Universidad de Alcalá
Escuela Politécnica Superior

Máster Universitario en Ingeniería Industrial

Trabajo Fin de Máster

Diseño de un sistema de realidad aumentada con aplicaciones a la cirugía mínimanente invasiva

Autor: Alberto Ceballos Fernández

Tutor: Daniel Pizarro Pérez

2016

UNIVERSIDAD DE ALCALÁ

ESCUELA POLITÉCNICA SUPERIOR

Máster Universitario en Ingeniería Industrial

Trabajo Fin de Máster

Diseño de un sistema de realidad aumentada con aplicaciones a la cirugía mínimanente invasiva

Autor: Alberto Ceballos Fernández

Tutor: Daniel Pizarro Pérez

Tribunal:

Presidente: Marta Marrón Romera

Vocal 1º: María Dolores Rodríguez Montero

Vocal 2º: Daniel Pizarro Pérez

Calificación: .....

Fecha: .....

"Empieza haciendo lo necesario, luego haz lo posible y de pronto empezarás a hacer lo imposible."
Francisco de Asís

Agradecimientos

A todos los que la presente vieren y entendieren.

Inicio de las Leyes Orgánicas. Juan Carlos I

Agradecer en primer lugar a Daniel Pizarro por haberme dado la oportunidad de realizar este interesante e innovador Trabajo Fin de Máster (TFM) y a Javier Macías por el apoyo prestado así como a todos mis compañeros del Grupo de Ingeniería Electrónica Aplicada a Espacios Inteligentes y Transporte (GEINTRA).

Por supuesto también agradecer a mis compañeros del Máster con quien he reído y sufrido, en especial a Luis Javier González y sus mágicas teorías.

Sobretodo dar las gracias también a mi familia en especial a mis padres y a mi hermana; sin vosotros no hubiera sido posible conseguir lo conseguido, aprender lo aprendido. Gracias de todo corazón.

A todos los que de una forma u otra me han ayudado a llegar a donde estoy. Mi más sincero agradecimiento.

Resumen

El presente trabajo pretende realizar una mejora en las técnicas presentes de Realidad Aumentada (AR) en operaciones de Cirugía Mínimamente Invasiva (MIS).

El objetivo último de este TFM es el del diseño y desarrollo de una herramienta software que ofrezca una reconstrucción incremental en tiempo real de un órgano del cuerpo humano. Dicha reconstrucción se llevó a cabo a partir de imágenes provenientes de una sóla cámara utilizada en operaciones de laparoscopia implementando algoritmos de Structure-from-Motion (SfM) y Multi-view Stereo (MVS) con el fin de mejorar la experiencia del cirujano en este tipo de operaciones médicas.

Palabras clave: Realidad Aumentada (AR), Cirugía Mínimamente Invasiva (MIS), Structure-from-Motion (SfM), Multi-view Stereo (MVS), Laparoscopia, 3D, Reconstrucción Incremental.

Abstract

This work is included within the computer vision and incremental 3D reconstruction research in Augmented Reality (AR) environments.

The main aim of this work is the design and development of a software tool that creates a three-dimensional-real-time reconstruction of an object using 2D images. This reconstruction was carried out using images from a single laparoscopic camera used in Minimal Invasive Surgery (MIS), implementing algorithms such as Structure from Motion (SfM) and Multi-View Stereo (MVS) algorithms. This work aims to improve current AR techniques in MIS operations.

Keywords: Augmented Reality (AR), Minimal Invasive Surgery (MIS), Structure-from-Motion (SfM), Multi-view Stereo (MVS), Laparoscopy, 3D, Incremental reconstruction.

Resumen extendido

Este Trabajo Fin de Máster (TFM) se engloba dentro de la línea de investigación relacionada con la visión artificial y reconstrucción 3D incremental en tiempo real en el ámbito de la Realidad Aumentada (AR).

El objetivo último de este TFM es el del diseño y desarrollo de una herramienta software que ofrezca una reconstrucción incremental en tiempo real de un órgano del cuerpo humano. Dicha reconstrucción se ha llevado a cabo a partir de imágenes provenientes de una sola cámara utilizada en operaciones de laparoscopia. En primer lugar, se ha diseñado un estudio del estado actual de la tecnología en este ámbito en particular. Posteriormente, se ha diseñado un sistema capaz de funcionar en tiempo real. El sistema incluye tecnologías recientes de reconstrucción tridimensional SfM y MVS. También se ha llevado a cabo el diseño de una herramienta de visualización basada en las librerías Point Cloud Library (PCL) y el visualizador Kit de Herramientas de Visualización (VTK). Además se ha desarrollado una interfaz gráfica interactiva para mejorar la interacción entre el usuario y la aplicación ofreciendo diversas funcionalidades. Por último se presentan los resultados de la reconstrucción satisfactoria de un conjunto de objetos.

El presente trabajo pretende realizar una mejora en las técnicas presentes de AR en operaciones de Cirugía Mínimamente Invasiva (MIS).

Índice general

Índice de figuras

Capítulo 1

Introducción

Desocupado lector, sin juramento me podrás creer que quisiera que este libro [...] fuera el más hermoso, el más gallardo y más discreto que pudiera imaginarse.

Miguel de Cervantes, Don Quijote de la Mancha

1.1 Presentación

En este documento se presenta el Trabajo Fin de Máster (TFM) correspondiente a la titulación Máster Universitario en Ingeniería Industrial (MUII) con la especialidad de Robótica y Percepción. Este TFM ha sido desarrollado dentro del Grupo de Ingeniería Electrónica Aplicada a Espacios Inteligentes y Transporte (GEINTRA) del departamento de Electrónica de la Universidad de Alcalá de Henares (UAH). Este TFM se enmarca dentro de la línea de investigación relacionada con la visión artificial. Más en concreto en reconstrucción de modelos 3D a partir de imagen medica en operaciones de Cirugía Mínimamente Invasiva (MIS) así como Realidad Aumentada (AR) en medicina.

El objetivo principal del presente TFM es el del desarrollo de una herramienta software que ofrezca una reconstrucción incremental en tiempo real de un órgano del cuerpo humano a partir de imágenes provenientes de una sola cámara utilizada en operaciones de laparoscopia. El resultado de este TFM completará una herramienta de AR para la mejora de la resección de tumores en el útero. Dicha herramienta ha sido desarrollada por la UAH y la Universidad de d'Auvergne (Francia).

1.2 Realidad aumentada en medicina

En las técnicas de MIS, la intervención quirúrgica se realiza a través de una pantalla que muestra las imágenes captadas por un endoscopio que, junto con el resto de instrumental, se introduce en el cuerpo a través de pequeñas incisiones. Las técnicas MIS mejoran la cirugía clásica en diferentes aspectos, magnificando la visión del experto y minimizando el impacto de las cicatrices y el post-operatorio del paciente.

En el procedimiento médico actual, el diagnóstico y planificación de la intervención quirúrgica se basan en el análisis de información capturada por el paciente con antelación a la operación lo que se conoce como información preoperatoria. Ésta se obtiene a partir de diferentes modalidades de imagen que permiten inspeccionar el interior de los órganos y tejidos del paciente. Las modalidades más comunes

son la Resonancia Magnética (IRM), los Ultrasonidos (US) y la Tomografía Computerizada (CT). La información obtenida mediante los datos preoperatorios es muy rica, pudiéndose distinguir con precisión el volumen que ocupa cada órgano así como estructuras complejas como capilares, ligamentos o tumores. Esto en teoría permite planificar una intervención quirúrgica con alta precisión. Sin embargo, durante la cirugía MIS, el especialista debe considerar mentalmente la posición de los órganos y las estructuras internas del paciente en la imagen de monitorización. Este proceso mental puede ser extremadamente difícil, incluso en aquellos especialistas con gran experiencia (ver Figura 1.1).

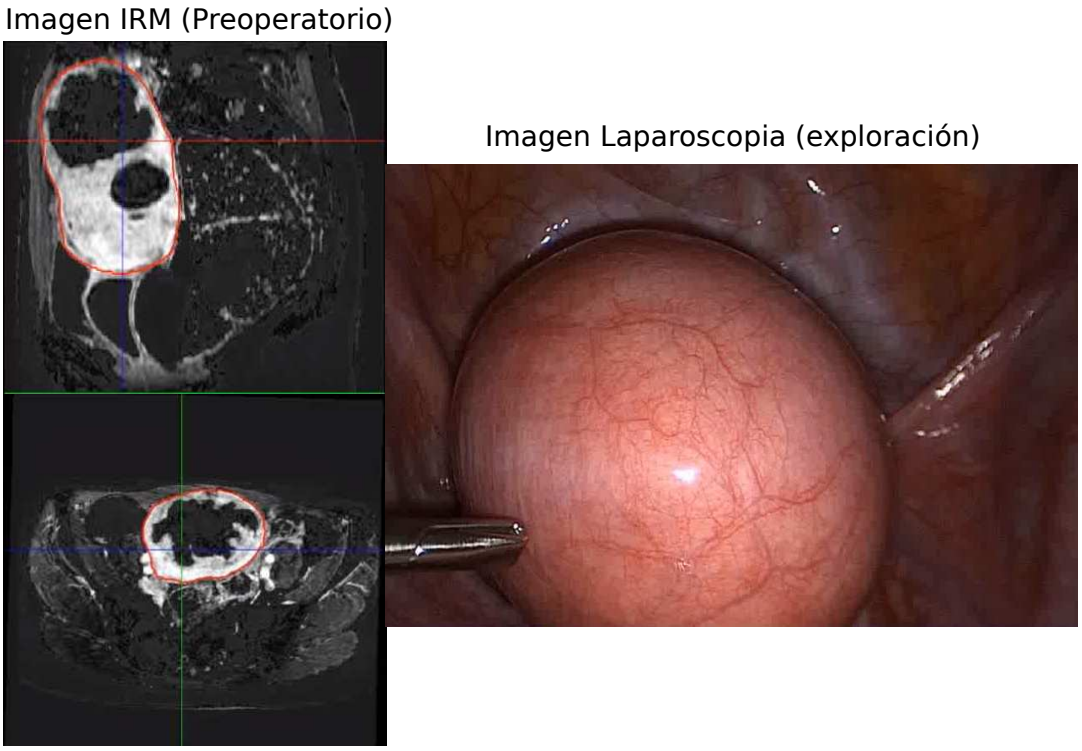

Figura 1.1: IRM vs Laparoscopia. Izquierda: Imágenes provenientes de IRM de la fase preoperatoria. Derecha: Imagen de la fase de exploración laparoscópica

Por ejemplo, la Figura 1.1 muestra un útero mediante dos modalidades de imagen diferentes: IRM y laparoscopia. El útero de la figura presenta dos tumores. Dichos tumores se pueden apreciar como dos manchas negras en las imágenes de IRM (izquierda). Si bien es cierto que la imagen es mucho más nítida y clara en la imagen proveniente de la cámara de laparoscopia, los tumores no pueden ser apreciados ya que es una modalidad de imagen bidimensional en la que sólo se puede apreciar la superficie. El cirujano debe de intuir donde están localizados dichos tumores en la imagen, generalmente a partir de marcas naturales distinguibles en ambas modalidades (trompas de falopio o ligamentos).

1.2.1 Cirugía del útero mediante laparoscopia

La laparoscopia es un caso particular de cirugía MIS centrada en la cavidad pélvica abdominal. En este tipo de cirugía el abdomen es inflado con gas y los instrumentos y la fibra óptica son introducidos en el abdomen mediante válvulas especiales llamadas trócares, que mantienen la presión de gas. El presente TFM se centra en la cirugía del útero mediante laparoscopia como entorno de aplicación y aborda, en concreto, la resección de los denominados miomas o fibroides uterinos. Los miomas son los tumores benignos más comunes en el tracto genital femenino y pueden causar problemas que incluyen anomalías en la menstruación, dolores y en algunos casos infertilidad. Los miomas pueden crecer en diferentes

partes del útero, siendo un emplazamiento común el interior de la pared uterina; son los denominados miomas subserosos. El tratamiento más común para dichos tumores es su extracción mediante cirugía por laparoscopia. Cuando los miomas son de un tamaño mediano (hasta 5 cm de diámetro) su presencia no afecta a la apariencia externa del útero, siendo por tanto difíciles de localizar en las imágenes. Por esta razón el especialista debe consultar previamente los datos obtenidos en preoperatorio y que generalmente consisten en imágenes de IRM como se vio en la Figura 1.1. En la Figura 1.2 se pueden ver varios fotogramas de una operación de laparoscopia así como la disposición de la cánula uterina en dichas operaciones.

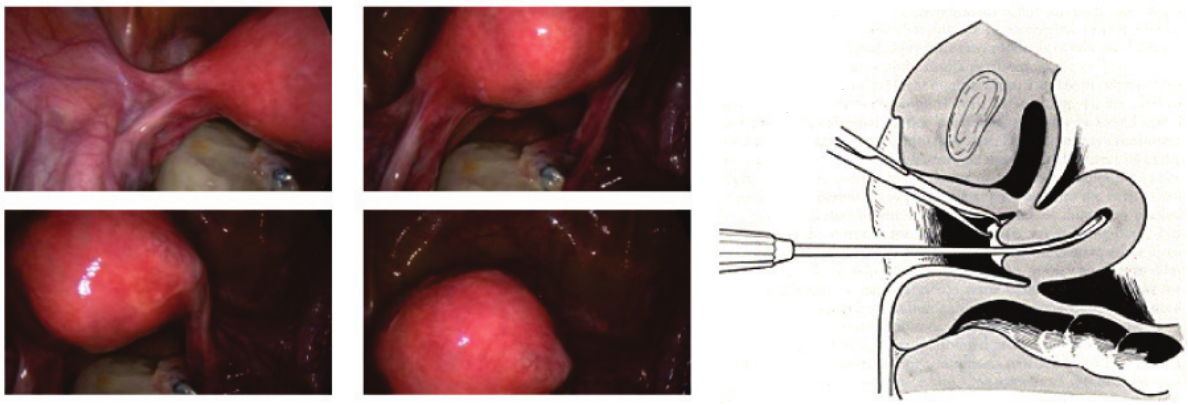

Figura 1.2: Fotogramas de un video de la fase exploratoria de una operación mediante laparoscopia

1.3 Motivación

Las técnicas MIS presentan algunos inconvenientes en comparación con la cirugía tradicional. Principalmente existe una pérdida de realimentación táctil y una disminución de la percepción tridimensional del especialista al utilizarse la imagen proveniente de la cámara endoscópica para realizar la operación. Como consecuencia, estas técnicas requieren un grado considerable de experiencia y entrenamiento por parte del cirujano.

