregir la distorsión (dcha)

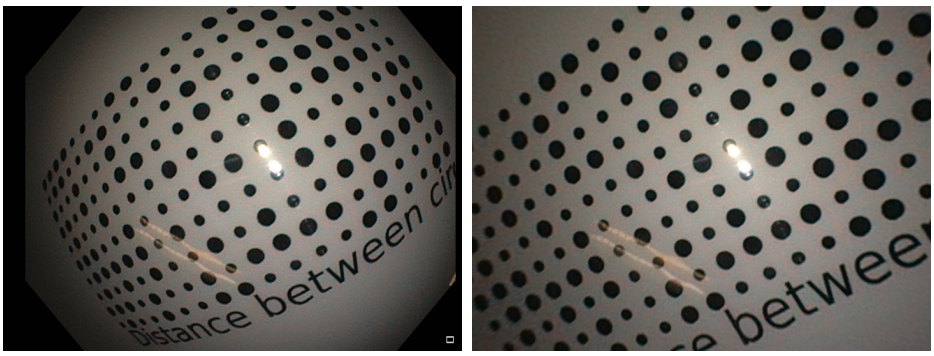

Figura 5.4: Imágenes del patrón de calibración del endoscopio: distorsionada (izq) y tras corregir la distorsión (dcha)

5.2. SLAM

Durante el procesado de las secuencias se han ido variando los parámetros del extractor ORB. Se ha buscado que el número de puntos de interés extraídos de cada frame fuese el máximo para tener la mayor cantidad de información posible a la hora de mapear.

Los parámetros del extractor que se pueden variar son los siguientes:

– El número máximo de puntos de interés a detectar en cada frame.

– El número de escalas a las que procesamos el frame.

– El factor de escala entre los niveles de la pirámide de escalas.

– Un *FAST threshold* que representa un valor mínimo para considerar un punto como punto de interés. Cuanto más bajo sea este valor más puntos obtendremos, pero si es demasiado bajo se corre el riesgo de que esos puntos no sean realmente diferentes a los de su entorno. Este *threshold* tiene 2 valores, uno inicial (*InitThreshold*) que se utiliza en una primera búsqueda de puntos y un mínimo (*MinThreshold*) que se utiliza si con el inicial se han encontrado muy pocos puntos.

A continuación se presentan los experimentos y resultados del procesado con ORBSLAM-Atlas de las secuencias de los endoscopios. En la tabla 5.3 aparecen en los parámetros del extractor ORB utilizados para: una secuencia de un entorno urbano grabada con una cámara genérica (primera fila); La secuencia no médica grabada con el endoscopio calibrado (segunda fila); Y la secuencia médica grabada con el endoscopio no calibrado (tercera fila).

5.2.1. Procesado de secuencia no médica grabada con endoscopio de calibración conocida

Con el endoscopio del Hospital Clínico Lozano Blesa al que se ha tenido acceso en este trabajo se grabaron varias secuencias, una de calibración y otra de la sala de endoscopias. No se ha podido trabajar con secuencias médicas reales grabadas con el endoscopio calibrado porque, debido al efecto que ha tenido la situación sanitaria extraordinaria en el hospital, no se han podido realizar los trámites administrativos para conseguir lo permisos éticos necesarios.

En la secuencia que se ha procesado se ha grabado la sala de endoscopias del hospital, concretamente parte del hardware del endoscopio. La secuencia procesada

	Nº puntos	Nº niveles	Factor de escala	InitThreshold	MinThreshold
Secuencia genérica	1000	8	1.2	20	7
Secuencia de endoscopio no médica	5000	8	1.2	20	7
Secuencia de endoscopio médica	10000	12	1.2	5	5

Tabla 5.3: Parámetros del extractor ORB para distintos tipos de secuencia

tiene una resolución de 720x540.

