Universidad Zaragoza
1542

Trabajo Fin de Máster en Ingeniería Industrial

Evaluación y procesamiento de escenas médicas con sistema VSLAM no rígido

Autor

Javier Morlana Ledesma

Director

José María Martínez Montiel

Codirector

José Lamarca Peiro

Escuela de Ingeniería y Arquitectura
2019

Resumen

En este proyecto se ha adaptado y evaluado un sistema de VSLAM (Simultaneous Location and Mapping with Visual sensor) deformable monocular para el procesamiento de escenas médicas. El objetivo es implementar un algoritmo que procese secuencias médicas, obteniendo un modelo 3D deformable de la escena, así como la posición relativa de la cámara respecto del modelo.

Para construir el software se modifica el algoritmo DefSLAM, un sistema VSLAM deformable monocular en etapa de desarrollo y que ha sido probado en una tela que se deforma. Se modifica y sintoniza para que pueda procesar secuencias médicas donde hay deformación.

Para ello se realiza una primera sintonía, de manera que se tenga el sistema en funcionamiento en escenas médicas. Se define una métrica propia para evaluar la calidad de la geometría estimada, por medio de la estimación estéreo de la superficie, calculada en cada imagen de la secuencia.

El sistema presenta algunas limitaciones que serán resueltas. Para reducir la deriva de escala, se propone modificar la política de inserción de keyframes. Para gestionar las oclusiones y reflejos que se puedan dar, se propone por un lado, incluir un prior de movimiento suave para la cámara, y por otro, la generación de unas máscaras que detecten las regiones ocluidas o que contengan reflejos, y eliminar los puntos detectados en ellas.

El sistema ha sido validado experimentalmente y es capaz de procesar las secuencias médicas seleccionadas, estimando las deformaciones.

La implementación se ha hecho en C++ y está disponible en un repositorio privado de GitHub.

Agradecimientos

El Trabajo de Fin de Máster se ha llevado a cabo con el apoyo financiero del Ministerio de Economía y Competitividad del Estado Español mediante el proyecto "DPI2017-91104-EXP:SLAM Visual Deformable para Endoscopia".

Agradezco a José María Fácil por el preprocesamiento de las máscaras de detección de herramientas. A Ignacio Cuiral, por la ayuda en la obtención de datos de calibración en el Hamlyn Dataset.

Índice

Capítulo 1

Introducción

Los métodos de VSLAM (Visual Simultaneous Localization and Mapping) monoculares han logrado avances significativos en los últimos años, considerando que la escena observada es rígida [1], [2], [3]. Estos sistemas consiguen reconstruir un mapa tridimensional del entorno, a partir de las imágenes que toma una cámara en movimiento que observa la escena. A su vez, estos algoritmos calculan la posición de la cámara respecto del mapa generado. La suposición de escena rígida funciona muy bien en algunos ámbitos industriales, donde la escena permanece estática. Sin embargo, esta consideración no es útil en escenas donde existen elementos deformables, como son las escenas médicas.

Alternativamente, existen también sistemas que no hacen esta suposición de rigidez, y por tanto, son capaces de procesar escenas deformables. El problema no rígido está subdeterminado para sistemas monoculares, por lo que es un problema más complejo y requiere de restricciones adicionales que se incluyen en forma de prior. La reconstrucción de escenas no rígidas ha sido conseguida en la literatura a través de dos métodos diferentes: Non-Rigid Structure-from-Motion (NRSfM) y Shape from Template (SfT) [4]. IsoNRSfM [5] es un método NRSfM que considera que la deformación de la escena es isométrica, asumiendo que la distancia geodésica entre dos puntos de una superficie se mantiene constante durante la deformación. IsoNRSfM toma un conjunto de imágenes y reconstruye la escena a partir de ellas, mientras que SfT toma un template tridimensional conocido y reconstruye la escena a partir de una única imagen. SfT no es capaz sobrevivir en secuencias exploratorias, ya que cuando se observa una nueva región es necesario cambiar la plantilla, mientras que en IsoNRSfM el cálculo del mapa es tan costoso que no puede funcionar a frecuencia de vídeo. Una combinación de ambos métodos sí que podría procesar secuencias exploratorias.

El sistema utilizado en este proyecto es DefSLAM [6], que combina ambos métodos. Se trata de un sistema VSLAM monocular capaz de procesar escenas deformables sin asumir rigidez, habiéndose sintonizado y validado para trabajar con una tela (Mandala)

que se deforma. Nuestro objetivo será poner en funcionamiento y evaluar este sistema por primera vez en secuencias médicas, como son las laparoscopias o endoscopias, que serán obtenidas del Hamlyn dataset. Este dataset ha sido escogido por disponer de datos de calibración, necesarios para procesar las escenas, así como secuencias estéreo. La información estéreo será crucial para evaluar las prestaciones del DefSLAM debido a que a partir de imágenes estéreo se puede obtener un Ground Truth (solución de referencia) muy preciso. En este caso, el Ground Truth estimado es la superficie de la escena respecto de la cámara para cada fotograma del vídeo.

La motivación que subyace en resolver problemas de SLAM en entornos deformables, es la facilidad con la que se podrían adaptar estos algoritmos a los procedimientos médicos existentes. Las inspecciones visuales mediante endoscopio y laparoscopio no suelen disponer de otro sensor que no sean cámaras, además de alguna herramienta para hacer pequeñas intervenciones, ya que la dimensión de estos instrumentos debe ser pequeña para poder introducirse en el cuerpo. La adición de un sistema SLAM no rígido a los equipos de inspecciones médicas les permitiría conocer en tiempo real la anatomía del paciente analizado, así como la posición de la cámara respecto de este. La información de la geometría de la escena respecto del instrumento es clave para cualquier interacción computerizada con la escena. En el futuro próximo, esta interacción computerizada se materializaría en forma de anotaciones de realidad aumentada. Un siguiente paso serían herramientas de asistencia a la navegación para los endoscopistas. En un horizonte temporal más largo, posibilitaría el empleo de robots autónomos en los procedimientos endoscópicos.

1.1. Objetivos

En este proyecto se va a adaptar y evaluar DefSLAM monocular en el procesamiento de secuencias médicas que presenten deformación. Será necesario poner en marcha el sistema en este tipo de secuencias, así como definir una métrica para poder cuantificar las prestaciones del sistema en términos de la calidad de la geometría estimada de la escena y de la posición de la cámara respecto de esta estimación. Una vez establecido el banco de pruebas, se investigará sobre cómo mejorar las prestaciones en las secuencias médicas seleccionadas.

Las secuencias a analizar se tomarán de la base de datos del Hamlyn Center de Londres [7]. En este dataset se recopilan secuencias de endoscopias y laparoscopias. Se dispone de dos secuencias in-vivo de un cerdo. La secuencia 1 muestra la cavidad abdominal y tiene carácter exploratorio, incluyendo deformaciones y oclusiones. La secuencia 2 muestra un corazón latiendo, mientras la cámara que observa la escena

permanece estática. Ambas secuencias disponen de información estéreo de manera que podamos evaluarlas.

La motivación de procesar en monocular es que la mayoría de las imágenes médicas disponibles son de este tipo, ya que los endoscopios monoculares se pueden miniaturizar mucho, al contrario que los estéreo, cuya precisión es proporcional a la separación entre ambas cámaras.

Los objetivos concretos son:

- Adaptación del sistema en secuencias médicas. Esto incluye el arranque del primer Template con el par estéreo y la modificación del detector de puntos.
- Definición de la métrica para evaluar la calidad del mapa, mediante el cálculo de una solución estéreo de referencia para el mapa local en cada frame. Evaluación inicial del sistema y estudio de áreas de mejora.
- Se propone como mejora la modificación de la política de inserción de KeyFrames para reducir la deriva de escala.
- Gestión de oclusiones y reflejos. Imposición del prior de movimiento suave para la cámara. Preprocesamiento de máscaras para eliminar reflejos y herramientas.
- Validación experimental en las secuencias seleccionadas.

Capítulo 2

Adaptación de DefSLAM a escenas médicas

En este capítulo se dará una breve explicación de los elementos principales de DefSLAM que son relativos a este proyecto, y se expondrá como se realiza la adaptación del sistema para escenas médicas.

2.1. DefSLAM

DefSLAM es un sistema VSLAM monocular capaz de reconstruir la escena mientras estima la deformación y calcula la posición de la cámara respecto de ella.

El algoritmo reconstruye la escena por medio de un mapa de puntos 3D observables cuya posición evoluciona a lo largo de la secuencia (Sec. 2.1.1). Los puntos se embeben en un template que representa la parte visualizada del mapa, gracias a la cual se puede estimar la posición de los puntos en cada imagen de la secuencia, mediante la optimización del error de reproyección y la energía de deformación. El Tracking es el encargado de estimar la posición de la cámara y la deformación de la escena (Sec. 2.1.2).

Por otro lado, el mapa crece conforme nuevas partes de la escena son observadas. Cuando esto ocurre, el template ha de ser actualizado para cubrir las nuevas regiones que son visitadas. Esta tarea es realizada por el Mapping (Sec. 2.1.3).

El sistema toma como entrada las imágenes de la secuencia, que en adelante se denominarán frames. Un tipo especial de frames son los keyframes, que representan el conjunto de frames es que permiten reconstruir la escena sin necesidad de otros frames intermedios. El Tracking procesará todos los frames de la secuencia, mientras que el Mapping solo procesa los keyframes.

Los principales componentes del sistema se explican a continuación.

2.1.1. Mapa

DefSLAM es capaz de procesar deformaciones por medio de un template $\mathcal{T}_k^t$, calculado para el keyframe k y cuya forma evoluciona en cada frame t de la secuencia. El template $\mathcal{T}_k^t$ consiste en una malla 2D triangular que se sitúa en el espacio 3D. Este template se compone de caras triangulares, formadas por nodos y por los arcos que los unen. La forma del template se define a través de la posición de sus nodos, modificando la posición de estos al estimar las deformaciones. Los nodos del template modifican su posición según el modelo de deformación que impone la ecuación 2.1.