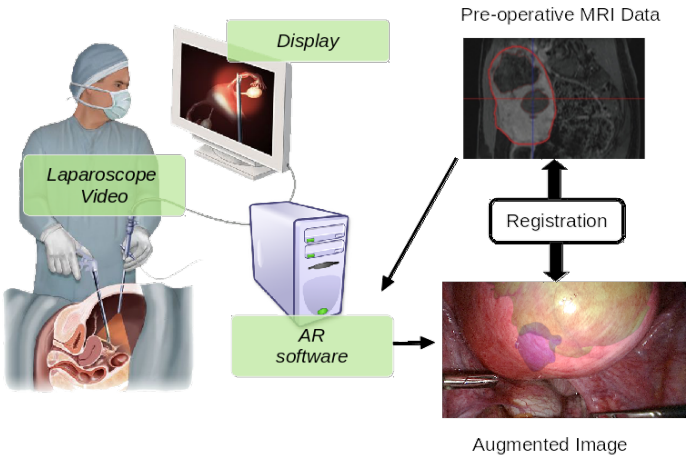

Figura 1.3: Diagrama general del sistema propuesto

En la Figura 1.3 se muestra el sistema de AR en MIS en el que se incluye el sistema propuesto en este TFM. En este sistema el cirujano ve en tiempo real la imagen proveniente de la cámara laparoscópica al mismo tiempo que un software de AR se encarga de hacer una reconstrucción 3D del útero incluyendo la información del proceso preoperatorio. Esta fase, que relaciona la información preoperatoria con el modelo 3D, se llama registro. Dicha fase es una tarea que parece sencilla en el diagrama anterior, pero en realidad es una tarea compleja que se puede desglosar en dos elementos fundamentales: reconstrucción incremental 3D en tiempo real y localización de la información de la fase preoperatoria en dicha reconstrucción 3D.

Este TFM se enmarca dentro de los trabajos desarrollados en [1] [2] donde se propone una solución al problema del registro entre datos preoperatorios y las imágenes de laparoscopia. El método propuesto se divide en dos fases fundamentales: fase exploratoria y fase operativa. El objeto de la fase exploratoria es obtener suficientes vistas del útero, de manera que su superficie pueda ser estimada mediante técnicas de Structure-from-Motion (SfM) (2.3.1) con una alta calidad. En esta fase se requiere que la laparoscopia se mantenga aproximadamente inmóvil mientras el útero es manipulado externamente mediante una cánula uterina. De este modo, el movimiento principal de la escena se debe al movimiento del útero mostrando la parte posterior, anterior y lateral del mismo en las imágenes. La fase exploratoria requiere menos de dos minutos de interrupción del protocolo médico y es una fase controlada, donde no existen oclusiones debidas a instrumentos ni movimientos bruscos de la cámara o el útero. Durante la fase operativa las imágenes muestran la manipulación normal del útero por parte del especialista, donde pueden existir oclusiones debidas a los instrumentos quirúrgicos, movimientos bruscos de la cámara y el útero, brillos debidos a líquidos introducidos para la limpieza e hidratación del tejido, etc.

El método propusto en [1] y [2] permite realizar el registro en tiempo real y de manera robusta. En [5] se demuestra mediante un estudio clínico que el sistema de AR mejora sustancialmente la precisión por parte del especialista.

Sin embargo, los trabajos desarrollados en [1] y [2] presentan ciertas limitaciones. En primer lugar, se debe realizar una segmentación manual del útero en la imagen. Además, el cirujano debe realizar una exploración previa del útero almacenando una batería de imágenes. A esas imágenes se les aplica posteriormente el procesado SfM para obtener la reconstrucción 3D. El cirujano no es consciente de si los fotogramas almacenados en dicha fase exploratoria dan lugar a una reconstrucción correcta hasta que no se ha realizado el procesado. Todo este proceso puede llevar dos minutos y, si al cabo de esos dos minutos el modelo no es satisfactorio el proceso ha de repetirse lo cual puede demorar decenas de minutos la intervención siendo los costes de tiempo en quirófano bastante elevados. Por lo tanto, una herramienta de reconstrucción incremental en tiempo real dónde el cirujano pudiera tener certeza de que el movimiento de la cámara es el correcto para la reconstrucción 3D mejoraría la eficiencia de la intervención en lo que a tiempos se refiere.

1.4 Descripción del sistema

Este TFM consiste en resolver la primera parte del registro mencionado anteriormente: desarrollar una herramienta de reconstrucción incremental 3D en tiempo real de la superficie del órgano a partir de las imágenes provenientes de la cámara laparoscópica.

Para llevar a cabo este TFM, se partirá exclusivamente de la información obtenida en la fase exploratoria de la operación, de tal forma que se dispone de un conjunto de imágenes bidimensionales del objeto a reconstruir. Hay que tener en cuenta que la cámara debe estar correctamente calibrada para poder realizar un correcto modelado del objeto.

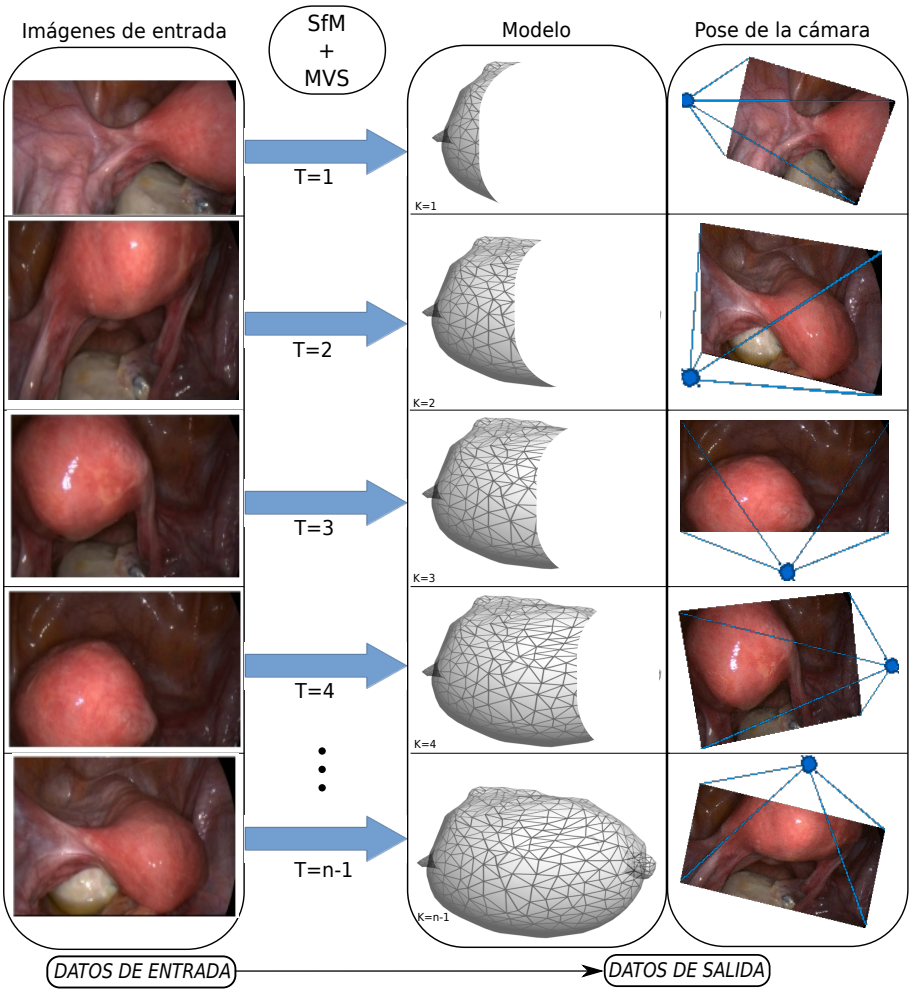

Figura 1.4: Modelado general en tiempo real

El sistema propuesto mostrado en la figura 1.4 recibe como entrada los nuevos fotogramas que la cámara va generando. Con esos fotogramas se realiza un proceso de SfM incremental, dónde se obtiene como resultado la pose de la cámara por cada fotograma y un conjunto de puntos tridimensionales dispersos en tiempo real. Ese conjunto de puntos tridimensionales será añadido a un conjunto de puntos tridimensionales global que irá incrementándose a medida que las imágenes son generadas (normalmente $\sim 30 fps$). Ese conjunto global de puntos será la reconstrucción tridimensional del objeto.

Cabe destacar que para que el proceso de SfM pueda ser llevado a cabo, se debe suponer que el objeto a reconstruir se trata de un elemento rígido. En el caso del útero, se ha demostrado que dicho modelo es suficientemente preciso para la aplicación que se plantea en este TFM.

Sin embargo las técnicas de SfM en la mayoría de los casos generan un conjunto muy disperso de puntos del objeto. Para mejorar la percepción del modelo reconstruido se utilizan técnicas de densificado a partir de algoritmos Multi-view Stereo (MVS). Este proceso de densificado precisa a su entrada datos generados por el proceso de SfM como la pose de la cámara. Las técnicas MVS generan con ese dato de entrada junto con el fotograma correspondiente, un conjunto denso de puntos

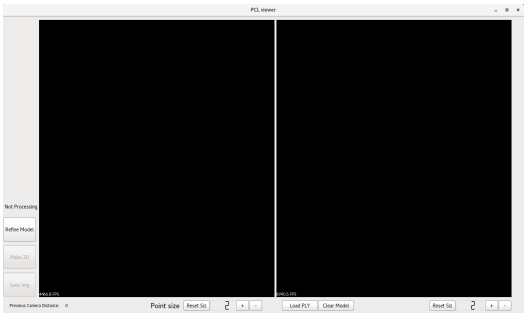

Figura 1.5: Interfaz gráfica de usuario

cuyos colores serán los correspondientes al color del objeto reconstruido. De igual manera, los puntos obtenidos son incluidos en tiempo real en un conjunto de puntos global, siendo en este caso un modelo denso del objeto como se dijo anteriormente.

Como resultado de todo este proceso se obtendrá la reconstrucción 3D completa del objeto en tiempo real así como la localización de la cámara en cada imagen (ver Figura 1.4). Dicha información se representará en una interfaz gráfica interactiva como la que se muestra en la Figura 1.5, de tal forma que sea de utilidad en la intervención quirúrgica.

1.5 Objetivos

El principal objetivo del presente TFM es la reconstrucción incremental de un órgano como el útero a partir de imágenes provenientes de operaciones de laparoscopia. De tal modo que se pueda visualizar el órgano reconstruido en cada instante de tiempo de forma incremental, permitiendo al especialista observar el área del órgano correctamente reconstruida. La supervisión del especialista es crucial para garantizar una reconstrucción completa y correcta.

Los objetivos principales son los siguientes:

- Estudio de técnicas de reconstrucción incremental a partir de imágenes con bajo contenido de textura y que permitan su funcionamiento en tiempo real
- Desarrollo de un algoritmo de reconstrucción incremental a partir de imágenes obtenidas durante la fase exploratoria.
- Diseño y desarrollo de una interfaz gráfica para facilitar la visualización de la reconstrucción en tiempo real.

El desarrollo del TFM propuesto comprende el diseño e implementación de los siguientes algoritmos y tareas:

1. Desarrollo e implementación de un sistema que establezca correspondencias de puntos en imágenes consecutivas. Para ello se recurre a técnicas basadas en flujo óptico o técnicas de obtención de correspondencias de puntos en imágenes.
2. Desarrollo e implementación de un sistema que obtenga la posición tridimensional del objeto con respecto a una nueva imagen a partir de un modelo tridimensional previo y conocido de dicho objeto.
3. Desarrollo e implementación de un algoritmo para realizar la reconstrucción incremental del objeto a partir de las correspondencias de puntos encontrados en la nueva imagen.
4. Desarrollo e implementación de un algoritmo de visualización tridimensional de la reconstrucción en cada instante de tiempo.
5. Desarrollo e implementación de una interfaz gráfica de usuario amigable para una fácil interacción entre el usuario y el modelo reconstruido en la fase exploratoria.

1.6 Organización de la memoria

El presente TFM se basa en cinco capítulos principales:

- *Estado del arte:* En este capítulo se detallará el último estado público del ámbito de conocimiento que se aborda en este TFM. Más en concreto se describirá el estado de la AR en medicina así como los avances y el estado de las técnicas de reconstrucción 3D partiendo de imágenes actuales.

- *Modelado del problema:* En este capítulo se describirá qué estrategias se van a adoptar para la resolución del problema al que se enfrenta este TFM. Para ello se describirán los algoritmos a utilizar para modelar el espacio de trabajo y el objeto a reconstruir.

- *Desarrollo/Solución:* En este capítulo se detallará el desarrollo completo de la reconstrucción en tiempo real así como el desarrollo de la interfaz gráfica.

- *Resultados:* Este capítulo consiste en la evaluación cualitativa de los resultados de la herramienta desarrollada.

- *Conclusiones y futuros trabajos:* Por último, en este capítulo final se detallarán las conclusiones obtenidas así como los trabajos futuros que podrán complementar y continuar este TFM.

Capítulo 2

Estado del Arte

Y así, del mucho leer y del poco dormir, se le secó el cerebro de manera que vino a perder el juicio.

Miguel de Cervantes Saavedra

2.1 Introducción

En este capítulo se detallará el estado actual de la tecnología en el ámbito del presente Trabajo Fin de Máster (TFM).

2.2 Técnicas de AR en MIS

Existe una cantidad considerable de trabajos científicos cuyo objetivo es la aplicación de Realidad Aumentada (AR) para solventar algunas de las limitaciones mencionadas en las técnicas de Cirugía Mínimamente Invasiva (MIS) [6] [7]. El objetivo general de estas técnicas es fusionar en tiempo real la información de video, captada por la endoscopia, con alguna modalidad de imagen que muestre la posición de las estructuras internas del órgano, invisibles para la cámara.

Los trabajos del estado del arte se pueden clasificar en dos tipos, atendiendo a si la información que se desea visualizar en la imagen proviene de una modalidad preoperatoria o si, por el contrario, dicha información se obtiene durante la operación y de manera síncrona con las imágenes de la endoscopia. El segundo caso requiere determinar una transformación rígida entre los sistemas de referencia de la cámara y el resto de modalidades intraoperatorias.

En cirugía laparoscópica se han propuesto métodos en la literatura [8] [9] que obtienen dicha transformación mediante marcadores magnéticos o marcas artificiales en el tejido que sean visibles a través de las diferentes modalidades. Estos métodos pueden ser muy precisos pero requieren alterar considerablemente el protocolo médico y también equipos de imagen intraoperativa, como puede ser Tomografía Computerizada (CT) intervencional, que en la mayoría de los casos no son parte del equipamiento estándar.

En el primer grupo de métodos la información detallada del interior del paciente es capturada en exámenes preoperatorios, generalmente mediante técnicas Resonancia Magnética (IRM) o CT, semanas antes de la intervención [10][1][11][12]. El principal problema de estos métodos es la necesidad de actualizar la información preoperatoria en cada instante de tiempo durante la cirugía. Se deben tener en cuenta

las deformaciones de los órganos internos, debidas a la presencia de gas en el abdomen y la posición y orientación desconocida de la laparoscopia en cada instante de tiempo. Este problema requiere de la estimación de una función de registro, que incluye deformaciones entre la modalidad de imagen preoperatoria, generalmente tridimensional y las imágenes de naturaleza bidimensional provenientes de la cámara de laparoscopia. Los trabajos previos en esta linea se pueden dividir entre aquellos que cuentan con imágenes de endoscopia monocular [10][1][2] o aquellos que disponen de endoscopia estéreo [11][12][13]. La endoscopia estéreo permite obtener superficies 3D de los órganos mediante técnicas de triangulación lo que facilita el registro. La mayoría de estos trabajos requieren de un registro manual [11] realizado a través de un interfaz de usuario o semi-manual utilizando marcas naturales de fácil identificación en las imágenes [8][13][14]. Mediante el registro manual se superpone el modelo preoperatorio con la imagen de laparoscopia, sirviendo normalmente de punto de referencia o inicialización de los algoritmos. Los modelos de deformación utilizados varían desde la suposición de rigidez [8] hasta el uso de modelos biomecánicos [14], que imitan el comportamiento del tejido cuando es sometido a una fuerza.