Los parámetros del extractor ORB utilizados para el procesado de la secuencia no médica grabada con el endoscopio aparecen en la segunda fila de la tabla 5.3

En la figura 5.5 podemos ver varias vistas del mapa realizado y varias imágenes de la escena mapeada, en este caso la botonera del endoscopio en la sala de endoscopias. En azul aparece la trayectoria que ha seguido la cámara (KeyFrames). Los puntos rojos y negros son los puntos del mapa. 2 de las vistas del mapa se corresponden a la parte frontal de la botonera del endoscopio donde se pueden intuir parte de los botones y mandos así como las formas cuadrangulares del mismo. La otra es una vista lateral del mapa donde se puede apreciar que los puntos del mapa forman las superficies rectas de la botonera.

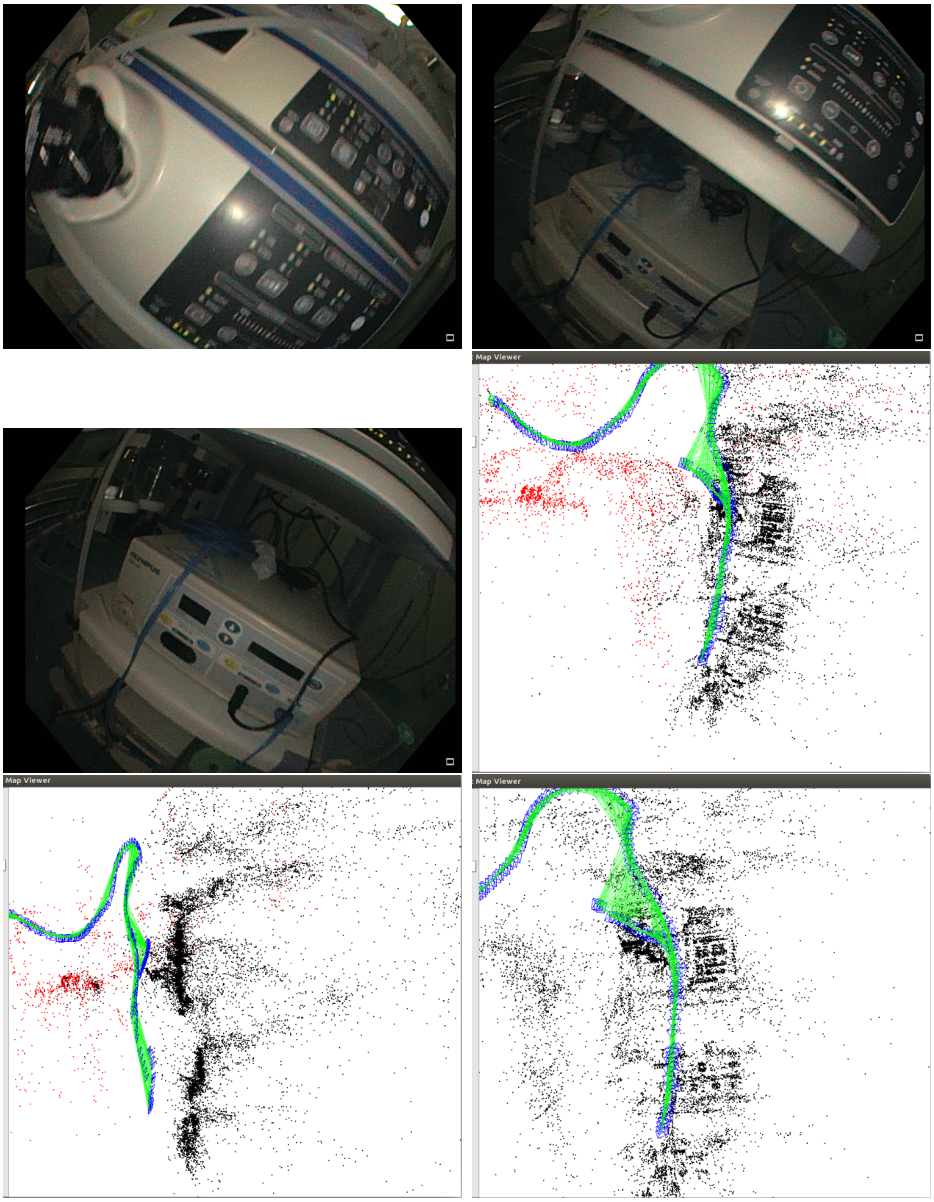

Figura 5.5: Imágenes del hardware del endoscopio y su mapa de puntos.