Por otro lado, se tienen los puntos del mapa 3D, que son las observaciones del sistema. El sistema crea puntos 3D a partir de la detección y emparejamiento de keypoints a lo largo de la secuencia. Estos keypoints son ORB y se localizan en cada frame mediante el detector de esquinas multiescala oFAST, pudiéndose reconocer si son vistos de nuevo mediante un descriptor binario de 256 bits.

Los puntos del mapa se embeben en las caras del template, y se relacionan con los nodos de la malla a través de las coordenadas baricéntricas del punto del mapa respecto de los nodos de la cara en la que está incrustado.

Cuando se estima la deformación para el frame t, se calcula la nueva posición de los nodos. Tras esto, se actualiza la posición de los puntos del mapa mediante las coordenadas baricéntricas.

2.1.2. Deformable Tracking

El tracking se encarga de estimar la pose de la cámara $\mathbf{T}_{cw}^t$ y la posición de los puntos embebidos en el template $\mathcal{T}_k^t$ para cada frame t de la secuencia. A partir de la forma en reposo del template, un algoritmo SfT permite calcular la forma deformada y la pose de cámara (fig. 2.1).

En las secuencias para SLAM, es habitual no observar la escena completamente en cada frame, sino una parte de ella. Por ello, solo se procesará una parte del template $\mathcal{T}_k^t$ a la hora de estimar las deformaciones y la pose de la cámara, que corresponde con la zona visualizada del template en el frame t, y sus vecinos más cercanos. Esta zona la denominamos como mapa local $\mathcal{L}_k^t$. Los bordes de $\mathcal{L}_k^t$ se mantienen fijos en la optimización.

Para estimar la forma del mapa local y la pose de cámara, se reduce el error de reproyección y la energía de deformación, expresados en la siguiente función de coste:

$$\underset{\mathcal{L}_k^t, \mathbf{T}_{cw}^t}{\arg\min} \quad \varphi_d(\mathcal{I}^t, \mathbf{T}_{cw}^t, \mathcal{L}_k^t) + \varphi_e(\mathcal{L}_k^t, \mathcal{L}_k^{t-1}, \mathcal{T}_k) \tag{2.1}$$

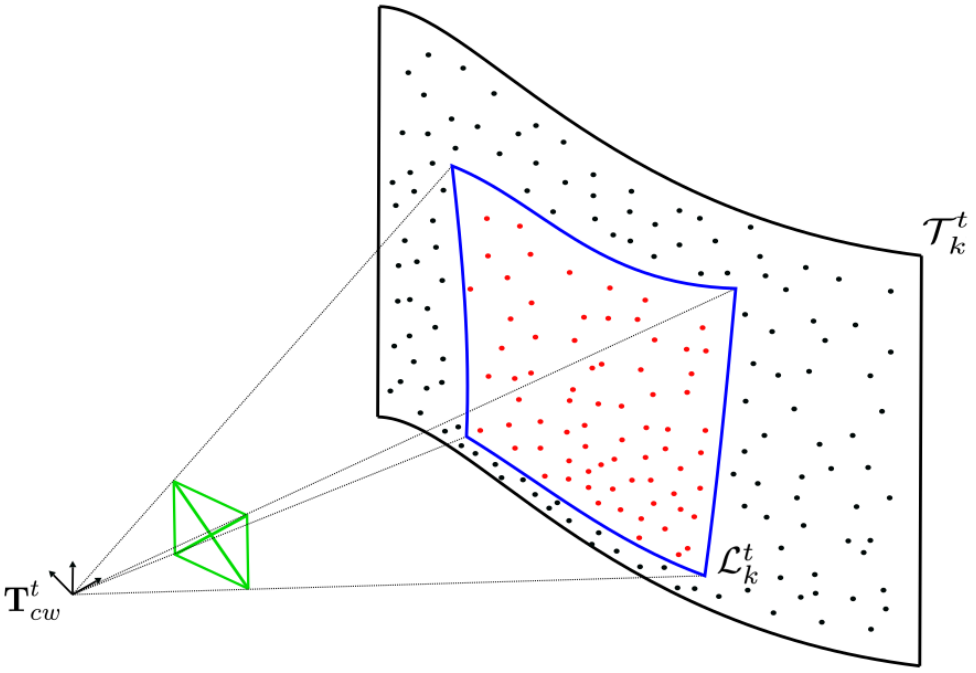


Figura 2.1: Cámara observando la escena en el frame t. $\mathbf{T}_{cw}^t$ es la pose de cámara, $\mathcal{T}_k^t$ es el template y $\mathcal{L}_k^t$ es el mapa local. El rectángulo verde representa la imagen observada.

El error de reproyección $\varphi_d(\mathcal{I}^t, \mathbf{T}_{cw}^t, \mathcal{L}_k^t)$ viene definido por:

$$\varphi_d(\mathcal{I}^t, \mathbf{T}_{cw}^t, \mathcal{L}_k^t) = \sum_{j \in \boldsymbol{x}^t} \rho\left(\left\|\pi(\mathbf{X}_j^t, \mathbf{T}_{cw}) - x_j^t\right\|\right) \tag{2.2}$$

Al tratarse de un problema de SLAM monocular deformable, si solo se incluye el término anterior el problema está subdeterminado. Es necesario incluir la energía de deformación $\varphi_e(\mathcal{L}_k^t, \mathcal{L}_k^{t-1}, \mathcal{T}_k)$, que es el prior que restringe los posibles movimientos del template, haciendo que el problema deje de estar subdeterminado y se pueda encontrar una solución. En el prior se incluyen 3 regularizadores, el de inextensibilidad, el laplaciano y el temporal.

$$\begin{aligned} \varphi_e(\mathcal{L}_k^t, \mathcal{L}_k^{t-1}, \mathcal{T}_k) = \quad & \lambda_s \varphi_s(\mathcal{L}_k^t, \mathcal{T}_k) + \lambda_b \varphi_b(\mathcal{L}_k^t, \mathcal{T}_k) \\ & + \lambda_t \varphi_t(\mathcal{L}_k^t, \mathcal{L}_k^{t-1}) \end{aligned} \tag{2.3}$$

Donde λ_s, λ_b y λ_t representan el peso que se le asocia a cada término en la optimización.

El Tracking es capaz de estimar las deformaciones de la región cubierta por el template $\mathcal{T}_k$. Si la secuencia es exploratoria, se necesita cubrir las nuevas regiones con un nuevo template. El Mapping es el proceso que crea nuevas partes del mapa y refina el template actual, y se ejecuta cuando se crea un nuevo keyframe. El criterio de

inserción de keyframes actual es el de insertar uno nuevo siempre que el Local Mapping esté disponible.

2.1.3. Local Mapping

Su función es añadir nuevos puntos al mapa de la escena y actualizar el template cada vez que un keyframe k es añadido.

Cada vez que un nuevo keyframe es añadido, DefSLAM estima una superficie $\widehat{S}_k$ por medio del algoritmo IsoNRSfM. Del IsoNRSfM se obtienen un conjunto de normales, a partir de las cuales se estima la superficie $\widehat{S}_k$, mediante el algoritmo Shape-from-Normals. Al ser un sistema monocular, $\widehat{S}_k$ está calculada hasta un factor de escala, por lo que antes de introducir $\widehat{S}_k$ como un nuevo template $\mathcal{T}_k$, se ha de estimar una solución con una escala coherente respecto del mapa.

La superficie $\widehat{S}_k$ incorporará puntos nuevos y puntos que ya se habían observado en la secuencia. Los puntos reobservados por el Local Mapping e incluidos en la superficie S_k tras corregir la escala deberán coincidir con la posición estimada por el Tracking cuando se insertó el keyframe k. Para calcular la escala se hace un alineamiento entre los puntos de S_k y los de $\mathcal{T}_{k-1}^k$ mediante un Sim(3).

Una vez se tiene S_k, se obtiene una malla triangular mediante una triangulación de Delaunay. Los puntos reobservados y los nuevos se embeben en la malla proyectándolos en las caras de esta. Así ya se tendría el template $\mathcal{T}_k^k$.

El tracking y el mapping son procesos concurrentes, siendo el Tracking capaz de funcionar a frecuencia de vídeo (10 Hz), procesando todos los frames, mientras que el Mapping únicamente procesa los keyframes (1 Hz). Por ello, cuando el Mapping haya obtenido $\mathcal{T}_k^k$, el Tracking ya habrá procesado N frames, por lo que es necesario aplicar un SfT a $\mathcal{T}_k^k$ para obtener $\mathcal{T}_k^{k+N}$. Después de ello, el Tracking puede procesar el siguiente frame empleando el template recién calculado.

2.2. Sintonía

El primer objetivo se centra en lograr que el sistema sea capaz de procesar secuencias médicas, total o parcialmente. Esto se ha conseguido mediante la sintonía de algunos parámetros, centrados en la detección de keypoints, y la decisión de inicializar con el primer par estéreo de la secuencia.

Una vez que el sistema está funcionando, ya podemos valorar sus prestaciones. Para ello se definirá una métrica y se hará una evaluación inicial del sistema, a partir de la cual se podrán tomar decisiones de diseño para mejorarlo.

2.2.1. Detección de features

Como se ha explicado anteriormente, los puntos de interés (keypoints) utilizados en todas las etapas de DefSLAM son ORB. El detector localiza puntos de la siguiente manera. Para cada pixel, analiza la diferencia de intensidad existente entre el pixel a analizar, que será el pixel central, y los pixeles que se encuentran formando un anillo respecto de él. Si existe un número de pixeles contiguos en el anillo cuya intensidad está por encima del límite inferior o superior del umbral ($I \pm umbral$), entonces ese pixel es una esquina FAST. En nuestro caso el anillo será de 12 puntos. Un ejemplo de FAST se puede ver en la figura 2.2.