En [1][2] se propone un sistema de AR para la visualización de miomas subserosos en cirugía laparoscópica del útero. Con respecto al estado del arte existente en técnicas AR, en dichos trabajos se realiza el registro y tracking en tiempo real del útero de manera robusta, limitando la intervención manual a la segmentación del útero en un conjunto reducido de imágenes. La principal suposición es que, si bien existe una deformación apreciable entre las imágenes preoperatorias y las imágenes laparoscópicas, el útero se comporta como un objeto rígido en movimiento durante el periodo intraoperatorio donde la visualización AR es necesaria. De este modo, se propone un sistema de registro dividido en dos fases. En una se obtiene un modelo tridimensional de la superficie del útero utilizando la técnica de Structure-from-Motion (SfM) [15], a partir de diferentes imágenes tomadas con la cámara de laparoscopia y que muestran diversos puntos de vista del útero. Esa reconstrucción es densa y mucho más detallada y completa que aquella obtenida por un par de imágenes estéreo. La reconstrucción obtenida se registra de manera automática con los datos preoperatorios. Una vez registrados los volúmenes preoperatorios, en el segundo paso del algoritmo se realiza el seguimiento del útero en las imágenes, suponiéndolo de geometría rígida durante el resto de la cirugía. La suposición de rigidez permite al sistema funcionar en tiempo real y de manera robusta.

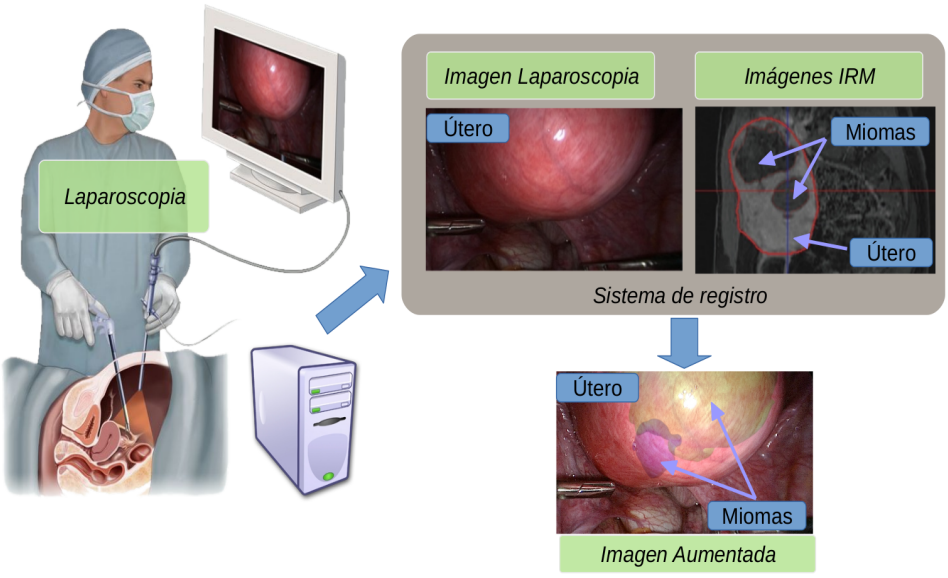

Figura 2.1: Sistema de AR en MIS propuesto en [1][2]

Los trabajos publicados en [1][2] presentan, no obstante, diversos problemas que dificultan su aplicación en un caso real. En primer lugar, la construcción del modelo mediante SfM requiere de un sistema de segmentación manual del útero en un conjunto de imágenes (20 imágenes aproximadamente). La elección de dichas imágenes, así como su segmentación, es crucial para obtener un buen modelo tridimensional. En muchos casos se obtienen reconstrucciones parciales o no satisfactorias, lo que implica repetir la adquisición y segmentación. Este hecho puede demorar varias decenas de minutos la intervención. Otra limitación destacable de [2] es el uso de una transformación afín para modelar la deformación existente entre el estado preoperatorio del útero y su estado intraoperatorio. Dicho modelo puede ser muy inexacto en determinados casos en los que el útero es presionado de forma no uniforme por diferentes órganos. Los errores de registro son importantes ya que se propagan a todos los instantes de tiempo posteriores. Por último, la hipótesis de rigidez durante el seguimiento del útero puede ser demasiado restrictiva en aquellos casos en los que la manipulación del útero mediante los instrumentos produzca deformaciones significativas.

Esto, junto con lo mencionado en el apartado 1.3 hace que sea necesario el desarrollo de una herramienta de reconstrucción 3D incremental en tiempo real en la que se elimine la necesidad de segmentación manual y que permita al especialista supervisar la reconstrucción en tiempo real.

2.3 Reconstrucción 3D basado en imágenes

El objetivo de una reconstrucción 3D basada en imágenes 2D podría ser descrito como *"dado una serie de fotogramas de un objeto o escena, estimar la forma 3D más probable bajo los supuestos de puntos de vista así como las condiciones de iluminación"*. Dicha definición resume de una forma muy precisa la complejidad de la tarea a la que se enfrenta este TFM. Si dichos elementos no son conocidos el problema es casi imposible de resolver ya que existen varias soluciones que no son las correctas que se corresponderían con dichos fotogramas iniciales. Sin embargo, teniendo en cuenta varios supuestos como por ejemplo suponiendo que se trata de superficies rígidas Lambertianas (superficies donde al variar el punto de vista, su luminancia no cambia) se puede producir reconstrucciones muy detalladas.

Existen muchas técnicas que pueden ser usadas para extraer geometría de fotogramas 2D: textura, sombras, contornos así como correspondencia estéreo entre fotogramas. Las tres últimas técnicas presentan muy buenos resultados, siendo la correspondencia estéreo la más exitosa en términos de robustez y número de aplicaciones. En este apartado se va a detallar el funcionamiento de SfM, con el cual se obtiene la pose de la cámara y una nube de puntos dispersa y Multi-view Stereo (MVS), cuyo resultado es una densificación del objeto a partir de fotogramas y poses conocidas de la cámara por cada fotograma.

2.3.1 Structure from Motion

Los algoritmos de SfM reciben como entrada un conjunto de imágenes y generan, como salida, dos elementos principales: La pose de la cámara por cada imagen y un conjunto de puntos 3D correspondientes a una escena u objeto rígido. Los algoritmos SfM consisten por lo general en los siguientes pasos:

- Detección e identificación de características en cada imagen de entrada.
- Correspondencia de las características detectadas entre imágenes.
- Selección de puntos clave provenientes de la correspondencia anterior.
- Obtención del modelo 3D (nube de puntos dispersa) a partir de los puntos claves.

- Refinado del modelo 3D usando *bundle adjustment*.

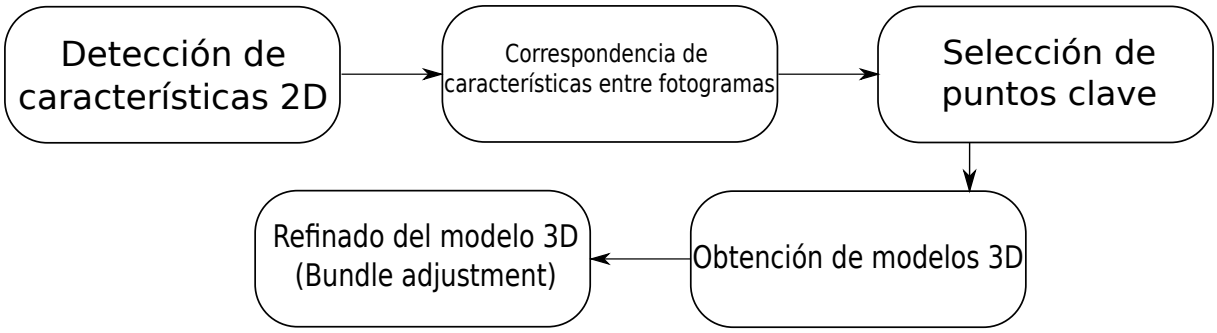

Figura 2.2: Diagrama de flujo genérico de SfM

A lo largo de la historia se han producido grandes avances en las técnicas de SfM que han evolucionado SfM a lo existente hoy en día. Dichos avances consistieron en lo siguiente:

- Uno de los avances clave para el desarrollo de SfM fue el uso de RANSAC [16] para estimar de manera robusta la geometría epipolar entre dos o más perspectivas con una correspondencia de puntos entre sí.
- Se mejoraron las técnicas para la obtención de una reconstrucción euclídea de múltiples cámaras o, lo que es lo mismo, estimar los parámetros extrínsecos de cada cámara.
- También, los avances se centraron en crear correspondencias (matching) de mayores cantidades de puntos; al final del siglo veinte dichos algoritmos eran capaces de computar robustamente grandes cantidades de datos de imágenes provenientes de secuencias de video permitiendo aplicar dichos algoritmos de SfM en la industria cinematográfica para efectos especiales o ediciones de video [17].
- El aliciente final para conseguir un buen modelo SfM a gran escala fue el de mejorar la fase de correspondencia o matching ya que cada imagen era correspondida con cada una de las imágenes, lo cual llevaba un coste computacional muy elevado. Para solventar eso, se mejoró la eficiencia en la indexación [18] y se incluyeron descriptores de alta calidad permitiendo realizar la correspondencia de millones de imágenes (ver Figura 2.3).
- Además se simplificó el gráfico de conectividad de los puntos característicos (tracks) [19] y paralelización ([20], [21]).

La etapa final del proceso SfM consiste en refinar el modelo obtenido. Ese refinado se puede resolver usando técnicas llamadas *Bundle Adjustment*.

2.3.1.1 Bundle Adjustment

Bundle Adjustment es una forma iterativa de resolver el problema de SfM. Aunque el Bundle Adjustment no es estrictamente una parte de SfM, es esencial para el refinado del modelo SfM. Se llama Bundle Adjustment al problema de refinar la reconstrucción visual para producir una estructura tridimensional y unos parámetros de la cámara óptimos.

Para conseguir reducir el error de reproyección se utiliza un algoritmo de minimización no lineal robusto que intenta ajustar el grupo de lineas que unen las proyecciones y los puntos físicos de tal forma que el error de reproyección sea el mínimo posible.

Figura 2.3: Modelos de SfM a gran escala; Izquierda: Modelo de la ciudad de Dubrovnik. Derecha: Modelo de la catedral de San Marcos de Venecia

Dado un conjunto de parámetros de la cámara P_i y un conjunto de tracks M^j, m_i^j donde M^j representa la coordenada 3D de un track y m_i^j la proyección 2D. En definitiva, la tarea de Bundle Adjustment consiste en minimizar el último error cuadrático de una función de coste.

$$E(P, M) = \sum_j \sum_{i \in V(j)} |P_i(M^j) - m_i^j|^2 \tag{2.1}$$

$V(j)$ representa la lista de índices de las cámaras donde M^j es visible y $P_i(M^j)$ representa la coordenada reproyectada de la imagen 2D del punto 3D (M^j) de la cámara i usando los parámetros P_i de la cámara. $E(P, M)$ se mide normalmente en píxeles cuadrados.

Según las necesidades de procesamiento se aplica un proceso de Bundle Adjustment más o menos optimizado. Existen varios métodos de optimización de la función de coste:

- Método de descenso por el gradiente
- Método de Newton-Rhapson
- Método de Gauss-Newton
- Método de Levenberg-Marquardt

- **Método de descenso por el gradiente:**

Éste método consiste en un algoritmo de optimización de primer orden. Para encontrar un mínimo local de una función utilizando descenso de gradiente, se toma medidas proporcionales al negativo de la pendiente (o de la pendiente aproximada) de la función en el punto actual.

Éste método es robusto cuando $\mathcal{X}$ se encuentra lejos del punto óptimo pero presenta una convergencia final muy pobre.

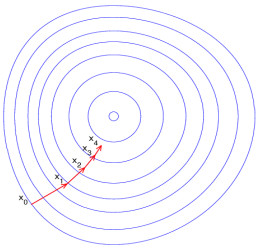

Figura 2.4: Optimización de la función de coste. Método de descenso por el gradiente.

- **Método de Newton-Rhapson:**

Es un método de optimización de segundo orden. El método de Newton a menudo puede converger con notable rapidez, especialmente si la iteración comienza lo suficientemente cerca del resultado deseado. Para funciones cuadráticas éste método converge en una sola iteración. Para otras funciones generales la convergencia asintótica del método es cuadrática.

La desventaja de éste método es la carga computacional que presenta.

- **Método de Gauss-Newton:**

El algoritmo de Gauss-Newton es un método utilizado para resolver problemas no lineales de mínimos cuadrados. Es una modificación del método de optimización de Newton que no usa segundas derivadas y se debe a Carl Friedrich Gauss.

- **Método de Levenberg-Marquardt:**

Éste método es el más utilizado en Bundle Adjustment. Es un algoritmo iterativo que minimiza una función de coste no lineal modificando los valores de un conjunto de parámetros. Éste método comienza con un conjunto de valores de los parámetros de la cámara así como las mediciones de error para dichos parámetros. En cada iteración se calcula un nuevo conjunto de parámetros reduciendo el error hasta llegar a la solución óptima. La modificación que se produce entre una iteración y la siguiente es aquella que minimiza el error en un cierto grado.

La particularidad de este algoritmo consiste en que el error se recalcula en cada iteración de tal forma que si en una iteración se encuentra una modificación de parámetros que reduzca el error en la siguiente el error tendrá un valor menor. Si en una iteración no encuentra una modificación que satisfaga el error lo aumenta. De esta forma, gracias al uso del límite de reducción de error éste método presenta un comportamiento que oscila entre el de una búsqueda "greedy" y el método de optimización de Gauss-Newton mencionado anteriormente.

2.3.2 Multi-View Stereo

MVS es el término general para referirse a un conjunto de técnicas que utilizan correspondencia estéreo como principal elemento usando un número mayor a dos imágenes [22] [23]. Los orígenes de MVS se remontan al término estereópsis humana o, lo que es lo mismo, el fenómeno dentro de la percepción visual por el cual a partir de dos imágenes ligeramente diferentes del mundo físico el cerebro es capaz de recomponer una tridimensional [24]. En los primeros intentos para solucionar este problema éste se consideró como un problema computacional [25].

Aunque MVS comparte los mismos principios que los algoritmos estéreo clásicos, los algoritmos MVS están diseñados para trabajar con imágenes con muchos puntos de vista así como con un gran número de imágenes, incluso del orden de millones de imágenes. Por ejemplo, aplicaciones industriales para mapeado 3D [[26], [27], [28]] procesan millones de fotogramas de miles de kilómetros al mismo tiempo, reconstruyendo así grandes superficies metropolitanas o, incluso, el mundo entero.

Los algoritmos basados en MVS parten de que se conoce la correspondencia paramétrica entre imágenes de un mismo conjunto.

La Figura 2.5 muestra un esquema de un proceso MVS genérico. MVS se usa en multitud de aplicaciones, pero el procedimiento es similar:

- Conjunto de imágenes.
- Cómputo de los parámetros extrínsecos obtenidos de SfM de la cámara para cada imagen.

- Reconstrucción 3D de la geometría de la escena del set de imágenes y su correspondencia con la pose de la cámara (parámetros extrínsecos).
- Reconstrucción de los materiales de la escena (densificado del modelo).

El término *parámetros de la cámara* se refiere a un conjunto de valores que describen la configuración de la cámara o, lo que es lo mismo, información de la pose de la cámara (localización y orientación) así como las propiedades intrínsecas de la cámara (distancia focal y tamaño del sensor).