En este caso el software mapea la secuencia, que tiene un total de 474 frames, en un solo mapa ya que no se pierde y no necesita inicializar ningún mapa más. Se ha mapeado casi la totalidad de la secuencia, salvo unos pocos segundos al inicio de la secuencia en los que el software inicializa el mapa.

5.2.2. Procesado de secuencia médica grabada con endoscopio de calibración desconocida

Aunque no se han podido grabar secuencias médicas con el endoscopio que hemos calibrado sí que se ha tenido acceso a secuencias médicas grabadas con otros endoscopios.

El problema que surge es que no conocemos la calibración de los endoscopios que han grabado las secuencias lo que, en un principio, haría imposible la desdistorsión de la imágenes para su posterior procesado. Sin embargo se ha considerado la idea de que los endoscopios pueden tener una óptica bastante parecida y, por lo tanto, la forma de distorsionar las imágenes captadas puede ser similar. Por ello como posible solución a este problema surge la idea de utilizar los parámetros de calibración obtenidos en la calibración del endoscopio del HCLB para desdistorsionar las imágenes de las secuencias médicas del dataset al que tenemos acceso. Tanto las distancias focales como las coordenadas del centro de la imagen se han transformado según lo expuesto en la sección 4.3, teniendo en cuenta que la resolución de las imágenes del dataset es de 624x540 y la del endoscopio del que conocemos la calibración es de 720x540.

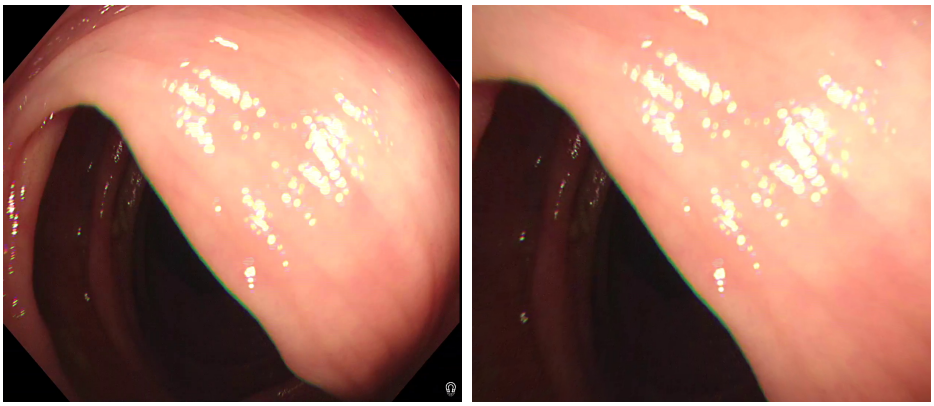

Figura 5.6: Imágenes de colonoscopia: distorsionada (izq) y tras corregir la distorsión (dcha)

Para comprobar lo anterior se ha procedido a la desdistorsión de la secuencia cuyo resultado podemos ver en las figuras 5.6 y 5.7. En este tipo de secuencias médicas la distorsión es menos visible porque las paredes del intestino, que son las zonas en el borde de la imagen donde la distorsión es mayor, tienen curvatura y el efecto de la