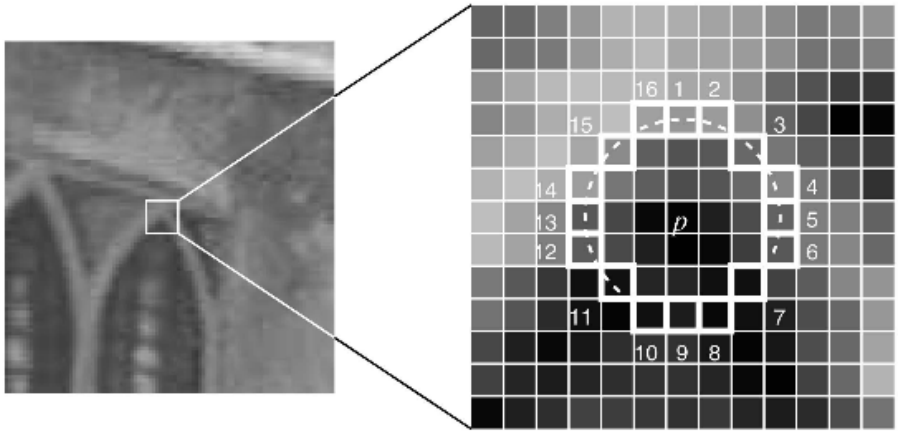


Figura 2.2: Detección de un FAST con un anillo de 12 puntos. © 2010 IEEE [8].

Cuando la textura de la escena es considerable, el detector FAST será capaz de localizar muchos keypoints. Este es el caso de escenarios industriales, por ejemplo, donde hay variedad de elementos diferenciados y fáciles de reconocer. Esto no ocurre al observar una cavidad intracorpórea en una escena médica, ya que la superficie de los órganos internos es mayormente lisa, con pocas características que destaquen sobre el resto.

Por esto mismo, el detector encontrará más dificultades en estas secuencias, que son el objetivo del trabajo. Con el umbral implementado inicialmente, el detector no era capaz de localizar suficientes puntos. Para solventar este problema, se reduce este umbral en el fichero de calibración .yaml, y se aumenta el número de keypoints deseados para obtener más puntos por imagen.

Además, para conseguir puntos multiescala, el extractor de puntos construye una pirámide de escalas con cada imagen, reduciendo su resolución de la imagen en cada piso. En el tipo de imágenes que vamos a analizar, la mayoría puntos a detectar se encontrará en la escala de mayor resolución, encontrando muy pocos en las escalas más bastas. Es por esto que solo consideraremos la escala más fina.

Descripción	Source	Mandala	Escena médica
Settings			
Niveles de escala	Settings.yaml	8	1
Umbral de oFast	Settings.yaml	15	8
Número de Features	Settings.yaml	1200	1600

Tabla 2.1: Sintonía de la detección de puntos.

2.2.2. Arranque con par estéreo

La etapa de inicialización es crítica en cualquier sistema de SLAM monocular. Esto supone la creación del primer mapa de puntos y la estimación de las primeras poses de cámara desde cero, puesto que el mapa es completamente desconocido al comienzo de la exploración. Si se dispusiera de dos cámaras, este proceso sería notablemente más sencillo, ya que con un único par de imágenes estéreo sincronizadas es posible estimar la profundidad de la escena, incluso para escenas deformables. Esto se logra encontrando correspondencias estéreo entre los puntos de ambas imágenes, para después triangular su posición 3D.

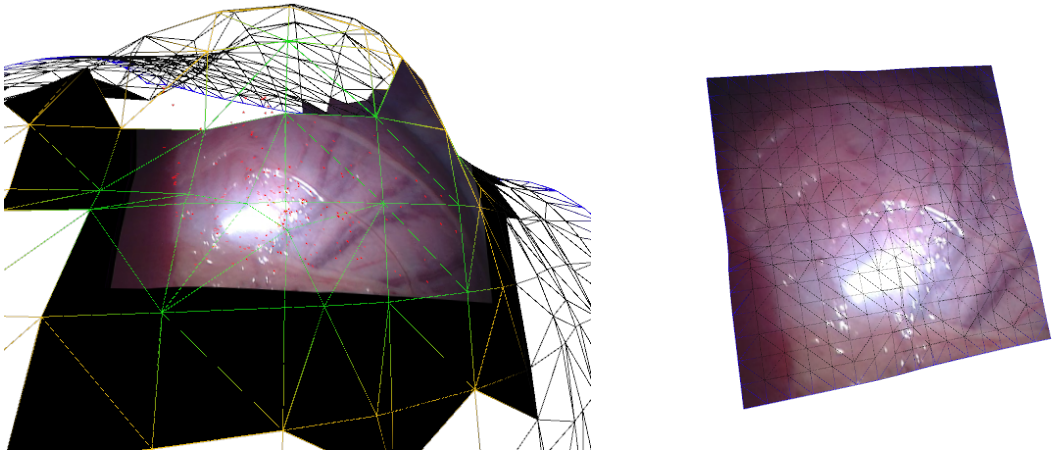
Afortunadamente, en el dataset de validación disponemos de esta información estéreo, por lo que se propone utilizarlo para arrancar el sistema. Se obtendrá una primera plantilla, mientras que para el resto de la secuencia se empleará únicamente la información monocular proporcionada por la cámara izquierda.

El estéreo calcula una nube de puntos en 3D, mientras que el template que utiliza DefSLAM es una malla triangular. Por ello se utilizará la técnica de Poisson [9] para reconstruir el primer Template, generando una malla de triángulos homogéneos a partir de la nube de puntos. La nube de puntos se obtiene a partir de los puntos detectados en las dos imágenes, de la siguiente manera:

- Se detectan los ORB en las dos imágenes del par estéreo y se hallan las correspondencias entre ambos, a través de los descriptores ORB y la geometría epipolar.
- Se triangula la posición de estos, obteniendo una nube de puntos en 3D.
- La reconstrucción de Poisson obtiene la primera plantilla.

El resultado que se obtiene es el de la figura 2.3a. Esto ocurre porque la reconstrucción tiene en cuenta todos los puntos de la escena, pero algunos de ellos están demasiado cerca o lejos de la cámara.

Para detectar y eliminar dichos puntos procesamos la nube mediante de un algoritmo de mínima mediana (LMedS) [10], que compara cada punto frente a todos los puntos del conjunto, obteniendo la distancia característica d_c de la escena (eq. 2.4).



(a) Reconstrucción inicial

(b) Reconstrucción eliminando los puntos alejados de d_c.

Figura 2.3: Reconstrucción inicial de la escena a través de una triangulación de Poisson.

Esta se corresponde con la distancia que devuelve el mínimo valor de mediana de los residuos cuadráticos $med(r_i^2)$, siendo cada residuo la diferencia entre la distancia del punto analizado y la de otro punto del conjunto.

$$\arg\min_{d_c} \quad \underset{\mathrm{i}}{\mathrm{median}} \ (\mathrm{d_i} - \mathrm{d_c})^2 \tag{2.4}$$

La aceptación o rechazo de un punto viene dada por la desviación típica s para la distancia característica calculada. Un punto será rechazado si su distancia d_i se aleja un umbral de la distancia característica, según 2.6.

$$s = 1{,}4826(1 + \frac{5}{n-k})\sqrt{med(r_i^2)} \tag{2.5}$$

$$\left| \frac{d_i - d_c}{s} \right| > 2{,}0 \tag{2.6}$$

Donde n es el número total de puntos y k es el número de puntos usados para crear el modelo, en este caso, uno.

El template que se obtiene tras eliminar los puntos alejados de d_c se observa en la figura 2.3b. Una vez calculado, DefSLAM puede estimar las deformaciones y la pose de cámara mediante SfT, mientras se procesan KeyFrames que, eventualmente, darán lugar a un segundo Template. Este segundo Template y los que le sigan, son calculados únicamente mediante DefSLAM monocular, sin ningún apoyo del estéreo.

Capítulo 3

Métrica del error en el mapa local

El segundo objetivo del proyecto consiste en hacer una evaluación cuantificable de las estimaciones geométricas que realiza el sistema. Cuando se está evaluando un sistema, es habitual disponer de una solución de referencia calculada con otro sistema que se considere altamente fiable. Esta solución se denomina Ground Truth, y mediante la comparación con la solución obtenida por el sistema en estudio, permite evaluarlo.

Para el caso del SLAM rígido existen datasets como el KITTI [11] o el EUROC [12], que incluyen una solución Ground Truth que consiste principalmente en la trayectoria real de la cámara para cada frame de la secuencia. De esta manera un sistema SLAM puede comparar la trayectoria de cámara calculada con la solución de referencia disponible. Esta es la forma habitual de evaluar estos sistemas, sin prestar mucha atención al mapa estimado.

En el caso médico, la trayectoria absoluta de la cámara es difícil de estimar, siendo además una información no muy relevante. La información realmente relevante es la posición relativa de la cámara respecto de la escena, que también representa la calidad del mapa visto desde la cámara.

Los actuales sensores de imagen médica no pueden proporcionar información sobre la geometría de las superficies de los órganos a frecuencia de vídeo. Desde este punto de vista, los sistemas de SLAM serían muy interesantes porque podrían proporcionar esa información por primera vez. La dificultad es que no tenemos un sensor alternativo que nos permita estimar la calidad de nuestras estimaciones. Por ello, se ha tenido que diseñar un algoritmo que permitiese evaluar los errores geométricos cometidos al estimar la superficie a través de DefSLAM monocular.

Algunas regiones del cuerpo permiten introducir instrumentos estéreo, lo cual da información suficiente para calcular una estimación de Ground Truth, ya que se puede obtener la profundidad de la escena mediante la triangulación entre las correspondencias de cada par estéreo.

La solución estimada por DefSLAM consiste en una superficie en la cual se embeben

las observaciones, que son los puntos ORB detectados, y que serán los puntos de medida para compararlos con la solución de Ground Truth. Este Ground Truth está formado por los mismos puntos de medida, pero calculados a través del estéreo. De manera que para cada punto 3D estimado por DefSLAM, se dispondrá de una estimación realizada mediante estéreo, con la que se comparará. Los puntos de medida utilizados son todos aquellos incluidos en el mapa local.

La evaluación del sistema se realiza de la siguiente forma:

- Cada imagen izquierda de la secuencia se procesa con DefSLAM.
- Tras ello, el mapa local se proyecta en la imagen izquierda y se empareja por correlación con la imagen derecha.
- Se calcula la posición 3D de cada punto emparejado en el par estéreo.
- Los dos mapas son alineados para evitar la influencia de la escala y se calcula el error RMSE.

A lo largo del capítulo se explica en detalle cada etapa de este proceso.