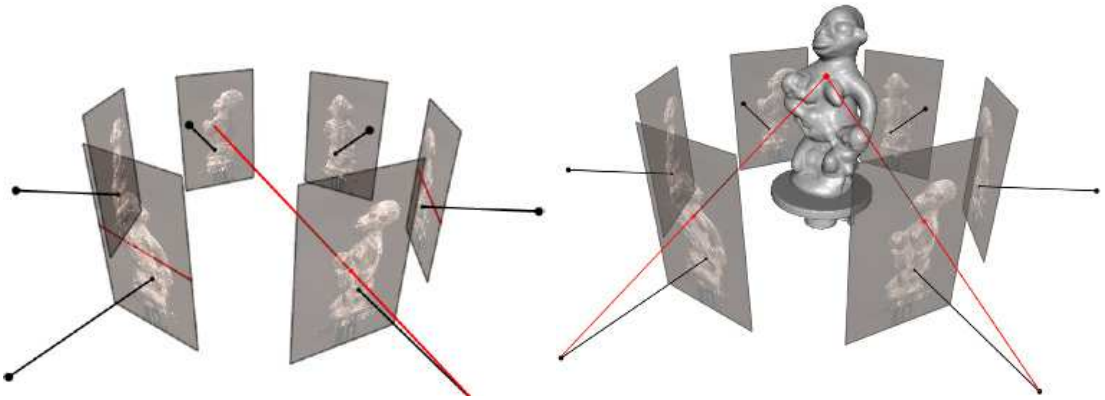

Figura 2.5: Correspondencia de imágenes con parámetros de la cámara conocidos. Izquierda: La geometría 3D de la escena define la correspondencia entre píxeles en diferentes imágenes. Derecha: Cuando se conocen los parámetros de la cámara, la correspondencia entre píxeles de una imagen en otra es un problema reducido a un problema de búsqueda en 1D

Conociendo los parámetros de cada cámara, solucionar el problema de correspondencia 3D es exactamente equivalente a solventar el problema de correspondencia entre imágenes. El proyectar un punto 3D en un set de cámaras visibles establece una correspondencia única entre las coordenadas proyectadas de cada imagen.

Para encontrar los píxeles correspondientes en otras imágenes, dado un pixel en una imagen se necesitan dos ingredientes fundamentales:

- Generar eficientemente posibles píxeles candidatos en otras imágenes.
- La probabilidad de que el pixel es el que corresponde en las demás imágenes.

Si no se conoce la geometría de la cámara, por ejemplo en el caso de flujo óptico, cada pixel en una imagen puede ser correspondido con otro pixel en otra imagen. Por el contrario, si se conocen los parámetros de la cámara (y se asume rigidez) el problema de matching se reduce a un problema de búsqueda 1D. Un pixel en una imagen genera un rayo óptico en tres dimensiones que pasa por el píxel y el centro de la cámara. Por su parte, el pixel correspondiente en la otra imagen únicamente recae en la proyección de ese rayo óptico en la segunda imagen. Las diferentes constantes geométricas que se originan cuando varias cámaras apuntan hacia el mismo punto 3D desde diferentes puntos de vista es lo que se conoce como geometría epipolar [15].

Por último, para evaluar como de probable es la correspondencia del candidato en cuestión, existe una gran documentación en la literatura de lo que es conocido como *photo-consistency measures* o consistencia del punto en la imagen. Estos algoritmos consisten en estimar la probabilidad de que dos o más píxeles estén en correspondencia [29].

En el caso que aborda este TFM, la técnica MVS complementa al proceso de SfM densificando, a partir de las poses generadas por SfM y las imágenes de entrada, el modelo generado lo cual mejora significativamente el resultado obtenido.

Capítulo 3

Modelado del problema

Leer, leer, leer; ¿seré lectura mañana también yo?
¿Seré mi creador, mi criatura, seré lo que pasó?

Miguel de Unamuno, Antología Poética

3.1 Introducción

En este capítulo se describirá el planteamiento y la descripción del problema, modelando todos los elementos del mismo matemáticamente; desde la imagen y el sistema de coordenadas hasta el modelo a reconstruir y la pose de la cámara.

3.2 Modelados de proyección de la imagen

Se tiene un espacio tridimensional con dos sistemas de referencia: El sistema de referencia del mundo $\mathcal{R}_w$ y el sistema de referencia de la cámara $\mathcal{R}_c$. $\mathcal{R}_w$ se define como $\mathcal{R}_w = \{O_w; \vec{e_{w1}}, \vec{e_{w2}}, \vec{e_{w3}}\}$. Siendo O_w el punto y $\{\vec{e_{w1}}, \vec{e_{w2}}, \vec{e_{w3}}\}$ la base del mismo. Por su parte $\mathcal{R}_c$ se define como $\mathcal{R}_c = \{O_c; \vec{e_{c1}}, \vec{e_{c2}}, \vec{e_{c3}}\}$ siendo de igual manera O_c el punto y $\{\vec{e_{c1}}, \vec{e_{c2}}, \vec{e_{c3}}\}$ la base del mismo.

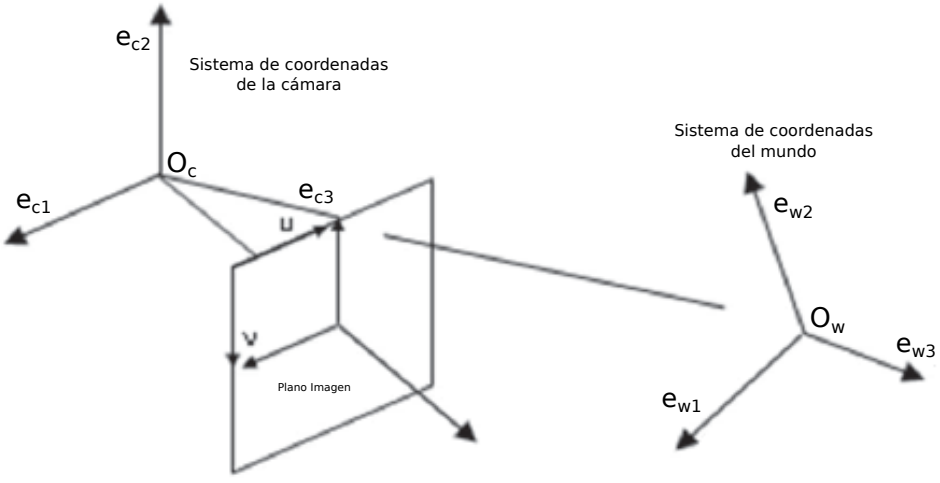

Figura 3.1: Sistemas de coordenadas

En la figura 3.1 se puede ver un ejemplo de cómo pueden estar posicionados los sistemas de referencia así como el plano imagen. El plano imagen es un plano 2D donde los puntos serán reproyectados para la formación de la imagen.

Los puntos tridimensionales reconstruidos en tiempo real deben estar referenciados a un mismo sistema de coordenadas, es decir, el sistema de coordenadas debe ser fijo. Por lo tanto, se deberá realizar una conversión entre los distintos sistemas de referencia y el plano imagen.

Para pasar un punto referenciado al sistema de referencia del mundo $\mathcal{R}_w$ llamado p_{jw} al sistema de referencia de la cámara $\mathcal{R}_c$ llamado p_jc se ha de multiplicar dicho punto por la pose P de la cámara:

$$p_{jc} = P * \begin{pmatrix} p_{jw} \\ 1 \end{pmatrix} \tag{3.1}$$

Siendo P una matriz que contiene la matriz de rotación R y el vector de traslación T:

$$P = \left(\begin{array}{ccc|c} r_{11} & r_{21} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{array} \right) \begin{array}{l} \\ \\ \underbrace{\qquad\qquad}_{R} \end{array} \quad \underbrace{\qquad}_{T} \tag{3.2}$$

3.2.1 Modelo de cámara pin-hole sin componente de distorsión

Para obtener la proyección del punto en el plano imagen se hace uso del llamado modelo de cámara pin-hole.

$$m_j = K * P_{jc} \tag{3.3}$$

Siendo m_j el punto en coordenadas homogéneas (m_{xj}, m_{yj}, m_{zj}) y K la matriz de 3x3 de parámetros intrínsecos.

$$K = \begin{pmatrix} f_u & s & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.4}$$

La matriz K se llama matriz de parámetros intrínsecos ya que está compuesta de magnitudes intrínsecas de la cámara: punto focal vertical y horizontal (f_u, f_v), punto principal (u_0, v_0) y factor de corrección w. Por su parte la matriz $(R|T)$ de la ecuación 3.2 se conoce como la matriz de parámetros extrínsecos, donde R es la matriz de rotación y T el vector de traslación de la cámara. Hay que decir que, debido a la calidad de los sensores digitales, en muy pocas ocasiones se tienen en cuenta los once parámetros; $(s = 0)$ y $(f_u = f_v)$. También, si la imagen no ha sido recortada, se puede considerar que el punto principal es el centro de la imagen. Por lo tanto, la matriz T quedaría compuesta por siete parámetros: la distancia focal f y las matrices de rotación R y traslación T.

Por último, la proyección del punto en el plano imagen se obtiene según lo siguiente:

$$\begin{aligned} u_j &= \frac{m_{xj}}{m_{zj}} \\ v_j &= \frac{m_{yj}}{m_{zj}} \end{aligned} \tag{3.5}$$

3.2.2 Modelo de cámara pin-hole con componente de distorsión

En las cámaras existen componentes de distorsión que pueden desviar el modelo pin-hole descrito anteriormente. En este subapartado se va a hacer una introducción a un modelo pin-hole teniendo en cuenta una componente de distorsión radial.

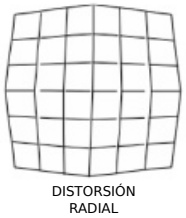

Figura 3.2: Distorsión radial

La distorsión radial es un caso de distorsión simétrica en la cual la magnificación de la imagen disminuye a medida que aumenta la distancia desde el eje central del objetivo. Las lentes grandes angulares, particularmente las lentes ojos de pez utilizan este tipo de distorsión como forma de mapear un plano objeto infinitamente ancho en un área de imagen finita. Esta distorsión hace que se desvíe la proyección del modelo pin-hole del apartado anterior. Las ecuaciones para calcular las coordenadas del punto proyectado en el plano imagen son ahora las siguientes:

$$\begin{aligned} u_j &= u_0 + f_u * \frac{r'}{r} * (\frac{x_{jc}}{z_{jc}}) \\ v_j &= v_0 + f_v * \frac{r'}{r} * (\frac{y_{jc}}{z_{jc}}) \end{aligned} \tag{3.6}$$

Las ecuaciones anteriores pueden considerarse como un modelo pin-hole con un componente de distorsión radial. Este componente de distorsión fue definido por Devernay [30]. Los parámetros de la cámara en este modelo consisten en la distancia focal (f_u, f_v), punto principal (u_0, v_0) y factor de corrección (s).

El componente de distorsión puede ser definido como:

$$r = \sqrt{\frac{x^2 + y^2}{z^2}} \tag{3.7}$$

$$r' = \frac{1}{w} \arctan(2r \tan(\frac{w}{2})) \tag{3.8}$$

Con las ecuaciones anteriores se puede realizar la transformación de los puntos entre sistemas de referencia y el plano imagen teniendo en cuenta la distorsión radial, por lo que la imagen queda totalmente modelada.

3.3 Modelado del problema

En este apartado se va a modelar las incógnitas presentes para la resolución del problema al que se enfrenta el sistema propuesto en este Trabajo Fin de Máster (TFM).

Se parte de una serie de fotogramas $\mathcal{K} = \mathcal{K}_1, \mathcal{K}_2, ..., \mathcal{K}_n$ que va aumentando con el tiempo t. Con un conjunto de $\mathcal{K} > 1$ se ha de obtener la pose $\mathcal{P}$ en el instante t para el frame $\mathcal{K}$. Dicha $\mathcal{P}$ se compone de una matriz de rotación y otra de traslación como se detalló en el apartado 3.2.

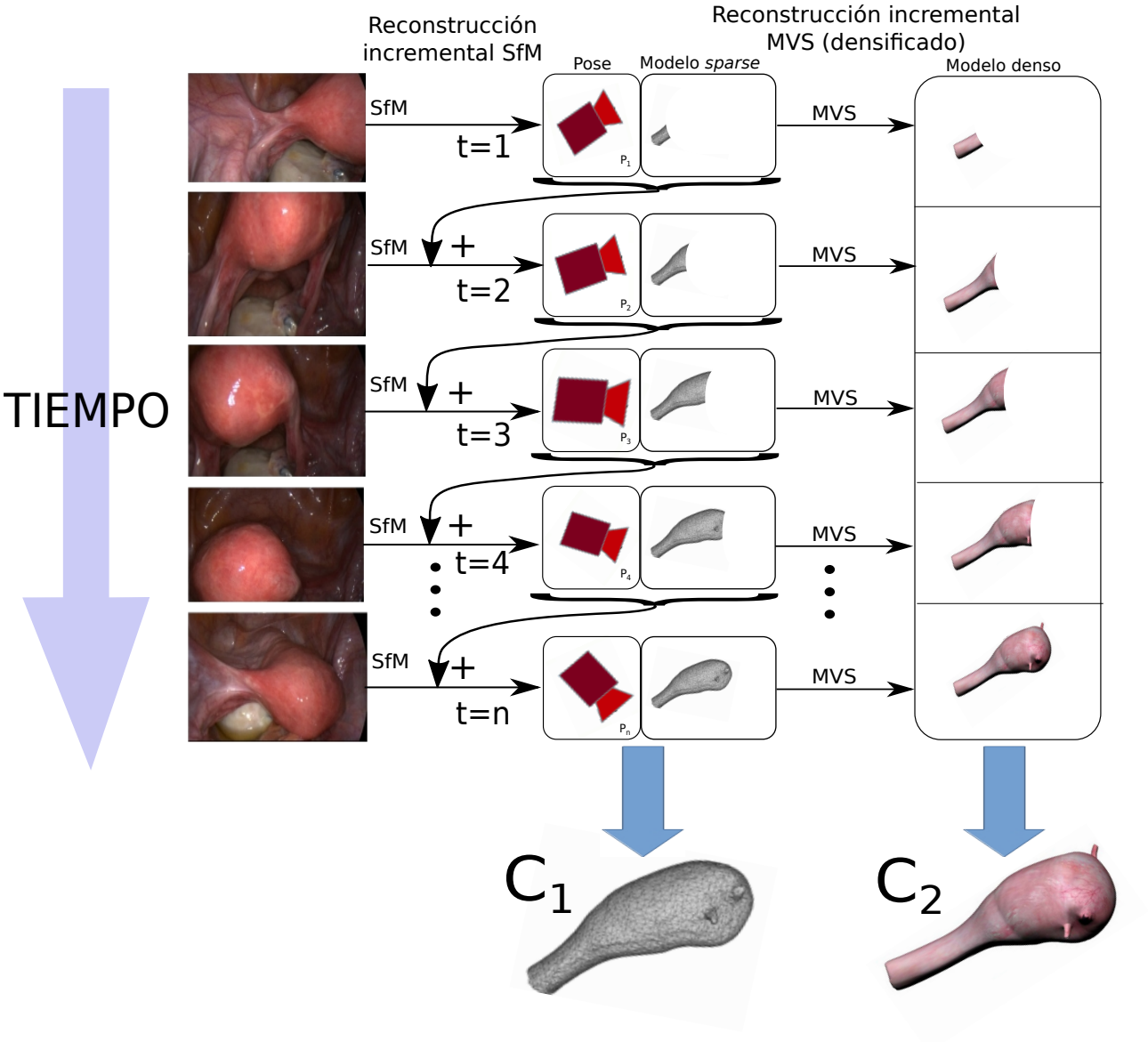

Figura 3.3: Modelado del sistema.