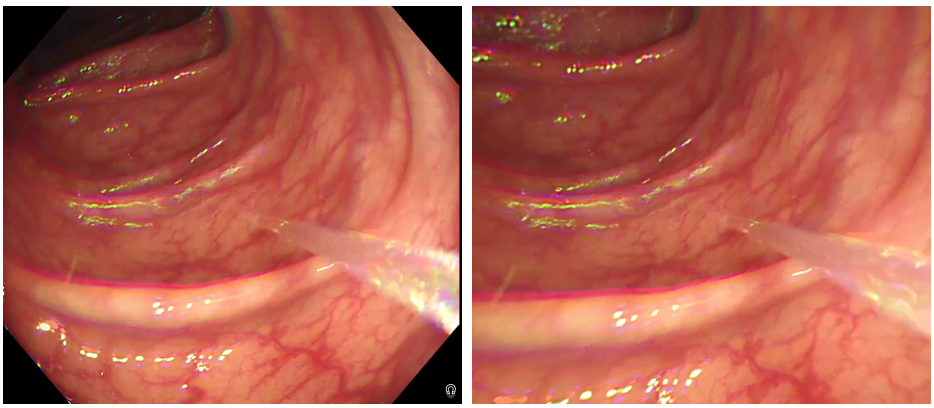

Figura 5.7: Imágenes de colonoscopia: distorsionada (izq) y tras corregir la distorsión (dcha)

distorsión se manifiesta modificando esa curvatura. Esta modificación de la curvatura es difícil de apreciar a simple vista pero su efecto en la reconstrucción de la escena es significativo.

A pesar de lo anterior, podemos concluir que la desdistorsión es suficientemente buena como para proceder a trabajar con estas secuencias.

Como ya se ha comentado, en las escenas médicas la extracción de puntos de interés es bastante complicada debido a la homogeneidad de las mismas. Por ello el ajuste de los parámetros del extractor ha ido encaminado a extraer el mayor número de puntos posible para tener suficiente información a la hora mapear. En comparación con lo que serían los parámetros para una secuencia "habitual", en nuestro caso hemos aumentado el número de puntos máximo extraído en cada imagen, el número de niveles de la pirámide de escalas y hemos reducido el *FAST threshold* para ser más permisivos a la hora de identificar un punto de la imagen como KeyPoint. Los parámetros del extractor ORB utilizados para el procesado de la secuencia médica aparecen en la tercera fila de la tabla 5.3.

En este caso, para una secuencia de 1139 frames y 86 segundos de duración, después de 10 ejecuciones, obtenemos las siguientes estadísticas del número de mapas obtenidos:

Media	Máximo	Mínimo
6.7	9	5

Tabla 5.4: Media, máximo y mínimo de número de mapas para el mapeado de la secuencia médica

A continuación se muestra información sobre los mapas de una ejecución típica de ORBSLAM-Atlas para la secuencia médica en la que se han obtenido 7 mapas. En la tabla 5.5 aparece información sobre el tamaño de los mapas y el instante de la secuencia en que empiezan y acaban.

Mapa	Tamaño (KeyFrames)	t_o (s)	t_f (s)
1º	18	2	12
2º	7	25	26
3º	7	30	31
4º	9	39	42
5º	4	46	47
6º	2	58	60
7º	8	81	86

Tabla 5.5: Tamaño de los mapas e instantes de inicio y final

En las figuras 5.8 a 5.13 podemos ver uno de los últimos frames de cada uno de los mapas y alguno de los inmediatamente posteriores. De esta forma es fácil apreciar la influencia de los movimientos bruscos de la cámara a la hora de perder la localización. No se ha hecho con el último mapa ya que permanecía activo al final de la secuencia. En el quinto mapa se puede observar que la pérdida se produce por un cambio de color de la imagen debido, posiblemente, a algún defecto de grabación, en este caso no hay un movimiento brusco. En el quinto mapa, que dura 1 segundo aproximadamente, se puede observar perfectamente este efecto de los movimientos bruscos, ya que en este caso el mapa se ha inicializado justo antes de uno de estos movimientos y la pérdida de localización se produce claramente durante el movimiento.