3.1. Estimación del Ground Truth a partir de estéreo

Para estimar la posición GT de un punto del mapa estimado, primero se proyecta este punto sobre la imagen de la cámara izquierda. A partir de la proyección se define el patch que permitirá encontrar la imagen del mismo punto sobre la imagen derecha. La búsqueda de la correspondencia sobre la imagen derecha se hace por correlación. restringiendo la búsqueda a una región de la imagen derecha definida por la geometría epipolar del par estéreo.

En la figura 3.1 se ejemplifica el proceso de obtención del GT. Los dos rectángulos azules de las imágenes representan dos patches emparejados, que a su vez se corresponden con los dos puntos verdes. El punto verde que está en el mapa local $\mathcal{L}_k^t$ es el obtenido mediante DefSLAM, mientras el otro que está en $\mathbf{GT}^t$, se ha obtenido a través del estéreo.

El primer paso para obtener la estimación de Ground Truth es proyectar el mapa local en la imagen, como se ha comentado. La proyección se realiza mediante una transformación perspectiva, a través de la función proyectiva π (eq. 3.1), que nos devuelve para el punto 3D j, $\mathbf{X}_{w,j}^t$, su posición en píxeles de la imagen para una cámara localizada en la pose $\mathbf{T}_{cw}^t$.

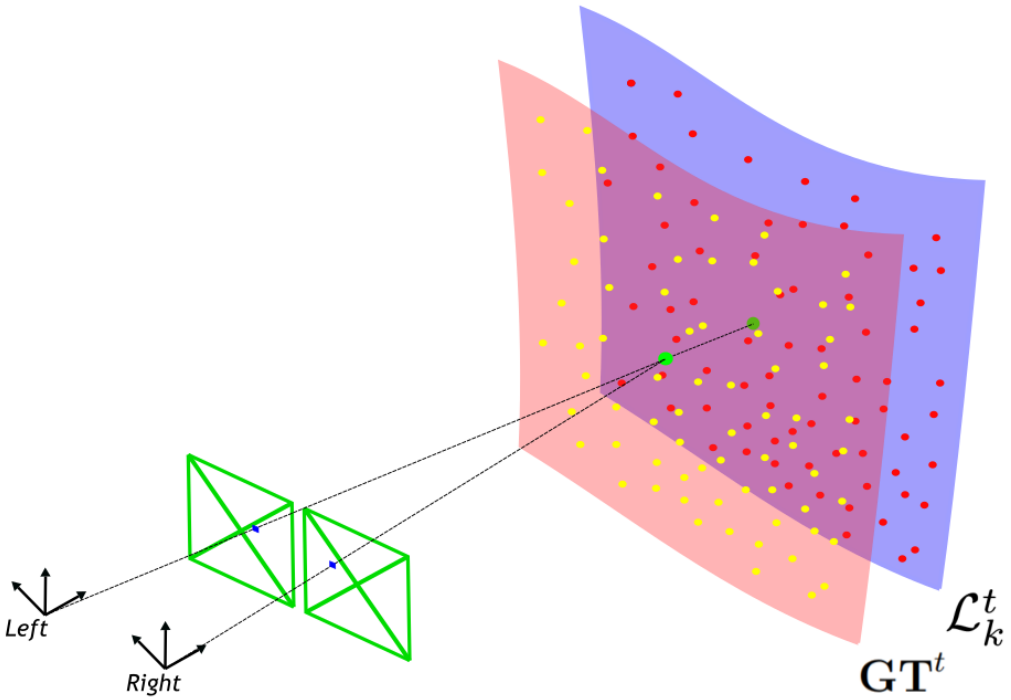


Figura 3.1: Obtención del GT a partir de estéreo. La plantilla azul con puntos rojos representa la zona local del template L_k^t, mientras que la plantilla salmón con puntos amarillos es la solución

$$\pi(\mathbf{T}_{cw}^t, \mathbf{X}_{w,j}^t) = \begin{bmatrix} f_x \frac{X_j^t}{Z_j^t} + C_x \\ f_y \frac{Y_j^t}{Z_j^t} + C_y \end{bmatrix} \tag{3.1}$$

$$\begin{bmatrix} X_j^t & Y_j^t & Z_j^t \end{bmatrix}^T = \mathbf{R}_{cw}^t \mathbf{X}_{w,j}^t + \mathbf{t}_{cw}^t \tag{3.2}$$

Siendo $\mathbf{R}_{cw}^t$ y $\mathbf{t}_{cw}^t$ la rotación y traslación de la pose $\mathbf{T}_{cw}^t$. (f_x, f_y) y (C_x, C_y) son las distancias focales y el punto principal de la cámara, en píxeles.

De esta manera tendremos todos los puntos locales en coordenadas de imagen. Para cada uno de estos puntos nos guardaremos un patch (figura 3.2). Este patch es una ventana centrada en el punto y que nos permitirá buscar cada punto en la imagen derecha para emparejarlo.

Tras tener el mapa reproyectado en la imagen izquierda, se emparejará con la imagen derecha mediante correlación. Este es un método más caro computacionalmente que el emparejamiento con ORB u otras features, pero es más estable y preciso. Consiste en buscar un área en la imagen que coincida con el patch que se ha guardado previamente para cada punto.

La región de búsqueda está definida en la imagen derecha por la geometría epipolar del par estéreo. Al estar rectificadas las imágenes, las líneas epipolares son horizontales. Buscaremos los emparejamientos en una ventana rectangular centrada en la línea

epipolar correspondiente a cada punto, como se ve el emparejamiento de la figura 3.2.

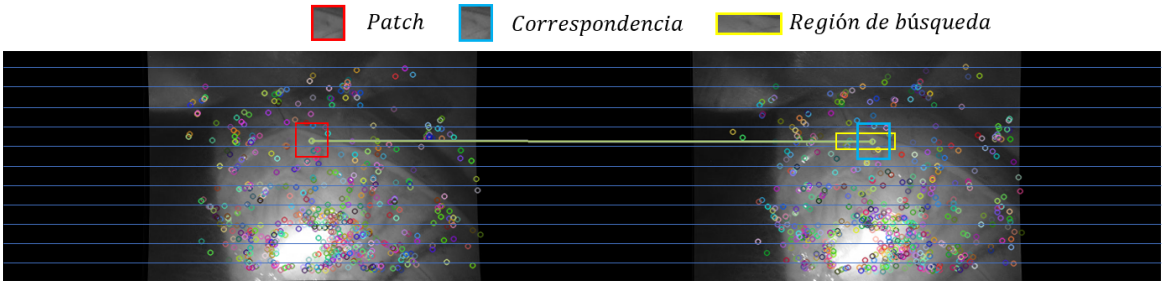


Figura 3.2: Ejemplo de un match entre pares estéreo a través del patch del punto reproyectado. En la imagen izquierda se muestra la proyección de todos los puntos del mapa de los que se quiere estimar el GT. En la derecha se muestra el resultado de la búsqueda de emparejamientos por correlación. Las líneas azules correponden con la geometría epipolar.

Una vez se tienen los emparejamientos, ya se puede calcular la posición en 3D de cada punto de la imagen. Primeramente, se estima la diferencia en pixeles en la horizontal de la imagen entre un punto emparejado en la cámara izquierda y su correspondiente en la derecha. Esta distancia se denomina disparidad d_j, y a partir de ella se puede estimar la profundidad Z_j^t de cada punto.

$$Z_j^t = B\frac{f}{d_j} \tag{3.3}$$

$$d_j = x_j^l - x_j^r \tag{3.4}$$

Donde x_j^l y x_j^r son la posición horizontal del punto emparejado en la imagen izquierda y derecha, respectivamente. Las coordenadas X_j^t, Y_j^t se pueden obtener reproyectando según el modelo de cámara pinhole (eq. 3.5).

$$\begin{bmatrix} X_j^t \\ Y_j^t \end{bmatrix} = \begin{bmatrix} Z_j^t \frac{u-C_x}{f_x} \\ Z_j^t \frac{v-C_y}{f_y} \end{bmatrix} \tag{3.5}$$

Donde u, v son las coordenadas de cada punto en la imagen izquierda, en píxeles. De esta manera se obtiene la posición 3D de cada punto del mapa local en coordenadas de cámara, $\mathbf{X}_{c,j}$, lo cual denominaremos Ground Truth. Este Ground Truth se comparará con la posición original de los puntos del mapa local para determinar el error tras realizar un alineamiento de escala, el cual se explica en la siguiente sección.

3.2. Alineamiento de escala y cálculo del error

Antes de comparar la solución Ground Truth con la de DefSLAM, hemos de tener en cuenta el hecho de que un sistema monocular solo es capaz de reconstruir la escena hasta

un factor de escala. Al disponer de una única cámara y no conocer ninguna distancia real en la escena, la escala puede ir derivando conforme avanza la secuencia. En estéreo se conoce la baseline entre el par de cámaras y por ello la escena se reconstruye en su verdadera escala.

Es por esto que el mapa local monocular debe alinearse con el Ground Truth antes de estimar los errores geométricos, ya que sino se estaría incluyendo el error de deriva. Este alineamiento se realiza en cada frame a través de un algoritmo LMedS, que nos devolverá la escala que minimice la expresión 3.6.

$$\arg\min_{\gamma} \quad \underset{\mathrm{j}}{\mathrm{median}} \; ||\gamma \mathbf{X}_{\mathrm{D,j}} - \mathbf{X}_{\mathrm{GT,j}}||^2 \tag{3.6}$$

Donde $\mathbf{X}_{D,j}$ y $\mathbf{X}_{GT,j}$ son las posiciones 3D en coordenadas de la cámara izquierda del punto j-ésimo del mapa local, en la solución de DefSLAM y en el Ground Truth, respectivamente. γ es la escala que se está calculando. La γ obtenida de este algoritmo se debería utilizar en un ajuste no lineal con los puntos aceptados, pero al ser un ajuste de un único parámetro los resultados son muy similares y no se ha realizado este ajuste por simplicidad. Cabe destacar que este alineamiento solo se aplica para la evaluación, el mapa no es cambiado para DefSLAM.