El sistema recibe como entrada fotogramas provenientes de la cámara laparoscópica. Se realiza en primer lugar procesado de Structure-from-Motion (SfM) incremental en tiempo real con dichos frames. Ese procesado es una reconstrucción iterativa o, lo que es lo mismo, SfM además de utilizar los fotogramas generados por la cámara, también tiene en cuenta el resultado del frame anterior, como se puede ver en la Figura 3.3.

Este primer procesado de SfM da como resultado la pose de la cámara que consistirá en la matriz de parámetros extrínsecos o, lo que es lo mismo, una matriz que engloba la matriz de rotación y el vector

de traslación de la cámara para ese frame. Además de la pose, SfM también da como salida un conjunto de puntos denotado por $C_1 = \{C_{1_1}, C_{1_2}.C_{1_3}, ..., C1_n\}$. Se denota C_1 al conjunto de puntos total al que se le van añadiendo subconjuntos $\{C_{1_1}, C_{1_2}.C_{1_3}, ..., C1_n\}$ a medida que pasa el tiempo, incrementando la densidad de puntos del modelo global C_1. Este conjunto de puntos global es una nube de puntos *sparse* resultante del proceso de SfM. A medida que pasa el tiempo los puntos generados se van añadiendo a un modelo global.

La ultima fase del sistema es densificar el modelo obtenido por SfM. Para ello se realiza un segundo procesado en tiempo real que hace uso de algoritmos de reconstrucción Multi-view Stereo (MVS). Esta parte del sistema recibe como entrada los fotogramas generados por la cámara así como la pose de la misma por cada fotograma. MVS da como resultado un conjunto de puntos en color $C_2 = \{C_{2_1}, C_{2_2}.C_{2_3}, ..., C_{2_n}\}$. Al igual que en el proceso anterior, C_2 es un conjunto de puntos global correspondiente al modelo de reconstrucción denso. A este modelo se le añaden nuevos conjuntos de puntos $\{C_{2_1}, C_{2_2}.C_{2_3}, ..., C_{2_n}\}$ a medida que pasa el tiempo. C_2 es una nube de puntos que constituye una reconstrucción densa del objeto, añadiendo mayor realismo al modelo. El resultado final de este procesado ofrece un valor añadido a la reconstrucción del objeto por el realismo y precisión del modelo 3D resultante.

Capítulo 4

Desarrollo/Solución

Let us read, and let us dance; these two amusements
will never do any harm to the world.

Voltaire

4.1 Introducción

En esta sección se detallará el desarrollo de la solución propuesta así como las herramientas usadas para el desarrollo de la misma mostrando el resultado obtenido.

4.2 Reconstrucción en tiempo real de un objeto con una cámara monocular

Una vez que se han modelado todos los parámetros del problema y se ha definido una solución para el mismo, se puede proceder con el desarrollo de la herramienta de reconstrucción incremental 3D en tiempo real.

Como se detalló en el apartado 3.3, el sistema presenta dos elementos principales para la reconstrucción del objeto:

- Reconstrucción incremental en tiempo real mediante técnicas Structure-from-Motion (SfM)
- Densificación incremental en tiempo real mediante técnicas Multi-view Stereo (MVS)

Se encuentran en el estado de la técnica herramientas desarrolladas de código abierto que pueden ser adecuadas para la aplicación que aborda este Trabajo Fin de Máster (TFM).

Se decide implementar primeramente un software llamado Parallel Tracking and Mapping (PTAM) desarrollado por la Universidad de Oxford [3] para llevar a cabo el SfM incremental en tiempo real. La elección de PTAM se debe a diversos motivos como son: su efectividad, funcionamiento en tiempo real y código abierto, entre otros.

Por otro lado, para la densificación del modelo se recurre a técnicas MVS como se comentó anteriormente. Más en concreto, se decide usar una herramienta llamada Patch-based Multi-view Stereo (PMVS) desarrollada por Yasutaka Furukawa y Jean Ponce [4]

Para llevar a cabo este desarrollo, se se ha procedido a aislar y testear cada módulo por separado para comprobar su funcionalidad y, una vez que este todo verificado, realizar la integración de los mismos.

En primer lugar se realiza la instalación del software PTAM. El código fuente está disponible al ser de código abierto [31]. Sin embargo, el usuario debe instalar las dependencias necesarias así como componer el archivo *CMakeLists.txt* para que sea posible el funcionamiento del software.

Cuando se realiza la compilación, se generan dos ejecutables; uno correspondiente a la calibración de la cámara y otro a la herramienta PTAM.

PTAM y PMVS son dos softwares independientes que se usan como base para el desarrollo del sistema de este TFM. Sin embargo, se ha de hacer uso de otras herramientas sin las cuales no seria posible el desarrollo e implementación del sistema de este TFM, que son las siguientes:

- **QtCreator:** Entorno de desarrollo con la que puede crear una interfaz gráfica de usuario.
- **VTK:** Visualizador para representación de las nubes de puntos. Este visualizador permite la integración con QtCreator.
- **PCL:** Librería de representación de nubes de puntos 3D.
- **PTAM:** Herramienta que implementa SfM incremental en tiempo real.
- **PMVS:** Herramienta que implementa un densificado incremental en tiempo real usando técnicas MVS.

Dichas herramientas se detallarán en las siguientes subsecciones.

4.2.1 Herramientas necesarias

4.2.1.1 Qt Creator

Qt Creator es el entorno de desarrollo en el que se ha desarrollado el programa principal y la interfaz de usuario de la herramienta de reconstrucción 3D en la que se va a realizar la implementación de todos los componentes necesarios.

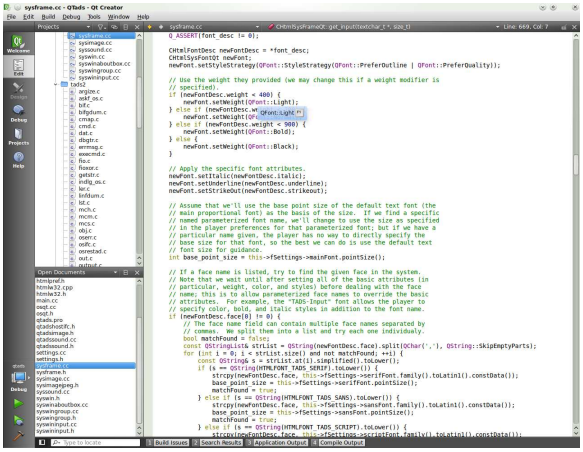

Figura 4.1: QtCreator

Qt es una biblioteca de software que permite crear interfaces gráficas de usuario. Aplicaciones como Google Earth, Skype, Adobe Photoshop y Virtual Box entre muchas otras hacen uso de ésta biblioteca de software. Sus características principales son las siguientes:

- Posee un avanzado editor de código C++.
- Soporta los lenguajes: .NET Languages (Mono), Python: PyQt y PySide, Ada, Pascal, Perl, PHP y Ruby.
- Presenta una GUI integrada y diseñador de formularios.
- Depurador visual.
- Herramienta para proyectos y administración.
- Control de versiones Git integrado.

Qt Creator es distribuido bajo tres tipos de licencias: Qt Commercial Developer License, Qt GNU LGPL v. 2.1, Qt GNU GPL v. 3.0 y está disponible para las plataformas: Linux, Mac OSX; Windows, Windows CE, Symbian y Maemo.

4.2.1.2 Point Cloud Library

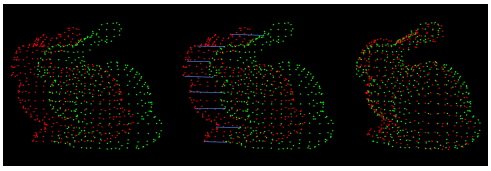

Figura 4.2: Point Cloud Library

PCL es un proyecto que inició Willow Garage en marzo de 2001 para el procesamiento de nubes de puntos en 3D. El proyecto ha continuado en todo el mundo con la colaboración de científicos, empresas y universidades gracias a que se trata de un código abierto y gratuito. PCL es multiplataforma y ha sido compilado con éxito y desplegado en Linux, MacOS, Windows, Android e iOS. Las librerías que componen PCL están preparadas para trabajar con nubes de puntos. Una nube de puntos es una estructura de datos en 4D en la que se encuentran cuatro elementos principales: las coordenadas xyz del punto y el color en formato RGB. El formato principal de los archivos que contienen las nubes de puntos tiene la extensión .PCD; un formato original de PCL basado en formatos como .PLY, .STL, .OBJ o .X3D. Cada archivo .PCD incluye una cabecera con la siguiente información:

- Versión
- Nombre de cada dimensión (x, y, z, rgb, $normal_x$, $normal_y$, etc)
- Tamaño (4 bytes)
- Tipo (F, float)
- Número de elementos que ocupa cada dimensión
- Ancho
- Alto (Si este valor es 1, los puntos están desordenados y el ancho es el total de la nube)
- Orientación en la que se han adquirido los datos
- Número total de puntos
- Tipo de código usado en el archivo (ASCII o binario)

Ejemplo:

```
# .PCD v.7 - Point Cloud Data file format VERSION .7
FIELDS x y z rgb
SIZE 4 4 4 4
TYPE F F F F
COUNT 1 1 1 1
WIDTH 213
HEIGHT 1
VIEWPOINT 0 0 0 1 0 0 0 POINTS 213
DATA ascii
0.93773 0.33763 0 4.2108e+06
0.90805 0.35641 0 4.2108e+06
0.81915 0.32 0 4.2108e+06
0.97192 0.278 0 4.2108e+06
0.944 0.29474 0 4.2108e+06
0.98111 0.24247 0 4.2108e+06
0.93655 0.26143 0 4.2108e+06
0.91631 0.27442 0 4.2108e+06
0.81921 0.29315 0 4.2108e+06
0.90701 0.24109 0 4.2108e+06
.
.
.
```

El lenguaje de programación usado en las librerías PCL es el lenguaje orientado a objetos C++. Dicha programación fue desarrollada esencialmente por las limitaciones que tiene la programación estructurada C la cual se basa en funciones con un propósito claramente definido: la información se pasa entre funciones utilizando parámetros y las funciones pueden tener datos locales a los cuales no se pueden acceder fuera del ámbito del procedimiento.

4.2.1.3 Kit de Herramientas de Visualización (VTK)

El Kit de Herramientas de Visualización (VTK) es un software libre para la realización de gráficos 3D por computadora, procesamiento de imagen y visualización. VTK consiste en una biblioteca de clases de C++ y ofrece compatibilidad con varias interfaces interpretadas tales como Tcl/Tk, Java, y Python.

VTK es un conjunto de herramientas creadas por Kitware. Este equipo de trabajo sigue ampliando las herramientas de visualización y también ofreciendo también soporte técnico para VTK. VTK soporta una amplia variedad de algoritmos de visualización como: herramientas para texturas, herramientas vectoriales y métodos volumétricos así como técnicas de modelado como: modelado implícito, reducción de polígonos y suavizado de malla (mesh smoothing) entre otras. VTK presenta una forma muy intuitiva de visualización de la información. Cuenta con un conjunto de widgets de interacción 3D, soporta el procesamiento en paralelo y se integra con diversas bases de datos de herramientas GUI como Qt y Tk. Esto último será de gran utilidad para este trabajo ya que se realizará la integración con QtCreator. (ver Figura 4.3).

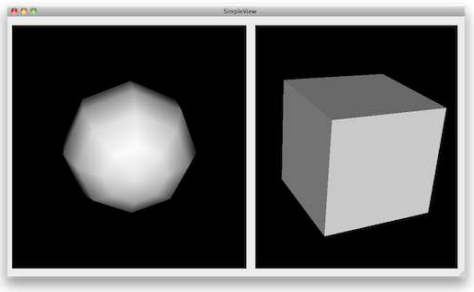

Figura 4.3: Integración de VTK con QtCreator

VTK es multiplataforma y puede ser ejecutado en diversas plataformas como Linux, Windows y Mac. VTK también incluye soporte auxiliar de widgets de interacción 3D, anotación bi y tridimensional y computación paralela. VTK es implementado como un conjunto de herramientas de C++. El sistema también soporta la integración de otros lenguajes tales como Python, Java y Tcl, para que también se puedan escribir aplicaciones VTK utilizando estos lenguajes de programación interpretados.

VTK es mundialmente utilizado en aplicaciones comerciales, investigación y desarrollo y es la base de muchas aplicaciones de visualización avanzadas tales como: ParaView, VisIt, VisTrails, 3DSlicer MayaVi2 y OsiriX.3. Por último cabe destacar que VTK es un kit de herramientas de código abierto bajo Licencia BSD.

4.2.1.4 PTAM

Para la reconstrucción incremental del objeto (Incremental Structure from Motion (iSfM)) se emplea la herramienta PTAM. En esta sección se va a introducir en lineas generales este software desarrollado por la Universidad de Oxford [3] que, entre otras, presenta las siguientes características:

- El seguimiento y el mapeado se realiza de forma separada, en dos procesos paralelos diferentes.
- El mapeado se basa en keyframes el cual es refinado usando técnicas de Bundle Adjustment 2.3.1.1.
- El mapa es inicializado mediante un par de imágenes estéreo usando el algoritmo de 5 puntos [32].
- Los nuevos puntos son inicializados mediante una búsqueda de geometría epipolar.
- Una gran cantidad (miles) de puntos son mapeados.

PTAM fue originalmente diseñado para cámaras monoculares, el cual es el caso. En lugar de actualizar un mapa 3D en cada frame como hace EKF-SLAM, en PTAM el mapeado y el seguimiento se hacen en dos procesos paralelos e independientes. Cabe destacar que al proceso de mapeado se le asigna mucha menos prioridad que al proceso de seguimiento. Como se ha dicho anteriormente, los dos procesos se ejecutan en paralelo y se comunican entre sí mediante el mapa 3D: el proceso de seguimiento estima la pose de la cámara basándose en el mapa 3D generado por el proceso de mapeado. Por su parte, el proceso de mapeado recibe nuevos *keyframes* del proceso de seguimiento y actualiza correspondientemente el mapa 3D.

Existen tres elementos fundamentales en el proceso de seguimiento:

- Primero, se usa un modelo de descomposición de la velocidad del movimiento para predecir la pose actual de la cámara.
- Segundo, los puntos 3D son reproyectados en el frame actual así como una búsqueda con rango fijo sobre el resto de puntos reproyectados.
- Tercero, los puntos 3D identificados así como sus medidas estéreo son utilizados para la estimación de la pose.

Por su parte, el proceso de mapeado también se compone de tres elementos fundamentales:

- En primer lugar se requiere de la acción del usuario para la inicialización del mapa.
- En segundo lugar, cuando se explore un nuevo área, el frame es almacenado en el mapa para su posterior reconstrucción 3D.

- Por último, se realiza un refinado del mapa mediante un proceso local y global de *Bundle Adjustment* lo cual optimiza simultáneamente los puntos 3D y las poses de la cámara.