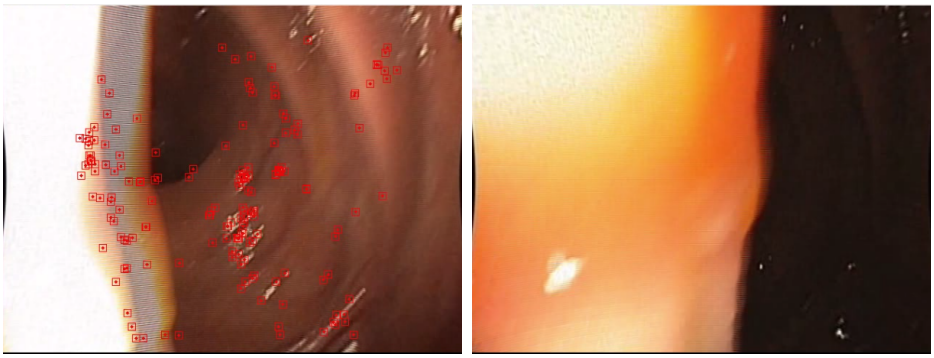

Figura 5.8: Final del mapa 1 y frame posterior

Y en la gráfica 5.14 podemos ver en que momentos de la secuencia se ha estado mapeando.

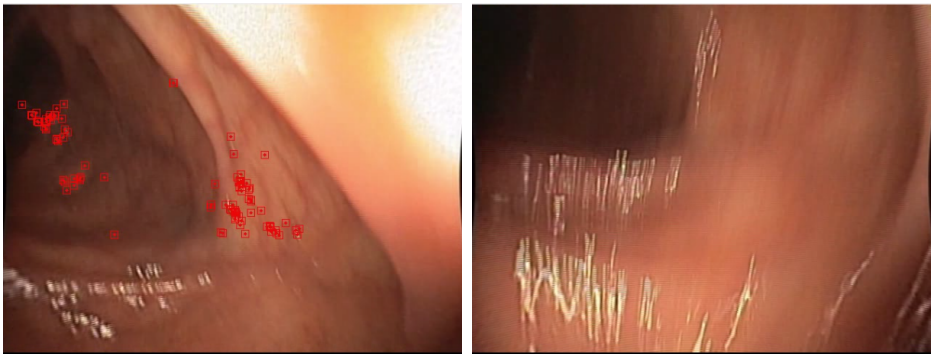

Figura 5.9: Final del mapa 2 y frame posterior

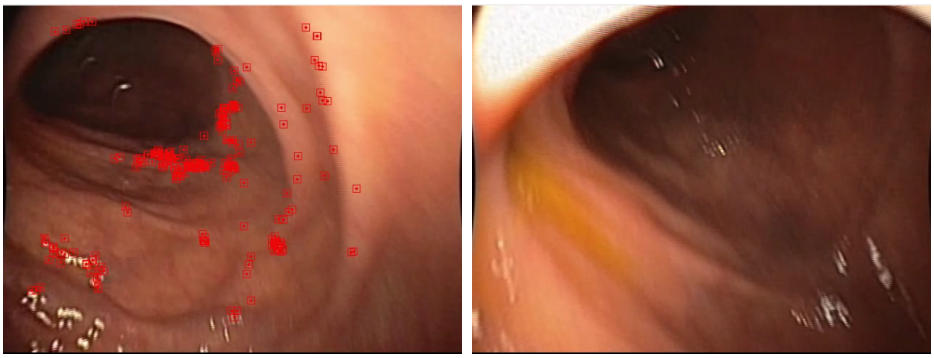

Figura 5.10: Final del mapa 3 y frame posterior

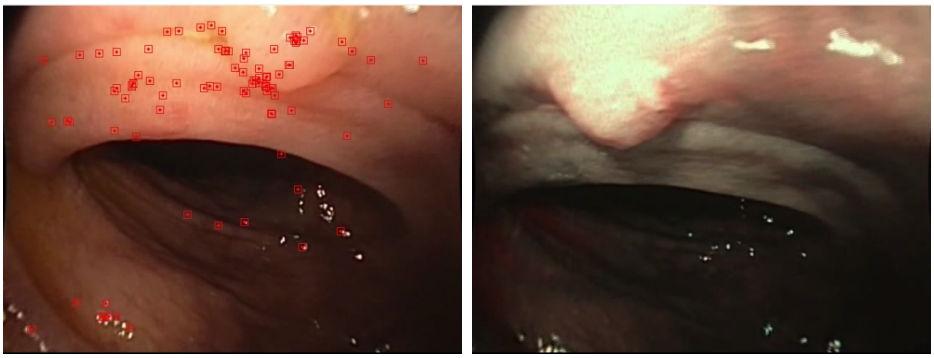

Figura 5.11: Final del mapa 4 y frame posterior

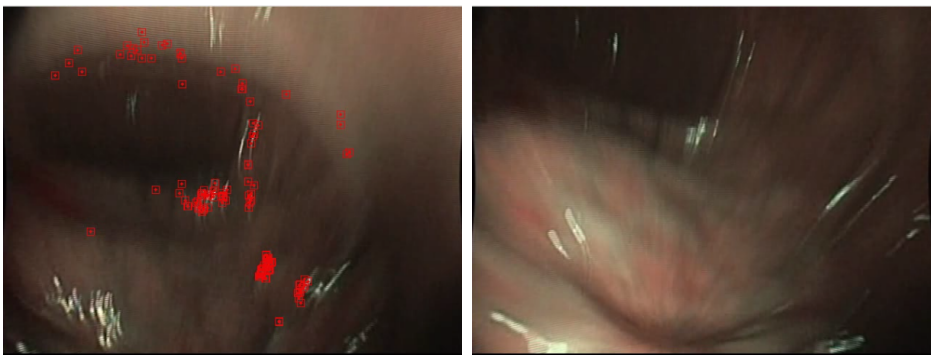