Una vez se tienen los mapas alineados, se pueden comparar entre sí y obtener las medidas de error correspondientes. La métrica escogida para este fin será el RMSE (Root Mean Squared Error), cuya expresión es:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} {e_j}^2}{N}} \tag{3.7}$$

$$e_i = ||\gamma \mathbf{X}_{D,j} - \mathbf{X}_{GT,j}|| \tag{3.8}$$

Donde e_j es el error para cada punto del mapa local, que será calculado como la distancia euclídea entre ambos (eq. 3.8), y N es el número de puntos incluidos en el mapa local.

El error de estimación del mapa se visualiza en la figura 3.3.

3.3. Evaluación inicial

Con la sintonía inicial se consigue que el sistema sea capaz de inicializar en una secuencia médica y sobrevivir en ella. Con ello se puede hacer una primera valoración su funcionamiento. En los experimentos se muestra por un lado, el error RMSE, y por otro, la evolución de la escala. La evolución de la escala nos indica si la escala está

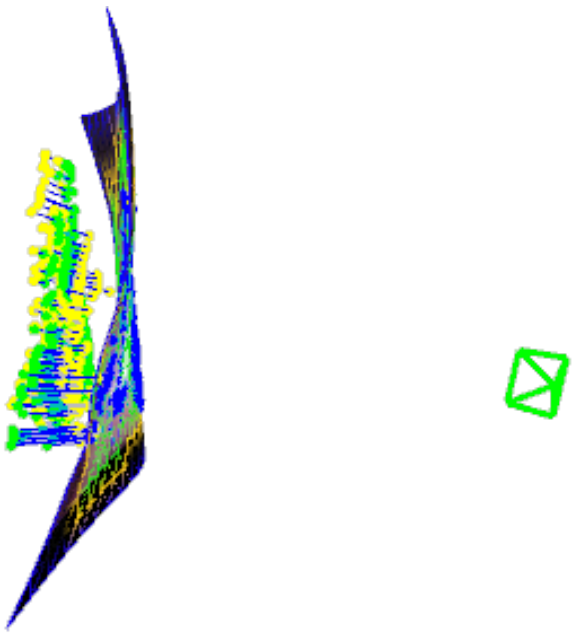


Figura 3.3: Error de estimación del mapa. En azul se muestran los puntos de medida del mapa, que están embebidos en el template deformado. La estimación del GT para los puntos de medida se muestra en amarillo. Los puntos verdes corresponden con los puntos de medida tras aplicar el alineamiento de escala. Las líneas azules muestran el error de cada punto.

derivando, y por tanto si el tamaño del mapa está cambiando, alejándose del tamaño real de la escena.

Para ambas secuencias se muestran 2 ejecuciones distintas del programa, para visualizar la variabilidad de los resultados obtenidos cuando se procesa la misma secuencia. Al tratarse de un sistema concurrente, la interacción entre procesos y la aleatoriedad de algunos algoritmos internos hacen que se obtengan resultados distintos en cada ejecución. Por ello veremos que en algunos casos el sistema se pierde durante la secuencia, mientras que en otros es capaz de sobrevivir.

La secuencia 1 corresponde con la exploración de la cavidad abdominal in-vivo de un cerdo, sobre la que se realizan punzamientos con una herramienta que ocluye la escena (fig. 3.4). La dificultad de esta secuencia radica en su componente exploratorio, ya que se observan nuevas regiones, además de que hay herramientas que interactúan con la escena.

Los resultados de la secuencia 1 corresponden a la figura 3.5, a partir de los cuales se concluye que el sistema funciona cuando la escena está despejada, pero se pierde completamente en algunas ocasiones en las que la herramienta interviene. El error oscila entre los 2-5 mm cuando el sistema funciona correctamente, apareciendo algún pico de error en algún frame. Cuando la herramienta ocluye parte de la imagen (frame 520) y cuando se trabaja con ella (frames 650-750 y 900-1200), el error se dispara

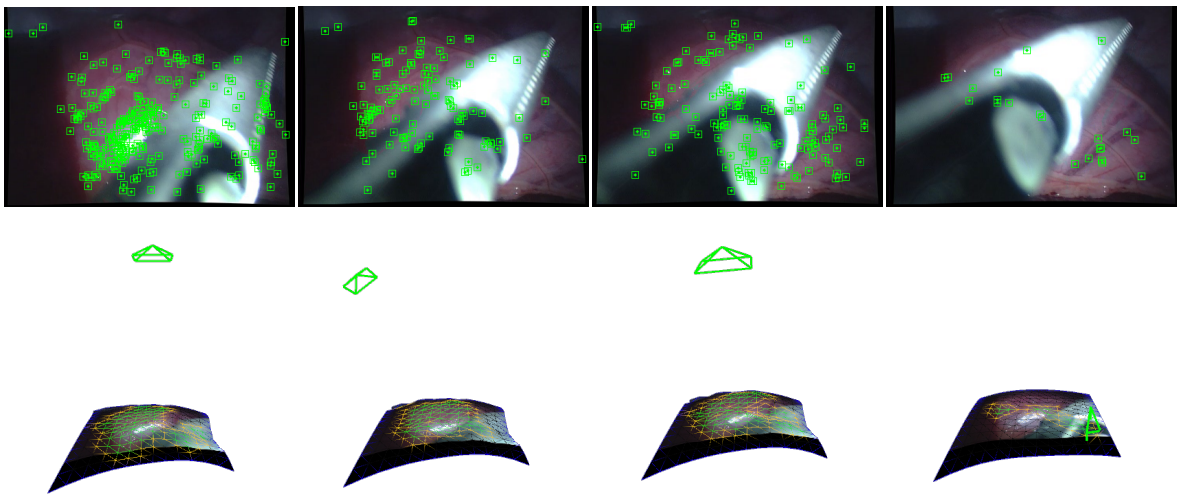
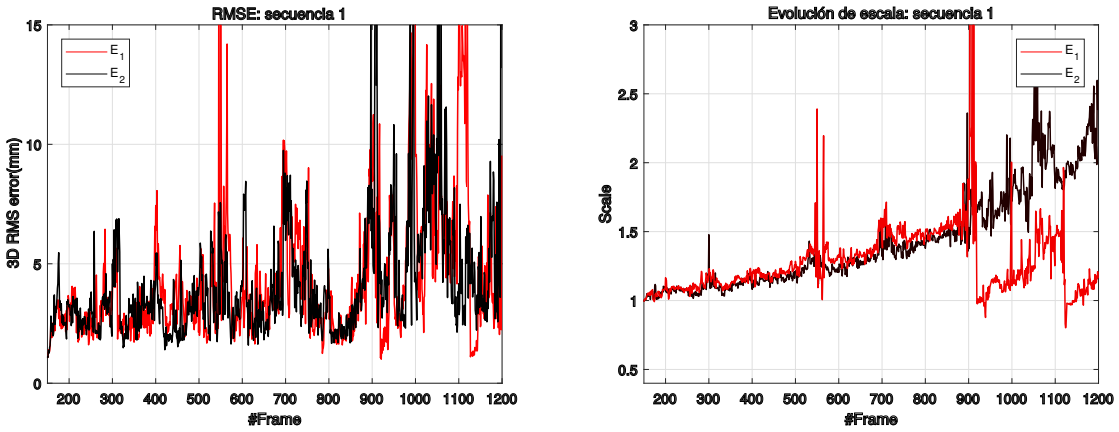


Figura 3.4: Herramienta ocluyendo la imagen en la secuencia 1, dando lugar a estimaciones erróneas de la pose de la cámara.

para finalmente perderse por completo e inicializar de nuevo en el caso de la ejecución E_1. La escala de E_1 se descontrola hasta que vuelve a la unidad cuando el sistema inicializa de nuevo. E_2 no tiene que volver a inicializar, pero igualmente la cámara se pierde y la estimación que realiza es errónea. En cuanto a la escala, vemos que aumenta progresivamente alcanzando una deriva del 150 % al final de la secuencia.



(a) RMSE

(b) Evolución de escala

Figura 3.5: Resultados para la secuencia 1 con los ajustes iniciales.

La secuencia 2 muestra la exploración del corazón in-vivo de un cerdo, mientras la cámara permanece estática observando la escena (fig. 3.6). En este caso, la dificultad de la escena radica en la constante deformación de esta. El sistema considera que la cámara está en movimiento y traslada parte de la deformación observada al movimiento de la cámara.

Los datos obtenidos son los de la figura 3.7. El error oscila entre los 1-4 mm, observándose algún pico superior cuando el sistema no se localiza bien. En concreto

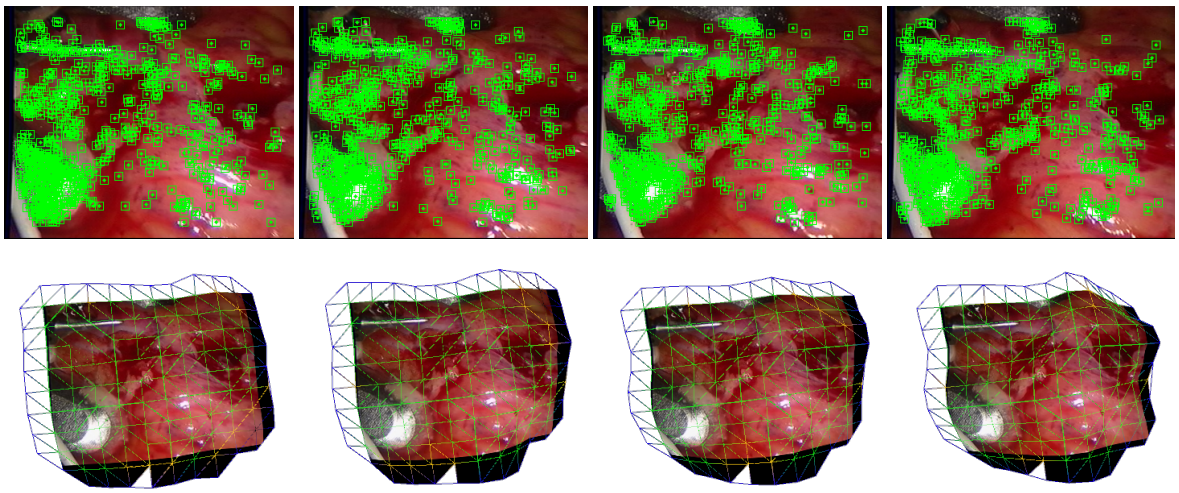
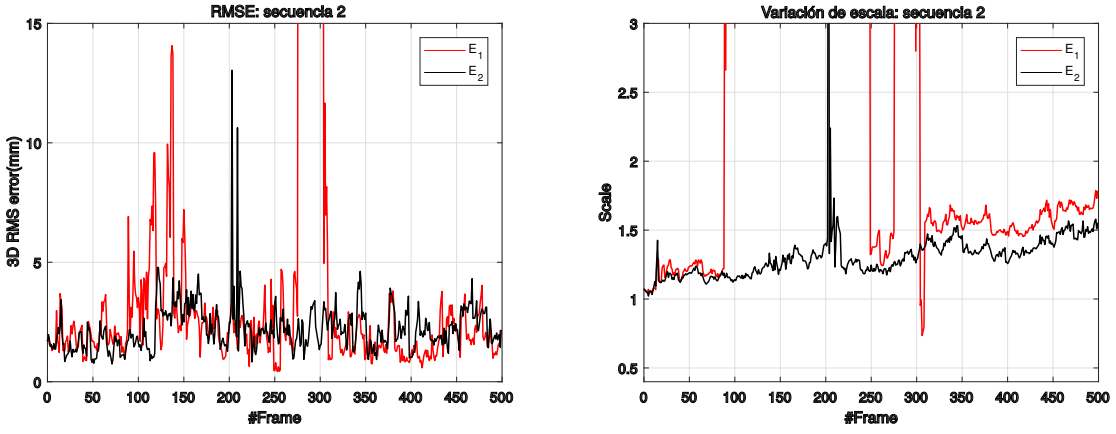