El funcionamiento del mapeado que realiza PTAM se puede resumir en el siguiente diagrama:

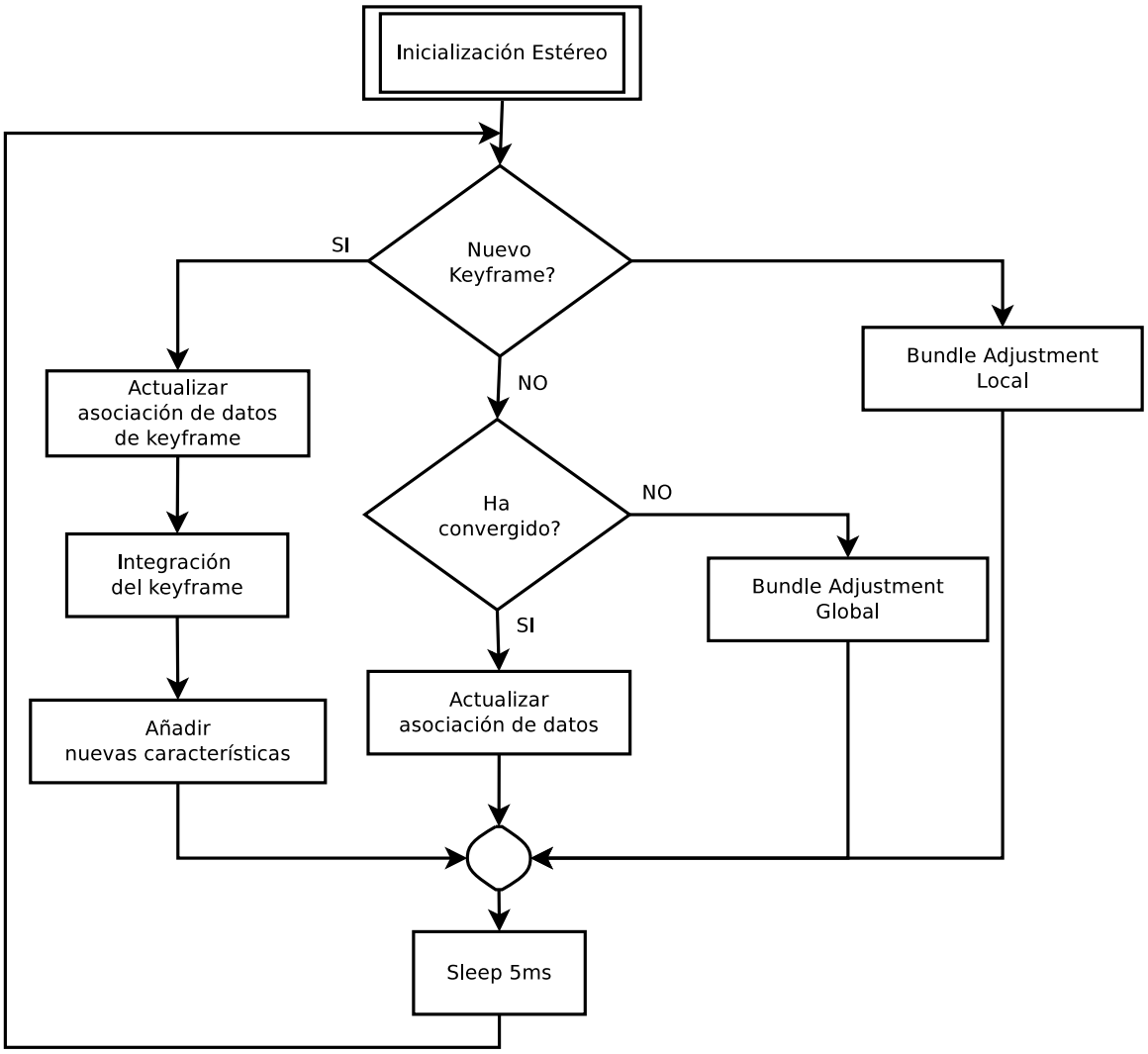

Figura 4.4: Diagrama de flujo del proceso de mapeado. Este hilo está dentro de un bucle infinito, recibiendo las nuevas imágenes procedentes del hilo de seguimiento. [3]

La mayor ventaja de PTAM es la capacidad de reconstruir un gran número de puntos 3D en tiempo real.

4.2.1.5 PMVS

PMVS es una herramienta que implementa un algoritmo MVS que da como salida un conjunto de parches consistentes en puntos que representan un modelo denso de las superficies visibles de las imágenes.

Este algoritmo está implementado en tres fases fundamentales: matching, expansión y filtrado. PMVS tiene como datos de partida la pose de la cámara por cada frame así como los frames correspondientes. El proceso empieza con un conjunto de puntos sparse. El algoritmo matchea y expande dichos puntos y a continuación realiza un filtrado de los mismos teniendo en cuenta la información contenida en los fotogramas.

La clave de PMVS consiste en aplicar funciones que mejoran la fotoconsistencia [4] del modelo. Esta herramienta no precisa de ninguna inicialización previa ni segmentación de las imágenes. El único requisito, al igual que en los procesos de SfM es que las imágenes estén tomadas con cierta separación.

Como se dijo en el párrafo anterior, PMVS recibe una serie de imágenes y los parámetros de la cámara para cada imagen. Con dicha información el algoritmo realiza una reconstrucción densa 3D de un objeto o escena visible en dichas imágenes. Según el resultado que se quiera obtener se aplican más o menos procesados a la imagen aumentando más o menos dicho tiempo consecuentemente. En el siguiente diagrama se puede ver el procesado típico de este tipo de algoritmos:

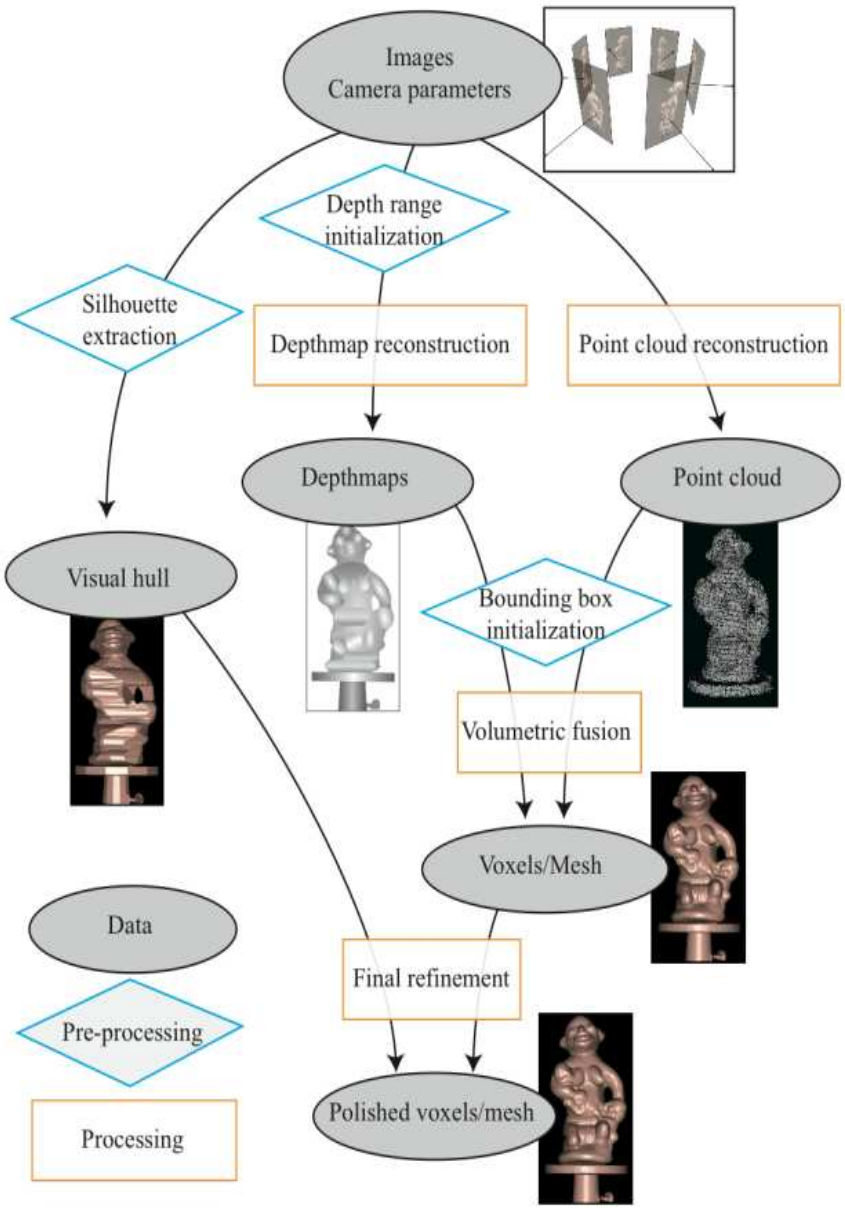

Figura 4.5: Diagrama de flujo del proceso típico de PMVS [4]

En la figura 4.5 se puede ver las diferentes etapas (algoritmos) de los que está compuesto PMVS con su diagrama de flujo típico. Los procesos en gris denotan representaciones de datos. Con el modelo resultante, el usuario puede aplicar los procesados que desee como puede ser una reconstrucción de la superficie mediante mesh, voxels, etc.

Cabe destacar que sólo pueden ser reconstruidas estructuras rígidas ya que el software ignora automáticamente objetos no rígidos como pueden ser peatones en una escena de un edificio o elementos deformables del fondo de la escena en la reconstrucción de un objeto.

4.3 Calibración de la cámara

Lo primero que hay que hacer es realizar una correcta calibración de la cámara. La cámara usada en este TFM es una webcam *Logitech C270* con resolución de hasta 720p.

Para calibrar la cámara se imprime un patrón de cuadrículas de calibración. Una vez hecho esto, se procede a calibrar la cámara usando la herramienta de calibración desarrollada por la Universidad de Oxford [3].

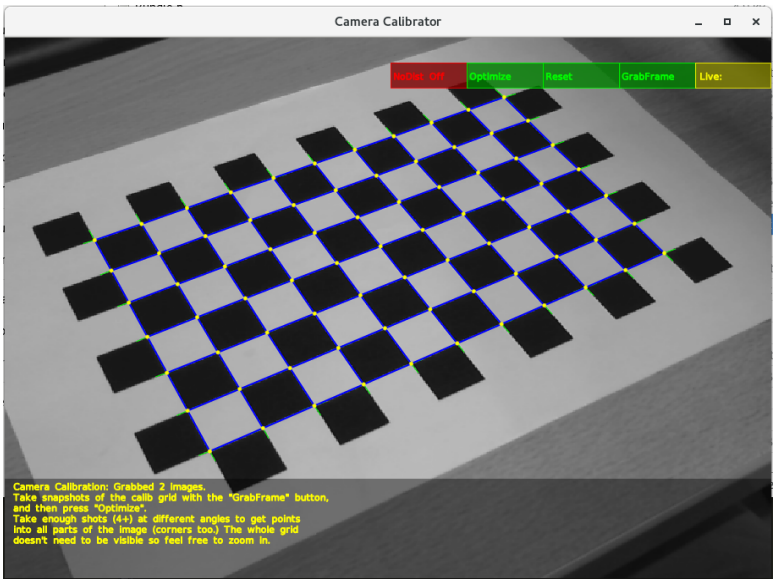

Figura 4.6: Calibración de la cámara

El procedimiento consiste en almacenar una serie de fotogramas válidos del patrón de calibración desde diferentes perspectivas. Con la herramienta en ejecución, cuando se apunta al patrón de calibración el software realiza automáticamente una búsqueda de las esquinas. Una vez las tiene localizadas las relaciona entre sí señalizando los rectángulos detectados con color azul. Esto se pude ver en la Figura 4.6. Una vez que la escena contenga todos los rectángulos azules del patrón, el usuario debe de presionar el botón *GrabFrame* para almacenar ese frame como frame de calibración. Se deberá hacer este procedimiento con diferentes perspectivas visuales.

Una vez se haya capturado varios frames desde diferentes puntos de vista (26 en el caso de este TFM), se debe de presionar el botón *Optimise*. Al pulsar este botón se iniciará un proceso en el cual se calcula iterativamente los parámetros de la cámara. Cuando este proceso finalice, el usuario puede comprobar la precisión del cálculo de los parámetros mediante el valor del error RMS mostrado en pantalla. Se aconseja que sea inferior a 0.3 píxeles. El usuario puede repetir el proceso cuantas veces quiera hasta que salga una calibración precisa de acuerdo al error RMS.

Cuando el error sea suficientemente bajo, el usuario deberá presionar el botón *Save* para almacenar los parámetros de la cámara. Los parámetros se almacenan en el fichero *camera.cfg* y corresponden a los parámetros intrínsecos de la cámara.

4.4 Ejecutando PTAM

Una vez se tiene la cámara calibrada es posible ejecutar el software PTAM.

PTAM leerá el archivo generado por la herramienta de calibración de la cámara, por lo que se debe comprobar que PTAM esté leyendo los parámetros de calibración correctos.

Para una primera evaluación de PTAM se toma como modelo un objeto cualquiera. Dicho objeto debe de presentar ciertas características que puedan ser detectados por los detectores de características incluidos en PTAM.

Para un primer testeo del software, el objeto se mantendrá fijo en la escena y será la cámara la que se mueva entorno a la cámara. Lo primero que se debe de realizar es inicializar el proceso de mapeado como se describió en la sección 4.2.1.4. Para ello se debe de apuntar la cámara al objeto a reconstruir y presionar la barra espaciadora al mismo tiempo que se hace un movimiento de traslación con una ligera rotación. Mientras esto se hace, se pueden ver unas líneas provenientes de las características detectadas que aumentan con el movimiento, como se ve en la figura 4.7.

Cuando se haya efectuado el movimiento de inicialización, el usuario debe de dejar de presionar la barra espaciadora para ver los primeros puntos mapeados sobre el objeto. Cabe destacar que a dichos puntos se les ha aplicado una transformación para su visualización en el plano imagen. Además de los puntos, se muestra rejilla con el plano base principal elegido por PTAM como se muestra en la Figura 4.7.

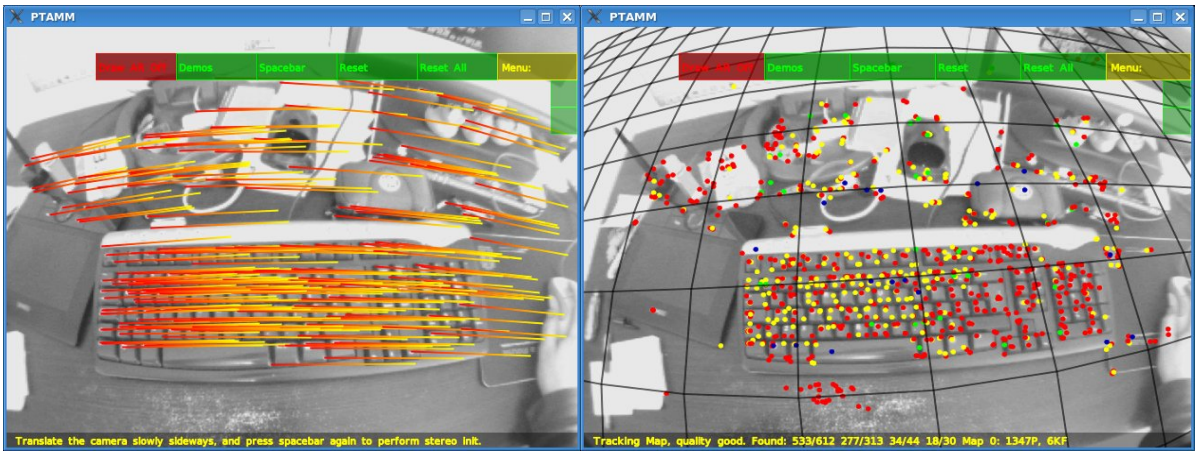

Figura 4.7: Inicialización del mapa por parte del usuario

Con la inicialización aplicada, se inicia el proceso iterativo descrito en 4.2.1.4 es decir, a medida que el usuario realiza el movimiento de la cámara nuevos puntos son mapeados y trackeados realizando el seguimiento de la cámara por cada imagen como se describió en las secciones anteriores.