Figura 5.12: Final del mapa 5 y frame posterior

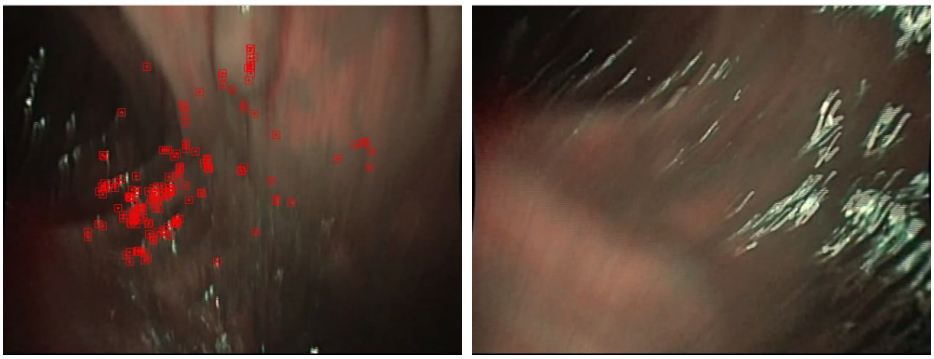

Figura 5.13: Final del mapa 6 y frame posterior

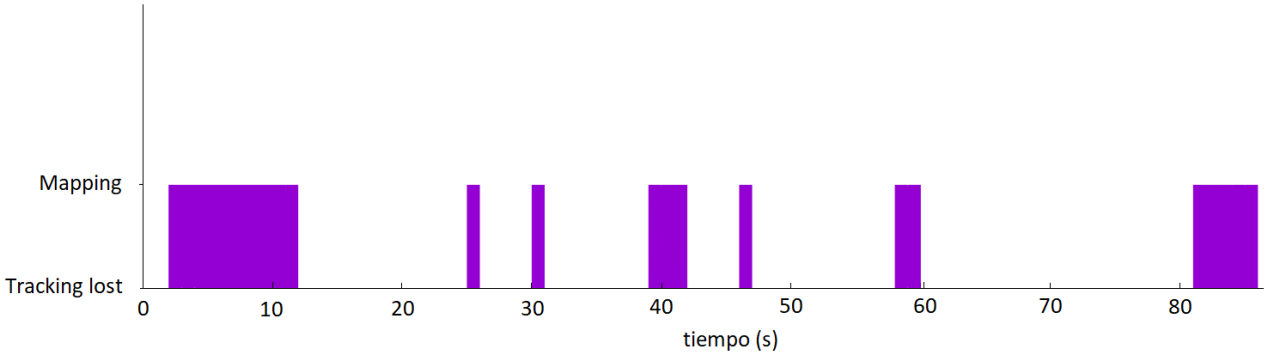

Figura 5.14

Capítulo 6

Conclusiones y trabajo futuro

6.1. Calibración del endoscopio

A la vista de los resultados obtenidos en la parte de calibración, podemos concluir que la óptica de un endoscopio se corresponde con la de una cámara Fish-Eye y el modelo de cámara de Kannala & Brandt la describe correctamente.