Figura 3.6: Corazón in-vivo de un cerdo de la secuencia 2.

para la ejecución E_1, se observa una etapa en la que el sistema se ha descontrolado entre los frames 90-250 y 270-300, en los que la escala se ha disparado, estimando un mapa muy distinto del real. En el frame 300, E_1 se pierde y tiene que volver a inicializar. En E_2 el sistema se pierde en el frame 200, pero logra localizarse de nuevo. Presenta una deriva de escala importante, al igual que E_1, superando ambas el 50 % de deriva.



(a) RMSE

(b) Variación de escala

Figura 3.7: Resultados para la secuencia 2 con los ajustes iniciales.

Se detectan dos fallos de evaluación que serán corregidos durante el proyecto. Por un lado, se tiene que el sistema tiene deriva de escala, lo que significa que la escala aumenta o disminuye partiendo de la escala inicial que es igual a la unidad. Esto provoca que el tamaño del mapa estimado varíe a lo largo de la secuencia.

Por otra parte, el detector de puntos dispara erróneamente en los reflejos ocasionados por las fuentes de luz de los aparatos de inspección visual. El detector también dispara en las herramientas que se utilizan en las intervenciones quirúrgicas, haciendo

que el sistema trate de emparejar puntos que no pertenecen a la superficie. Cuando la herramienta ocluye la imagen actual, el sistema puede perderse.

Estos problemas serán resueltos en los capítulos 4 y 5, respectivamente.

Capítulo 4

Política de inserción de KeyFrames

Cuando se decide insertar un nuevo keyframe en el mapa, DefSLAM comienza el proceso de cálculo de un nuevo template en el hilo del Local Mapping. El tiempo de cómputo de este nuevo template es muy superior al tiempo de procesamiento que ocupa un frame en el tracking, en el que se estima la pose de cámara y la deformación del mapa.

El nuevo template tiene un factor de escala arbitrario, por lo que deberá alinearse con el anterior antes de sustituirse. La alineación se realiza mediante un Sim(3), haciendo coincidir los puntos comunes entre el template a insertar y el actual. Este alineamiento no es perfecto en la práctica, ya que siempre existirá algún desajuste entre el mapa nuevo y el actual.

Además, la falta de sincronía entre el Local Mapping y el Tracking provoca que para cuando el Local Mapping haya terminado de procesar un keyframe, el Tracking habrá procesado ya un cierto número de frames. Por ello habrá que deformar el nuevo template mediante un SfT antes de introducirlo y procesar el siguiente frame.

El error que se produce en los alineamientos es acumulativo y da lugar a una deriva de escala que aumenta cada vez que se crea un nuevo template, y por consiguiente, cada vez que se inserta un keyframe.

La inserción de keyframes es necesaria para la supervivencia del sistema, pero es un proceso que se puede regular. Lo que se propone es reducir el número de keyframes insertados para reducir esta deriva de escala, restringiendo la inserción de keyframes a las situaciones en las que sea realmente necesario.

La decisión de insertar un nuevo keyframe en la implementación para la Mandala depende de la siguiente fórmula booleana:

$$\text{NeedNewKeyframe} = c_1 \wedge c_2 \tag{4.1}$$

Donde el significado de cada elemento es:

– c_1: el Local Mapping está libre, no está procesando ningún keyframe.

– c_2: hay más de 30 emparejamientos en el frame actual.

De manera que se inserta un nuevo keyframe cuando los emparejamientos en el frame actual son mayores de 30 y el mapa local está disponible. Es decir, prácticamente se inserta un keyframe siempre que se pueda.

Esto no es necesariamente positivo, ya que en nuestro caso el sistema incluye más keyframes de los que necesita, modificando el template en ocasiones en los que la estimación es buena. Por ello se ha propuesto una nueva condición para insertar keyframes, restringiéndolo.

$$\text{NeedNewKeyframe} = (c_1) \wedge (c_2 \vee c_3) \tag{4.2}$$

En este caso, cada término significa::

– c_1: el Local Mapping está libre.

– c_2: hay más de 30 emparejamientos en el frame actual y el porcentaje de nodos vistos del template es inferior al 50 %.

– c_3: el error de reproyección medio es superior a los 10 pixeles.

Como se ha expuesto en la figura 2.1, el mapa local $\mathcal{L}_k^t$ es la parte visualizada del template, que representa una porción del template $\mathcal{T}_k^t$. Si $\mathcal{L}_k^t$ es una fracción muy pequeña del template, significa que o bien la cámara se ha acercado mucho al template, o que la escena que se está visitando es nueva y el template no la cubre. En ambos casos conviene insertar un nuevo keyframe para actualizar el template.

Por otro lado, el error de reproyección es el error que existe entre los puntos reproyectados del template y su emparejamiento con los ORB de la imagen. Si este error aumenta, significa que el template del que se dispone no está haciendo buenas estimaciones de las deformaciones que se producen, y por tanto, conviene recalcularlo mediante la inserción de un nuevo keyframe.

Lo que se propone es introducir keyframes de manera controlada, favoreciendo la inserción en el caso de que se estén explorando nuevas zonas o cuando el template del que se dispone no es capaz de hacer buenas estimaciones. El hecho de insertar menos keyframes tiene consecuencias directas sobre la deriva de escala al acumular menos error en los alineamientos.

Capítulo 5

Gestión de oclusiones y reflejos

Los instrumentos de inspección visual utilizados en la toma de imágenes médicas disponen de fuentes de luz que iluminan la zona que se está grabando. Esto produce la aparición de reflejos en la imagen, lo que puede ocasionar que el detector de puntos dispare en dichos reflejos. Al moverse la fuente de luz, estos reflejos cambian de lugar, por lo que se producen falsos emparejamientos en el procesamiento de la secuencia.

Además, si se ha de interactuar con la zona observada para realizar alguna intervención quirúrgica, necesariamente aparecerá una herramienta en la imagen, ocluyendo parte de esta. El sistema tratará de procesarla como si formase parte de la superficie observada, dando lugar a falsos emparejamientos y errores en la estimación, llegando a perderse completamente en ocasiones.

La solución propuesta para abordar este problema es detectar las regiones en las que aparezcan reflejos o herramientas, y eliminar los keypoints detectados en estas zonas, a través de dos máscaras que almacenen esta información. Se tendrá una máscara para reflejos y otra para herramientas. Esta última tiene un alto coste de procesamiento y por ello será preprocesada.

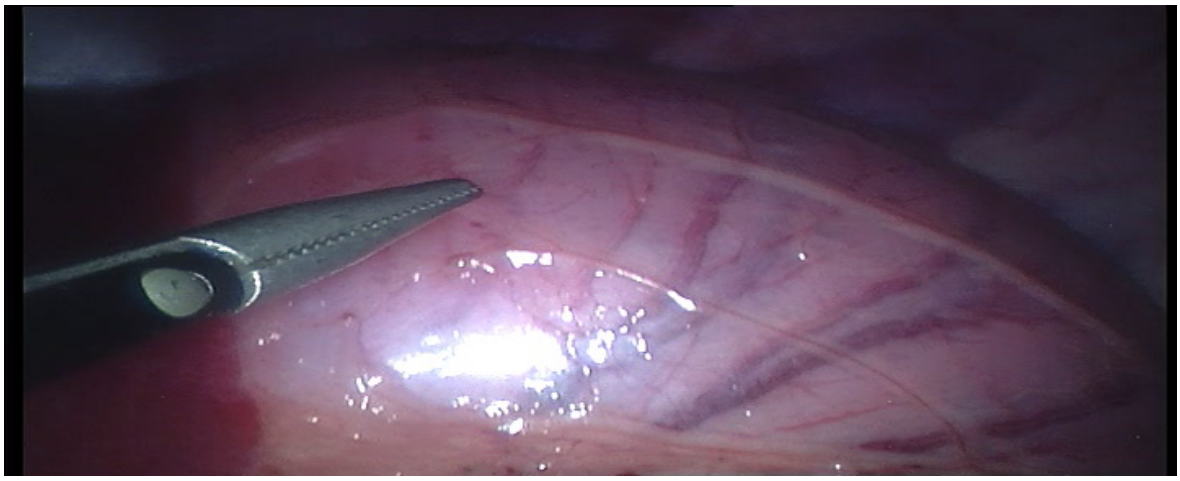
Si una herramienta ocluye la mayor parte de la imagen, el sistema detectará puntos que no debería procesar, estimando erróneamente la pose de la cámara (fig. 3.4). Si por el contrario, se elimina la zona ocluida mediante las máscaras, el problema de calcular la pose de cámara queda mal condicionado, al no tener puntos suficientes para hacer una buena estimación. Por ello también se ha restringido el movimiento de la cámara mediante un prior de movimiento suave, de forma que si el sistema no dispone de los puntos necesarios para hacer una buena estimación de la pose de cámara durante la oclusión, esta no cambie radicalmente.