La herramienta PTAM ofrece varias características que no serán de interés para este TFM. Presenta una opción de una visualización de elementos de Realidad Aumentada (AR), sobre el objeto mapeado. Además de eso, PTAM ofrece una visualización desarrollada con la librería OpenGL de los puntos tridimensionales haciendo click en el botón *View Map*. Los puntos son representados en un espacio tridimensional donde también se muestran las poses de la cámara.

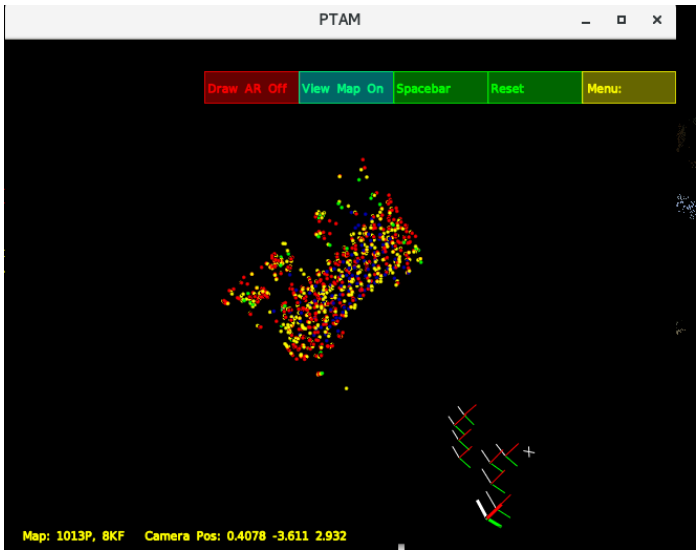

Figura 4.8: Visualización del mapa 3D ofrecido por PTAM

4.5 Ejecutando PMVS

Como se dijo anteriormente, PMVS recibe como entrada la pose de la cámara y los fotogramas para realizar la reconstrucción densa. Originalmente, dichos datos hay que almacenarlos en un directorio con un formato específico para que PMVS realice la lectura de los mismos y genere un archivo de nube de puntos con extensión .ply.

Para realizar una prueba inicial se escogen unos fotogramas de un edificio y se generan los parámetros de la cámara mediante un software externo de SfM. Dichos fotogramas provienen de la galería de imágenes propia de PMVS.

PMVS también recibe como entrada un fichero de parámetros donde se puede configurar los parámetros de la tabla:

Tabla 4.1: Resultados de la correlación cruzada.

Nombre	Valor por defecto	Explicación
timages	-	Este parámetro se refiere al número de imágenes a combinar. Dicho parámetro puede ser una enumeración de imágenes (1 2 3 4 5), o un rango de imágenes (-1 0 6)
oimages	-	Parámetro auxiliar para mejorar la reconstrucción 3D. Establecido en 0
level	1	El software internamente crea una pirámide de imágenes. Este parámetro establece el nivel en que se quiere aplicar de la pirámide para la cómputo. En definitiva, si se establece en 0 se utiliza la resolución completa de la imagen
csize	2	Este parámetro se refiere al tamaño de la celda y controla la densidad de la reconstrucción. PMVS trata de reconstruir al menos un parche en cada región de tamaño csize x csize en todas las imágenes especificadas por timages. Este parámetro hace más o menos sparse la reconstrucción

Continúa en la página siguiente

Tabla 4.1 – continúa en la página anterior

Nombre	Valor por defecto	Explicación
threshold	0.7	Umbral en el que se acepta el nivel de fotoconsistencia de la reconstrucción
wsize	7	Factor relacionado con el tamaño de los píxeles para su procesado y posterior evaluación de fotoconsistencia
minImageNum	3	Cada punto 3D debe tener correspondencia en al menos minImageNum frames
CPU	4	Número de hilos en paralelo que pueden ser ejecutados en la CPU del ordenador
useVisData	0	Este parámetro se usa si se tiene datos adicionales que relacionan una imagen con las otras. Softwares como Visual SfM pueden dar como salida que frames se corresponden con otros frames para poder realizar el procesado con mayor velocidad de cómputo
sequence	-1	En algunos casos las imágenes pueden provenir de una secuencia estando dichos frames numerados. Este parámetro es para establecer la secuencia que se quiere procesar
quad	2.5	Este es un parámetro que modifica la efectividad del filtrado del modelo. Este filtrado se realiza teniendo en cuenta la vecindad de píxeles. Con este parámetro se modifica dicha vecindad de píxeles
maxAngle	10	El modelo 3D no es reconstruido si el ángulo entre las cámaras supera este umbral. Se tiene como medida de seguridad para asegurar una correspondencia entre las imágenes

En este primer test, se dejan los parámetros por defecto para la reconstrucción 3D usando cerca de treinta imágenes de un edificio. El resultado de la reconstrucción densa se muestra en la figura 4.9.

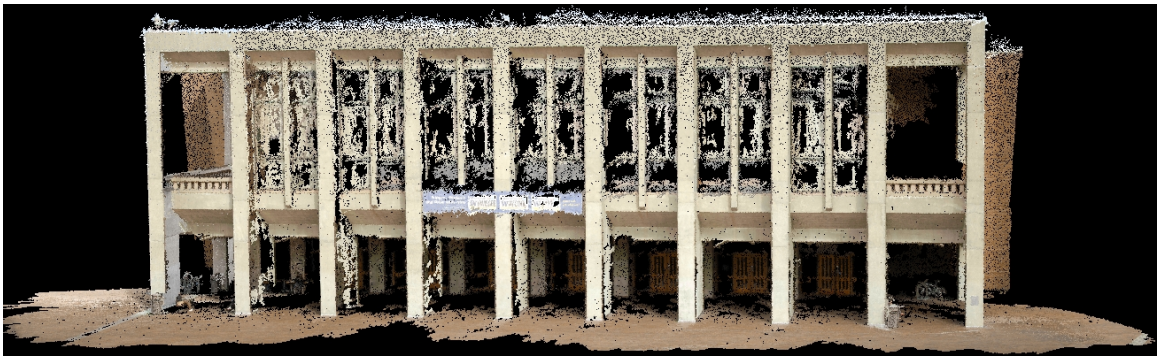

Figura 4.9: Reconstrucción de un edificio con el software PMVS

Se puede comprobar el realismo del objeto reconstruido comparándolo con las imágenes originales. En este proceso es crítico la correcta obtención de los parámetros de la cámara lo cual engloba el proceso de calibración de la misma, ya que se tienen en cuenta tanto los parámetros intrínsecos como los extrínsecos de la misma.

4.6 Programa principal e Interfaz gráfica de usuario

Una vez que se han probado los componentes del programa por separado y se ha verificado su correcto funcionamiento. Lo siguiente es realizar la integración de las herramientas modificado dichos códigos para un funcionamiento continuado en tiempo real.

Se decide utilizar la librería Point Cloud Library (PCL) para la representación gráfica de los modelos generados. PCL requiere un visualizador donde mostrar los puntos. Se decide utilizar el visualizador VTK el cual se puede integrar con el entorno de desarrollo QtCreator.

Por lo tanto, lo primero que se realizó fue la inclusión del visualizador VTK en el entorno de desarrollo Qt. VTK ofrece varios widgets para su inclusión en la interfar gráfica. Sin embargo estos widgets no aparecen en el entorno de desarrollo por lo que directamente se programó en formato XML en el código de la interfaz gráfica.

```
<?xml version="1.0" encoding="UTF-8"?>
  <customwidgets>
  <customwidget>
   <class>QVTKWidget</class>
   <extends>QWidget</extends>
   <header location="global">QVTKWidget.h</header>
  </customwidget>
 </customwidgets>
</xml>
```

En dicho código XML se especifica donde se encuentra el archivo de cabecera que implementa los widgets del visualizador VTK.

Para testear el visualizador se abrió un par de modelos tridimensionales y se comprobó la fluidez y el funcionamiento de la integración de ambas herramientas (ver Figura 4.3).

Posteriormente, se desarrolló una pequeña interfaz de usuario donde se representaban una nube aleatoria de puntos de la librería PCL. Los puntos de la nube se representaron sobre el visualizador VTK. Para familiarizarse con PCL se incluyó en esta interfaz de usuario algunos controles para cambiar el color de los puntos y el tamaño de los mismos entre otros.

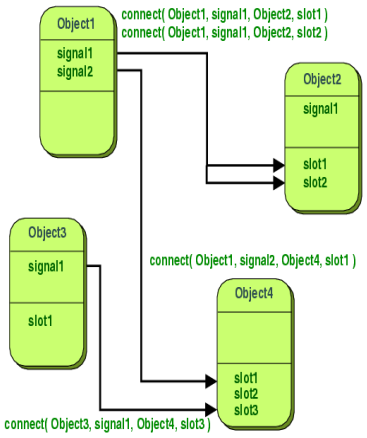

Figura 4.10: Signals and slots

El siguiente paso fue el de la inclusión de PTAM en el programa. Este paso no es una tarea sencilla. En primer lugar, se tuvo la dificultad de que el proyecto desarrollado en Qt estaba en formato de proyecto de Qt con extensión *.pro mientras que PTAM es un tipo de proyecto de *CMakeLists.txt*. Por lo tanto, se tuvo que realizar una migración del proyecto de Qt con extensión *.pro a un tipo de proyecto con *CMakeLists.txt* adaptando la inclusión de todas las librerías de PCL y VTK.

Para la inclusión de PTAM en el programa se tuvo que lanzar un hilo paralelo para asegurar un funcionamiento fluido de la interfaz de usuario. Como se detalló en la sección 4.2.1.4 PTAM lanza dos hilos paralelos uno para el tracking y otro para el mapping.

Con este hilo implementado se comprobó el funcionamiento fluido de PTAM corriendo en paralelo con la interfaz de usuario.

El siguiente reto fue el de comunicar el hilo encargado de PTAM con el hilo principal. Para ello se usó una herramienta que ofrece Qt llamada *Signals and slots* (ver Figura 4.10). Este mecanismo esta diseñado para comunicar objetos en Qt y es una de las características principales de Qt.

La teoría de *Signals and slots* recae principalmente en que una señal es emitida cuando un evento particular ocurre. Los widgets de Qt presentan muchas señales predefinidas. Sin embargo se pueden diseñar señales específicas para una comunicación en particular. Por lo tanto, cuando PTAM finaliza la reconstrucción por cada imagen, emite una señal enviando los datos al hilo principal mediante este mecanismo de Qt (ver Figura 4.11).

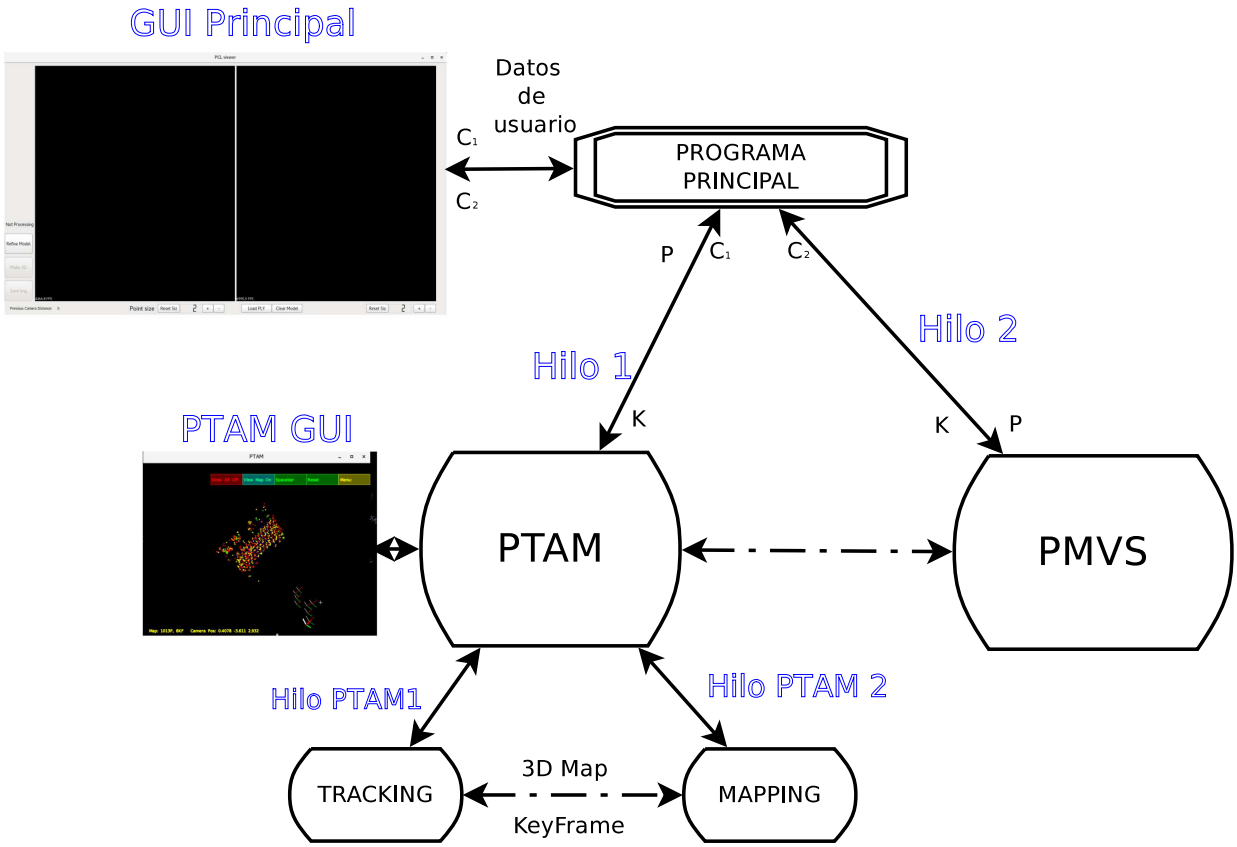

Figura 4.11: Diagrama de flujo del software desarrollado

Una vez que se tienen los puntos y las poses procedentes de PTAM en el programa principal, se ha de traducir los conjuntos de puntos tridimensionales al formato de la librería PCL y representarlos en el visualizador VTK.

Por lo tanto, en este punto la visualización del modo mapa de la interfaz de usuario de PTAM debe corresponderse perfectamente con la visualización de los puntos en la interfaz principal desarrollada.

Una vez que se tienen en el programa principal las poses P y el conjunto de puntos C_1 de PTAM se procede a integrar la segunda parte del programa: PMVS.