Si comparamos las imágenes 5.1 5.2, captadas con una cámara Fish-Eye en la laboratorio, con la imagen 5.3 podemos ver que la manera de distorsionar la escena es similar. Además, la desdistorsión de la imágenes realizada a partir de los parámetros obtenidos del calibrado de dicho modelo, nos devuelve unas imágenes poco o nada distorsionadas. Una forma de valorar esta desdistorsión es comprobar que las líneas rectas no se curvan en los extremos, lo que podemos ver en la figura 5.3.

Otro aspecto que nos sirve para valorar la adecuación del modelo de cámara elegido es el hecho de que se haya podido extrapolar la calibración de un endoscopio a otro y hayamos podido desdistorsionar las imágenes (figuras 5.6 y 5.7).

Todo esto justifica que los endoscopios tienen una óptica Fish-Eye lo que, además, tiene sentido, dado que en una secuencia exploratoria, no solo de tipo médico, nos interesa que el ángulo de vista de la cámara que se utilice sea amplio.

6.2. Aplicación de SLAM a endoscopias

Como podemos ver en la tabla 5.4 el número de mapas necesarios para el mapeado de una secuencia médica es mucho mayor que para otro tipo de secuencia, incluso para una secuencia no médica grabada con un endoscopio.

La creación de un nuevo mapa se lleva a cabo en el momento en que se pierde el *tracking* en el mapa que se esté realizando y no se consigue hacer una relocalización en en ese mapa, algo que, a la vista de las figuras 5.8 ... 5.13, sucederá con bastante probabilidad. Esto significa que cada vez que se ha creado un mapa se ha dado una

situación en la que, con un sistema que no fuese multimapa, se habría perdido por completo el mapeado. Por lo tanto la elección de un sistema de SLAM multimapa, como ORBSLAM-Atlas, se presenta como algo imprescindible a la hora de aplicar SLAM a secuencias médicas.

Como hemos visto en el primer capítulo, la extracción de puntos de interés y el posterior emparejamiento de esos puntos entre imágenes en forma de correspondencias, es la base de la reconstrucción 3D. Uno de los problemas que se presentaban a la hora de realizar este trabajo era la deficiente extracción de puntos en escenas médicas debido a la homogeneidad de las mismas. La solución que se ha implementado ha sido modificar los parámetros del extractor utilizado y, de hecho, se ha conseguido mejorar en cierta medida esa extracción de puntos.

Sin embargo cabe plantearse otras posibles soluciones al problema de la extracción de puntos de interés y el emparejamiento de los mismos. El algoritmo FAST de detección de puntos presenta 2 problemas principales a la hora de aplicarlo a secuencias médicas:

- El número de puntos extraído es reducido. Lo que implica, en cualquier caso, menos información para el mapeado.

- La repetitividad de los puntos detectados por el algoritmo es baja incluso para imágenes de la misma escena. Esto afecta directamente al cálculo de correspondencias. Ese cálculo se realiza comparando los descriptores BRIEF de los KeyPoints de unas imágenes con otras y, si esos descriptores son suficientemente similares, se establecerá una correspondencia. Si en 2 imágenes próximas de la secuencia, en las que la escena captada es similar, los puntos detectados como KeyPoints no se corresponden al mismo punto de la escena, esta forma de cálculo de correspondencias no va a dar buenos resultados y, por lo tanto, la reconstrucción 3D basada en ellas empeorará.

Por ello, una solución que se presenta, es la de utilizar otro algoritmo de detección de puntos diferente a FAST, cuyas características sean más apropiadas para el tipo de imágenes que tenemos en las secuencias médicas. Si se aumenta el número de puntos detectado y/o la repetitividad de la detección estaríamos dando solución a los 2 problemas anteriores.