5.1. Generación de máscaras

Se han generado dos máscaras, una para detectar los reflejos y la otra para detectar las herramientas quirúrgicas. Se muestran los efectos sobre la secuencia 1.

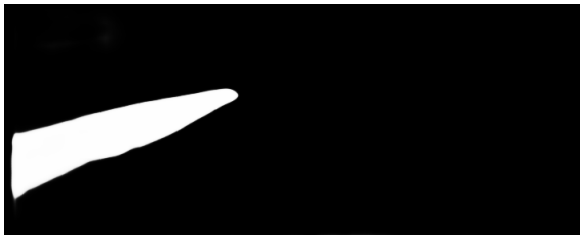
Para crear la máscara que detecte los reflejos, se asume que estos producen píxeles con altos niveles de gris. Se ha aplicado un umbral de 200 sobre 255, siendo incluidos en la máscara todos aquellos píxeles que estén por encima de este umbral.

La máscara que detecta las herramientas se ha generado siguiendo lo propuesto en [13]. Se trata de una red neural convolucional entrenada para detectar herramientas quirúrgicas en escenas médicas. La red tiene gran precisión para detectar la herramienta, solo devolviendo algún falso positivo en la zona del fondo de la escena, mientras que el recall es bajo, no siendo capaz de detectar la herramienta cuando está lejos de la cámara, habiendo falsos negativos. Los falsos positivos no nos afectan mucho ya que se dan en zonas con baja textura donde se detectan pocos puntos.



(a) Imagen de la secuencia 1.

(b) Reflejos.



(c) Herramientas.

Figura 5.1: Ejemplo de obtención de máscaras para una imagen de la secuencia 1. Las zonas en blanco representan las regiones detectadas como reflejos o herramientas.

Las máscaras generadas para la imagen de la figura 5.1a son la figura 5.1b para los reflejos, y 5.1c para las herramientas. Ambas máscaras se combinan y se engrosa mediante una dilatación la máscara final, para evitar que se detecten puntos en los bordes de reflejos y herramientas.

5.2. Movimiento suave de cámara

La optimización realizada por el algoritmo SfT toma la función de coste de la ecuación 5.1, reduciendo el error de reproyección y la energía de deformación del template $\mathcal{T}_k^t$. De esta manera se estima la deformación que ha sufrido el mapa local $\mathcal{L}_k^t$ en el frame t, así como la pose de cámara $\mathbf{T}_{cw}^t$.

$$\underset{\mathcal{L}_k^t, \mathbf{T}_{cw}^t}{\arg\min} \quad \varphi_d(\mathcal{I}^t, \mathbf{T}_{cw}^t, \mathcal{L}_k^t) + \varphi_e(\mathcal{L}_k^t, \mathcal{L}_k^{t-1}, \mathcal{T}_k) \tag{5.1}$$

Para suavizar los movimientos de la cámara, se añade un nuevo término a la función de coste. Este nuevo término corresponde la transformación $\mathbf{T}_{t,t-1}$ (ec. 5.2) entre la pose de la cámara en el frame anterior, $\mathbf{T}_{cw}^{t-1}$, y la pose de la cámara en el frame actual $\mathbf{T}_{cw}^t$. La optimización tratará de minimizar esta transformación, de manera que se limite la rotación y la traslación que puede sufrir la cámara entre el instante previo y el actual.

$$\mathbf{T}_{t,t-1} = \mathbf{T}_{cw}^t (\mathbf{T}_{cw}^{t-1})^{-1} \tag{5.2}$$

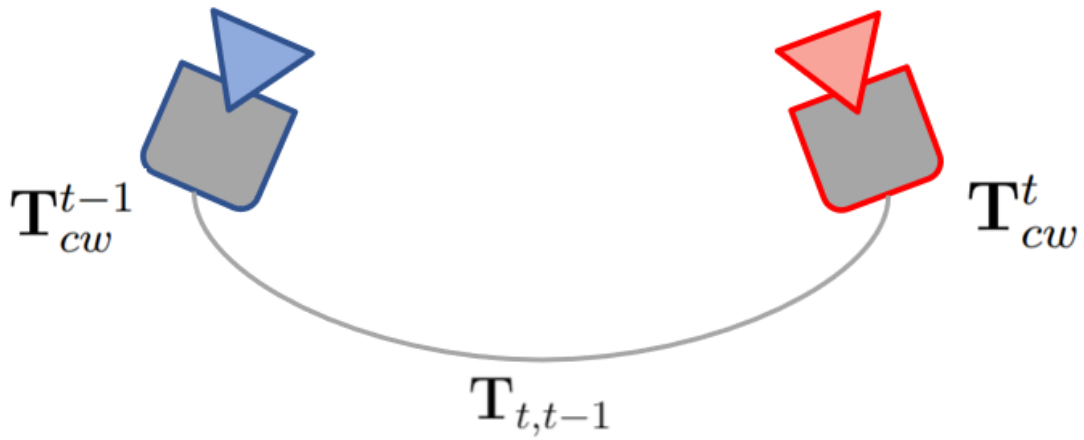


Figura 5.2: Transformación entre dos poses de cámara consecutivas.

La transformación $\mathbf{T}_{t,t-1} \in SE(3)$ está formada por una rotación $\mathbf{R}_{cw}^t \in SO(3)$ y una traslación $\mathbf{t}_{cw}^t \in \mathbb{R}^3$. Se puede representar como una traslación en el plano 3D más una rotación con los 3 ángulos de Euler: yaw(ϕ), pitch(χ) y roll(ψ).

$$\mathbf{T}_{t,t-1} \equiv \begin{bmatrix} x & y & z & \phi & \chi & \psi \end{bmatrix}^T \tag{5.3}$$

La ecuación de coste al añadir este término queda como:

$$\underset{\mathcal{L}_k^t, \mathbf{T}_{cw}^t}{\arg\min} \quad \varphi_d(\mathcal{I}^t, \mathbf{T}_{cw}^t, \mathcal{L}_k^t) + \varphi_e(\mathcal{L}_k^t, \mathcal{L}_k^{t-1}, \mathcal{T}_k) + \varphi_c(\mathbf{T}_{t,t-1}) \tag{5.4}$$

Donde $\varphi_c(\mathbf{T}_{t,t-1})$ correspondiente con el prior de movimiento suave de cámara. Este prior (eq. 5.5) se compone de dos términos, por un lado $\lambda_{tr}(x, y, z)$ suaviza las traslaciones y por otro, $\lambda_{rt}(\phi, \chi, \psi)$ suaviza las rotaciones. λ_{tr} y λ_{rt} son los pesos respectivos que se les da en la optimización.

$$\varphi_c(\mathbf{T}_{t,t-1}) = \lambda_{tr}(x, y, z) + \lambda_{rt}(\phi, \chi, \psi) \tag{5.5}$$

Capítulo 6

Validación experimental

En este capítulo se recopilan los resultados obtenidos con el sistema para las dos secuencias seleccionadas, evaluados según la métrica propuesta. Los resultados se analizarán en función del error RMSE y de la deriva de escala sufrida a lo largo de la secuencia, al igual que en la evaluación inicial de la sección 3.3.

Los pesos que se han asignado en la función de coste 5.4 para ambas secuencias son:

Término	Peso
Inextensibilidad (λ_s)	1.6x10^4
Laplaciano (λ_b)	65
Temporal (λ_t)	0.03
Traslación de cámara (λ_{tr})	0.75
Rotación de cámara (λ_{rt})	4

Tabla 6.1: Asignación de pesos para la optimización.

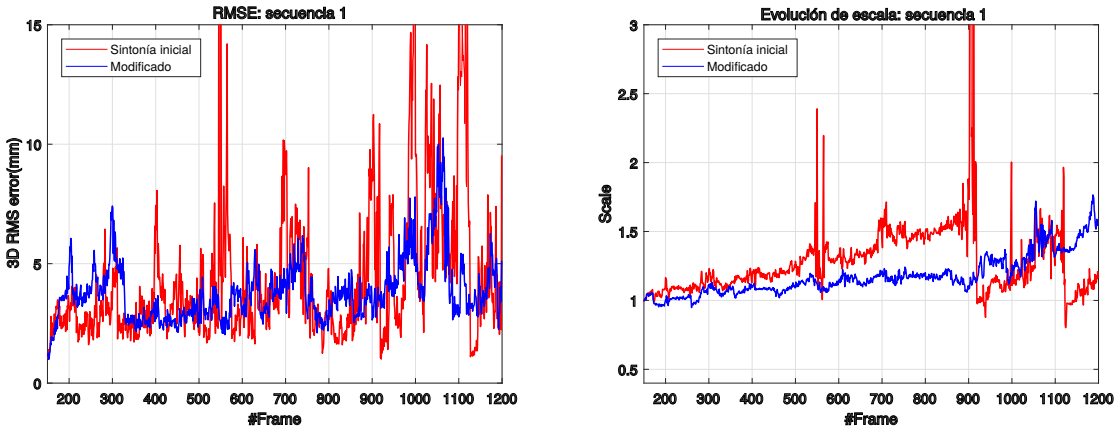
El computador utilizado para estos experimentos es un Intel® Core™ i7-4790 CPU @ 3.60GHz 8 con 15'6 GB de memoria RAM.

6.1. Secuencia 1

Se presentan los resultados para la secuencia 1 una vez implementada la modificación en el criterio de inserción de keyframes, el prior de movimiento suave de cámara y las máscaras, comparándose con la evaluación inicial.

Las contribuciones añadidas consiguen que el sistema sea capaz de procesar la secuencia completa sin perderse. En este caso la oclusión no produce un pico de error, ya que se eliminan los puntos detectados en ella. El error se ve reducido, situándose entre los 2-4 mm durante la mayor parte de la secuencia, a excepción del tramo final. En este tramo la herramienta hace punzamientos en el órgano pero la máscara no lo detecta, por lo que el programa trata de procesar la herramienta como si formase parte de la superficie cubierta por el template.

Con la escala ocurre algo similar. Esta se mantiene cercana a la unidad durante gran parte de la secuencia, pero cuando llega al tramo final va derivando hasta el 50 %. En cualquier caso, esta deriva es muy inferior a la de la evaluación inicial, ya que antes de perderse en el frame 900, presentaba una deriva del 50 %, frente al 20 % del sistema modificado en ese instante.



(a) RMSE (b) Evolución de escala

Figura 6.1: Resultados para la secuencia 1 con todas las contribuciones.

El sistema es capaz de estimar las deformaciones que se producen en el órgano observado cuando la herramienta presiona sobre él, como se ve en la figura 6.2.

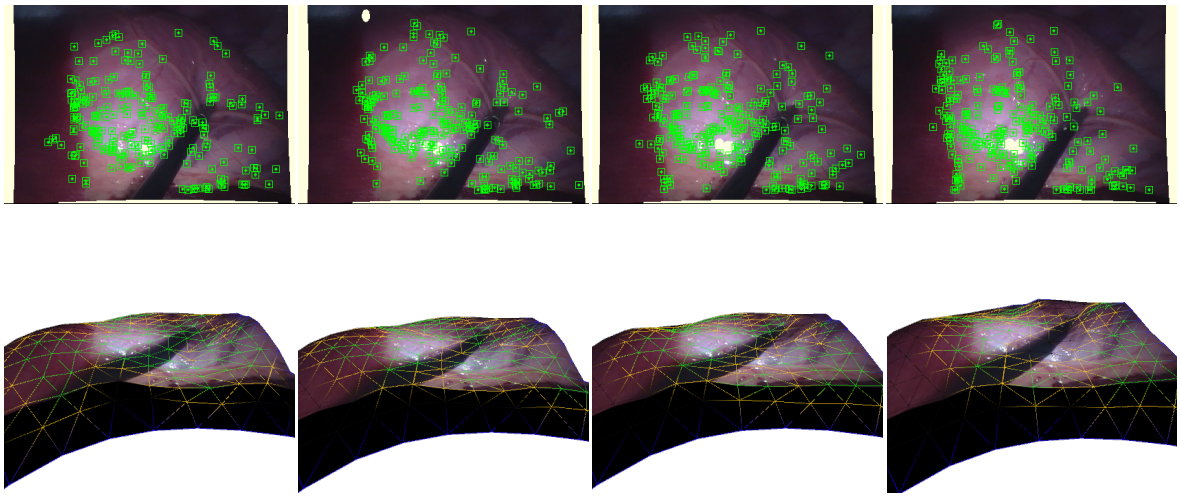


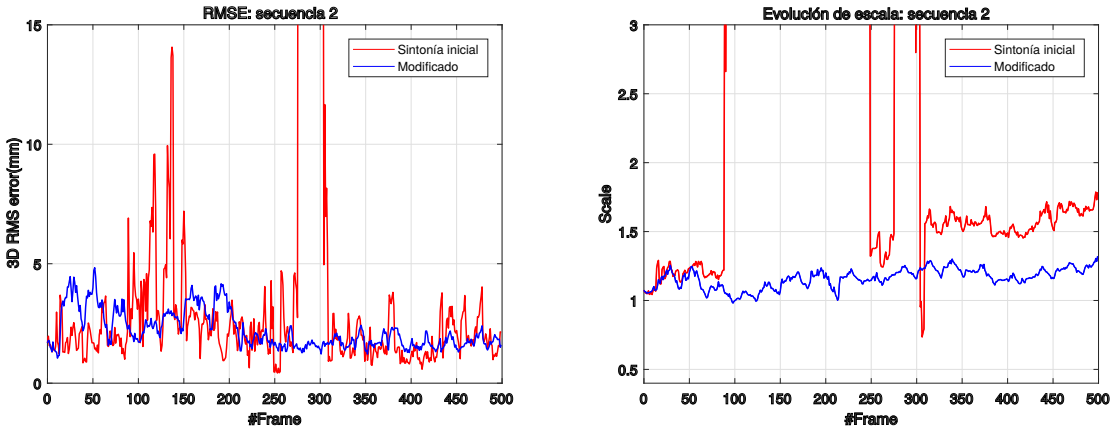
Figura 6.2: Estimación de la deformación producida por la herramienta en la secuencia 1.

6.2. Secuencia 2

Para la secuencia 2 solo se aplica la máscara de reflejos, ya que solo aparecen herramientas estáticas que no causan problemas. También se incluyen la modificación

en la política de inserción de keyframes y el prior de movimiento suave, que en esta secuencia son muy apropiados al tener siempre el mismo punto de vista de la escena. En esta secuencia se busca estimar las deformaciones para una escena en la que la cámara no se mueve y toda la deformación ocurre en el mapa.

El sistema procesa la secuencia de forma robusta, sin perderse. El error oscila entre los 2-4 mm en los primeros frames, para después reducirse a partir del frame 230 y estabilizándose en los 2 mm. La deriva de escala se sitúa entre el 25 % para casi toda la secuencia, mejorando la situación inicial.



(a) Error RMSE

(b) Variación de escala

Figura 6.3: Resultados para la secuencia 1 con todas las contribuciones.

El sistema reproduce los latidos del corazón de forma robusta, como se ve en la figura 6.4.

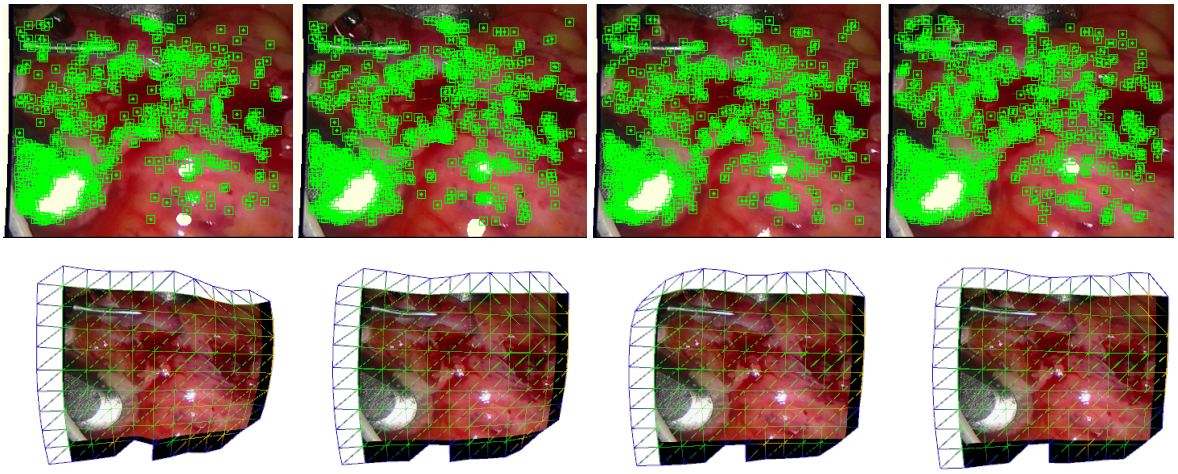


Figura 6.4: Estimación de los latidos del corazón en la secuencia 2.

Capítulo 7

Conclusiones

Este proyecto supone una de las primeras aproximaciones al procesado de secuencias médicas a través de un sistema de VSLAM deformable. Se ha modificado el sistema DefSLAM, uno de los primeros sistemas VSLAM monoculares capaces de sobrevivir en secuencias deformables exploratorias, para dotarle de capacidad para procesar escenas médicas y sobrevivir en ellas.

El sistema se ha validado en una secuencia de carácter exploratorio y otra de carácter estático. Para la evaluación, se ha diseñado un algoritmo que estima una solución Ground Truth mientras se procesa la secuencia, a través de la estimación estéreo de la superficie. Las contribuciones propuestas han conseguido que el sistema sea más robusto y preciso, reduciendo el error y la deriva de escala.

A pesar de los hitos logrados, sigue habiendo áreas de mejora:

- Al tratarse de escenas de bajo contraste, se detectan pocos puntos y por tanto hay menos emparejamientos en cada frame. Para aumentarlos,se podría añadir una búsqueda guiada por correlación tras el emparejamiento con ORB.
- Implementación de un Bundle Adjustment que refine el mapa completo, al igual que los sistemas SLAM rígidos.
- Detección en tiempo real de herramientas, de forma que no sea necesario conocer la escena a priori.
- Reducir los tiempos de cómputo en vistas a un sistema que pueda trabajar en tiempo real.

Capítulo 8

Bibliografía

[1] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.

[2] Raul Mur-Artal and Juan D Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Transactions on Robotics*, 33(5):1255–1262, 2017.

[3] Jakob Engel, Vladlen Koltun, and Daniel Cremers. Direct sparse odometry. *IEEE transactions on pattern analysis and machine intelligence*, 40(3):611–625, 2017.

[4] Jose Lamarca and JMM Montiel. Camera tracking for slam in deformable maps. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 0–0, 2018.

[5] Shaifali Parashar, Daniel Pizarro, and Adrien Bartoli. Isometric non-rigid shape-from-motion in linear time. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4679–4687, 2016.

[6] José Lamarca and J.M.M Montie. DeFSLAM: Monocular tracking and mapping of deforming scenes. *Draft*, 2019.

[7] Peter Mountney, Danail Stoyanov, and Guang-Zhong Yang. Three-dimensional tissue deformation recovery and tracking. *IEEE Signal Processing Magazine*, 27:14–24, 2010.

[8] Edward Rosten, Reid Porter, and Tom Drummond. Faster and better: A machine learning approach to corner detection. *IEEE transactions on pattern analysis and machine intelligence*, 32(1):105–119, 2008.

[9] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 7, 2006.

[10] Peter J Rousseeuw. Least median of squares regression. *Journal of the American statistical association*, 79(388):871–880, 1984.

[11] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013.

[12] Michael Burri, Janosch Nikolic, Pascal Gohl, Thomas Schneider, Joern Rehder, Sammy Omari, Markus W Achtelik, and Roland Siegwart. The euroc micro aerial vehicle datasets. *The International Journal of Robotics Research*, 2016.

[13] Alexey A Shvets, Alexander Rakhlin, Alexandr A Kalinin, and Vladimir I Iglovikov. Automatic instrument segmentation in robot-assisted surgery using deep learning.

Lista de Figuras

Lista de Tablas