Otra idea que se puede plantear, si decidimos no cambiar de detector de puntos, es cambiar el algoritmo de emparejamiento. En vez de basarlo en la comparación exhaustiva entre los descriptores de KeyPoints de imágenes consecutivas, se podría implementar un algoritmo que detectase KeyPoints en una imagen y, mediante la estimación del movimiento de la cámara, predijera donde van a aparecer esos puntos en

las imágenes próximas y centrase la detección de puntos a esas zonas. De esta forma estaríamos forzando un aumento de la repetitividad.

Otra funcionalidad de ORBSLAM-Atlas que también podría tener sentido adaptar a secuencias médicas es el *Merging* de los mapas. Como hemos visto una de las condiciones que se debe cumplir para que se realice la fusión de los mapas es que exista una transformación rígida entre 2 KeyFrames que hayan sido seleccionados como candidatos. El problema viene de que el interior del cuerpo humano no es rígido, como se puede observar en cualquier endoscopia. Esto puede ocasionar que, aunque el algoritmo esté proponiendo 2 KeyFrames como candidatos debido a sus similitudes, los mapas, rara o ninguna vez, lleguen a fusionarse porque no se estará cumpliendo la condición de rigidez.

En cuanto a las imágenes que utilizamos como entrada, se puede observar en la sección 5 que al desdistorsionar las imágenes originales perdemos gran parte de la información visual. Esto, obviamente, reduce la eficacia del mapeado ya que estamos trabajando con menos información de la que capta la cámara en un inicio. Una manera de subsanar este problema sería añadir el procesamiento de imágenes captadas con una cámara Fish-Eye a ORBSLAM-Atlas, aportando como dato de entrada la calibración de la misma, para de esta manera no perder una información de la escena realmente útil para el proceso de mapeado.

En definitiva, podemos concluir que la utilización de un sistema de SLAM multimapa para el procesado de secuencias médicas es algo realmente interesante por la naturaleza de las mismas. Sin embargo, hay una gran línea de trabajo abierta en cuanto a la adaptación de muchas de las características del sistema a este tipo de secuencias.

Bibliografía

[1] Richard Elvira, Juan Tardos, and J.M.M. Montiel. Orbslam-atlas: a robust and accurate multi-map system. pages 6253–6259, 11 2019.

[2] Varios autores. *Conceptos y métodos en Visión por Computador*. 2016.

[3] Montiel J. M. M. Mur-Artal, Raúl and Juan D. Tardós. ORB-SLAM: a versatile and accurate monocular SLAM system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.

[4] Documentación fish-eye de mathworks. `https://www.mathworks.com/help/vision/ug/fisheye-calibration-basics.html`.

[5] Juho Kannala and Sami Sebastian Brandt. A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:1335–1340, 2006.

[6] Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.

[7] Repositorio de vicalib en github. `https://github.com/arpg/vicalib`.

[8] Alexandro Delgado Llamas and José María Martínez Montiel. Adquisición de endoscopias en alta calidad. 2019.

[9] Documentación fish-eye de opencv. `https://docs.opencv.org/3.4/db/d58/group__calib3d__fisheye.html`.

[10] Richard Szeliski. *Computer Vision: Algorithms and Applications*. 2010.

[11] & Zisserman A. Hartley, R. *Multiple view geometry in computer vision*. Cambridge university press., 2003.

[12] Barbara Frank, Cyrill Stachniss, Giorgio Grisetti, Kai Arras, and Wolfram Burgard. Documentación sobre calibración de cámaras de la universidad de friburgo. `http://ais.informatik.uni-freiburg.de/teaching/ws10/robotics2/pdfs/rob2-10-camera-calibration.pdf`.

[13] Introduction to orb (oriented fast and rotated brief). https://medium.com/data-breach/introduction-to-orb-oriented-fast-and-rotated-brief-4220e8ec40cf.

Lista de Figuras

Lista de Tablas