Para la integración de la herramienta se realizó la inclusión de los códigos en el programa principal pasando a formar parte del proyecto, así como sus dependencias en el archivo *CMakeLists.txt*. Al igual que con PTAM, PMVS debe correr en un hilo en paralelo (Hilo 2) para no crear conflicto con el hilo principal, el cual se encarga de la interfaz gráfica y, consecuentemente, de su fluidez.

Como se dijo en 4.2.1.5 PMVS está programado originalmente de tal forma que busca en un determinado directorio una serie de ficheros con un preciso formato. Para mejorar el funcionamiento de PMVS en tiempo real, se modificaron varios métodos de tal forma que dicha información se pasa como argumento. La información es transmitida usando nuevamente la herramienta de comunicación de Qt *Signals and slots*. PMVS recibe las pose de la cámara P y los fotogramas capturados por la cámara denotados en el diagrama 4.11 con la letra K.

Además de eso, para beneficiar el funcionamiento de este software en tiempo real, se ajustaron los parámetros PMVS de tal forma que se genera una nueva nube de puntos cada 3 frames siempre y cuando dichos frames estén separados una distancia euclídea entre cámaras establecida en 150mm. Dichos frames son seleccionados automáticamente por el programa principal teniendo en cuenta la pose recibida por el hilo de PTAM.

La nube parcial de puntos del modelo 3D densificado obtenida de PMVS es recibida por el programa principal el cual se encarga de añadir dicha nube parcial a la nube global de puntos C_2.

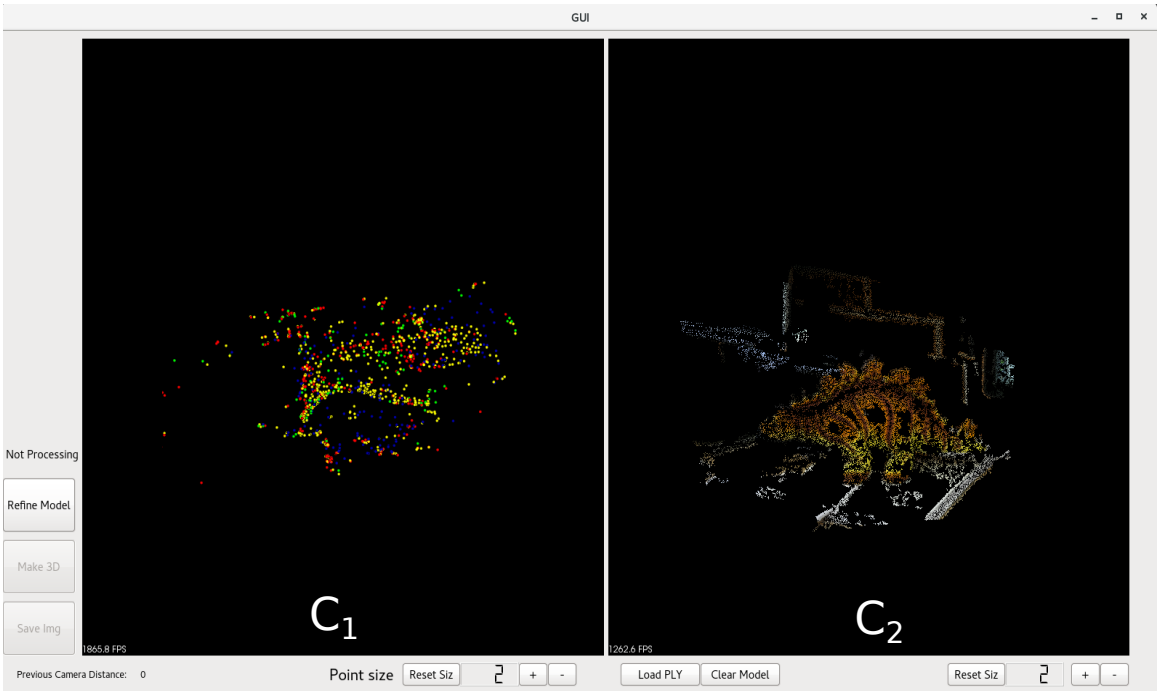

Figura 4.12: Aspecto de la interfaz gráfica de usuario principal

El resultado de todo el proceso es una representación de dos nubes de puntos en la interfaz gráfica principal las cuales se van incrementando a medida que pasa el tiempo y el usuario realiza el movimiento de la cámara alrededor del objeto.

Como se puede ver en la Figura 4.12 la interfaz gráfica desarrollada se compone principalmente de dos ventanas de visualización. En la primera se representa la nube de puntos C_1 obtenida de PTAM mientras que en la segunda ventana se muestra la nube de puntos correspondiente al modelo denso C_2 proveniente PMVS.

Además de las ventanas de visualización de las nubes de puntos, la interfaz gráfica de usuario presenta la posibilidad de cargar y almacenar dichas nubes de puntos en archivos con formato *.ply. También se puede ver la distancia absoluta entre la cámara anterior y la actual. El usuario también puede ajustar parámetros de visualización como el tamaño de los puntos, refinado del modelo e incluso almacenar screenshots de lo que el usuario está viendo en ese preciso instante.

Capítulo 5

Resultados

I've always believed that if you put in the work,
the results will come.

Michael Jordan

5.1 Introducción

En este capítulo se detallarán de forma gráfica los resultados obtenidos con la herramienta propuesta y desarrollada según se indica en los apartados anteriores.

5.2 Resultados

En esta sección se muestran los resultados obtenidos. Cabe destacar que, al no tener un modelo de referencia o ground truth, no se pueden dar resultados cuantitativos de los resultados obtenidos, por lo que todos los resultados podrán ser evaluados únicamente cualitativamente.

Para la realización de este Trabajo Fin de Máster (TFM) no ha sido posible obtener imágenes médicas reales que implementasen el movimiento de inicialización estéreo necesaria para el correcto funcionamiento del sistema mencionado en capítulos anteriores. Dicha inicialización es un componente fundamental en el funcionamiento de la aplicación ya que las nubes de puntos obtenidas se referencian de acuerdo a ese primer movimiento.

Por lo que, al no tener secuencias las de video adecuadas, el sistema se ha testeado usando diferentes objetos que, aunque no están relacionados con las operaciones de laparoscopia, son suficientes para testear y evaluar el funcionamiento del sistema. Dichos objetos presentan diversas texturas y componentes las cuales sirven para testear la robustez del sistema en cuanto a reconstrucción y mapeado se refiere. Además de la textura, se han seleccionado objetos con diferentes formas y colores para comprobar su funcionamiento.

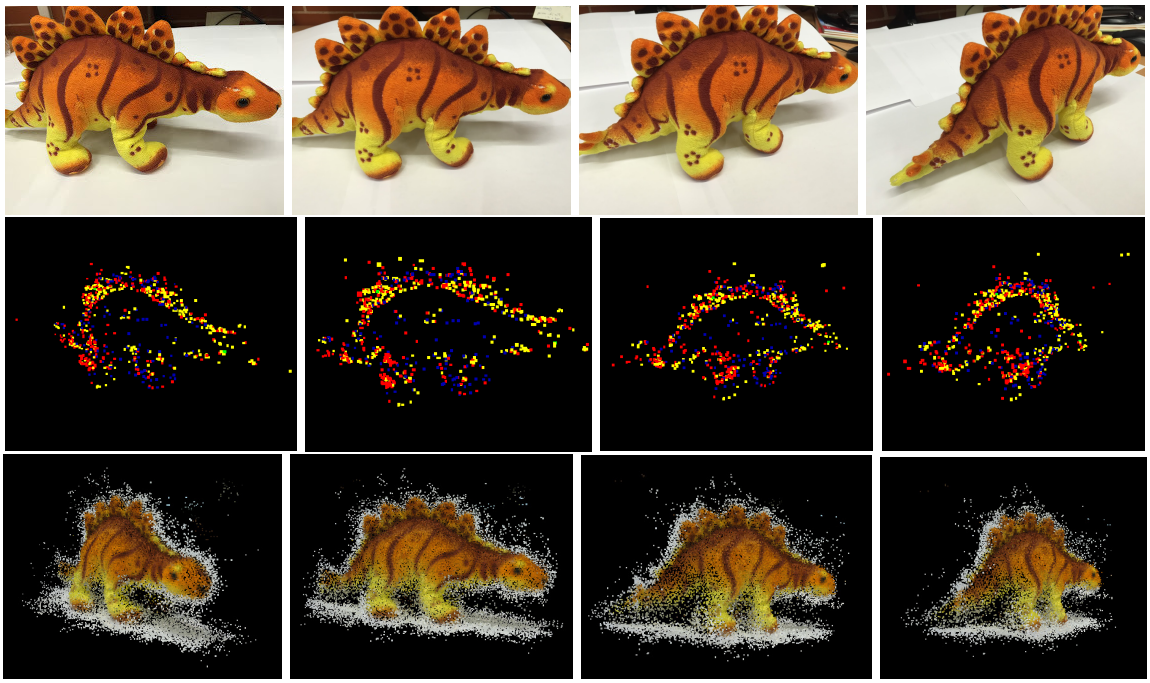

Figura 5.1: Objeto 1: Arriba: Imágenes RGB. Medio: Reconstrucción sparse. Abajo: Reconstrucción densa

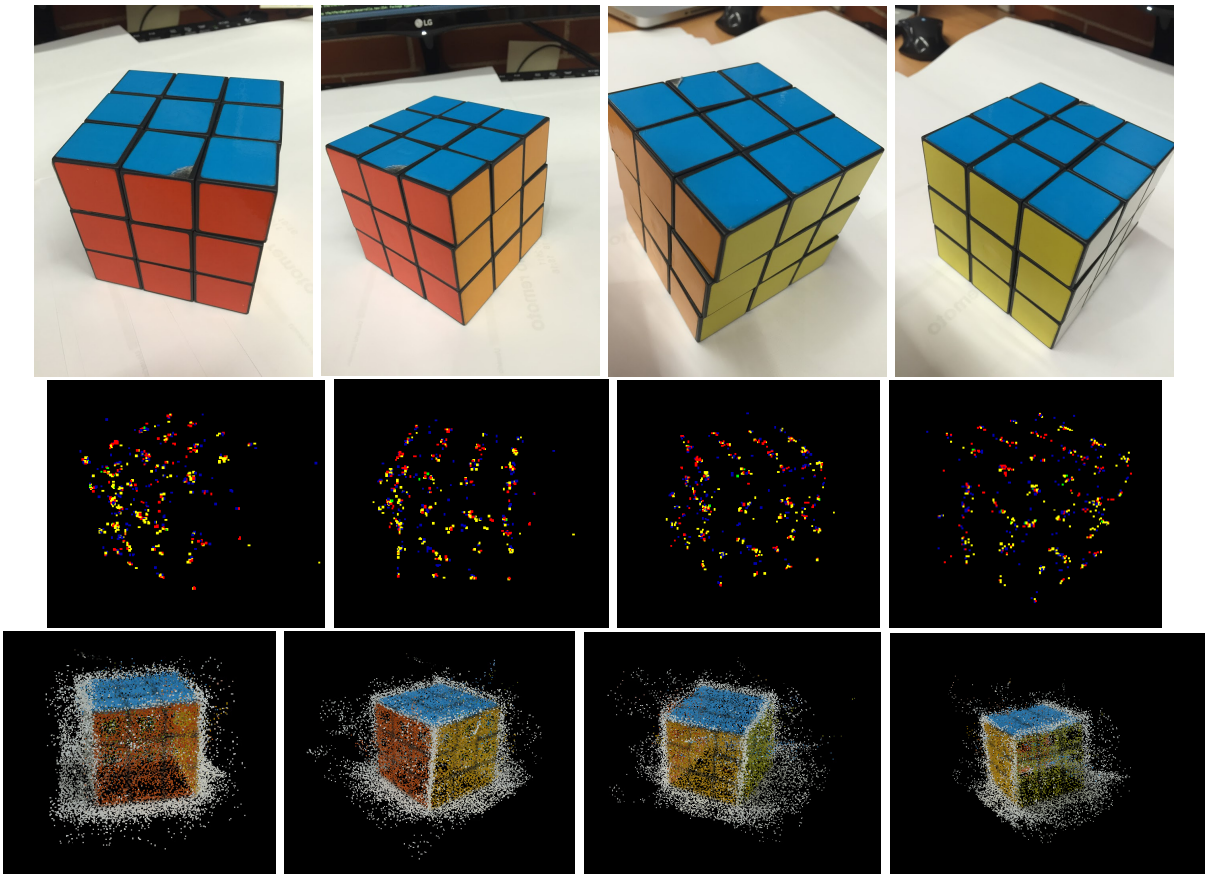

Figura 5.2: Objeto 2: Arriba: Imágenes RGB. Medio: Reconstrucción sparse. Abajo: Reconstrucción densa

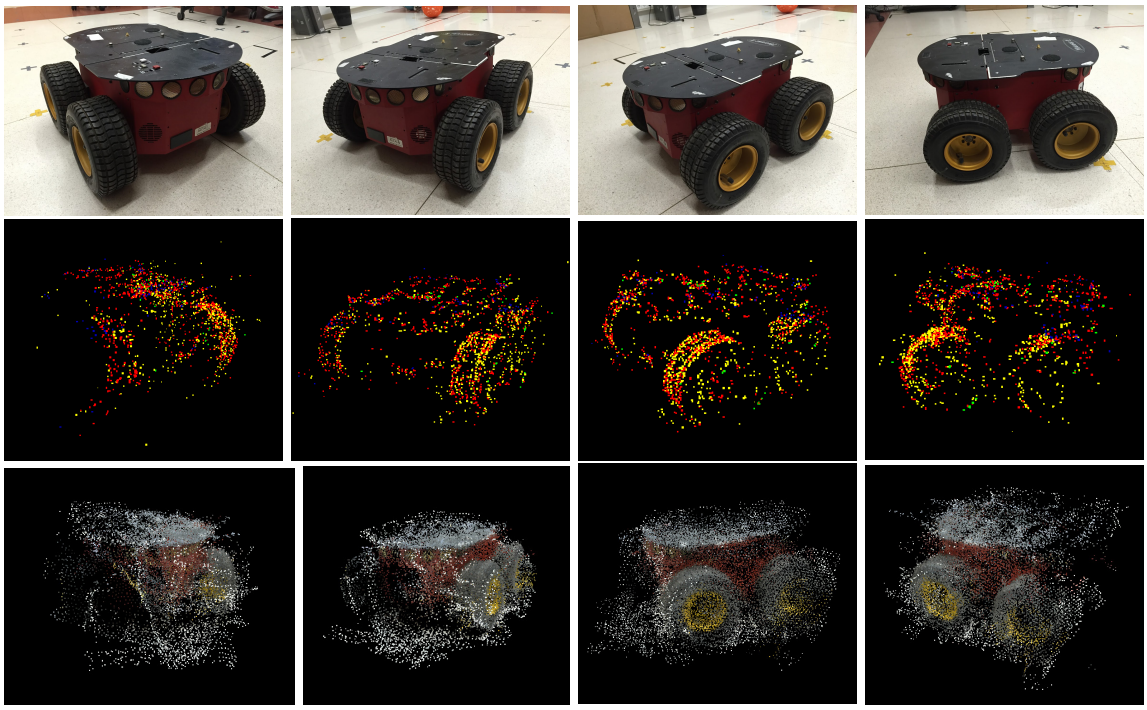

Figura 5.3: Objeto 3: Arriba: Imágenes RGB. Medio: Reconstrucción sparse. Abajo: Reconstrucción densa

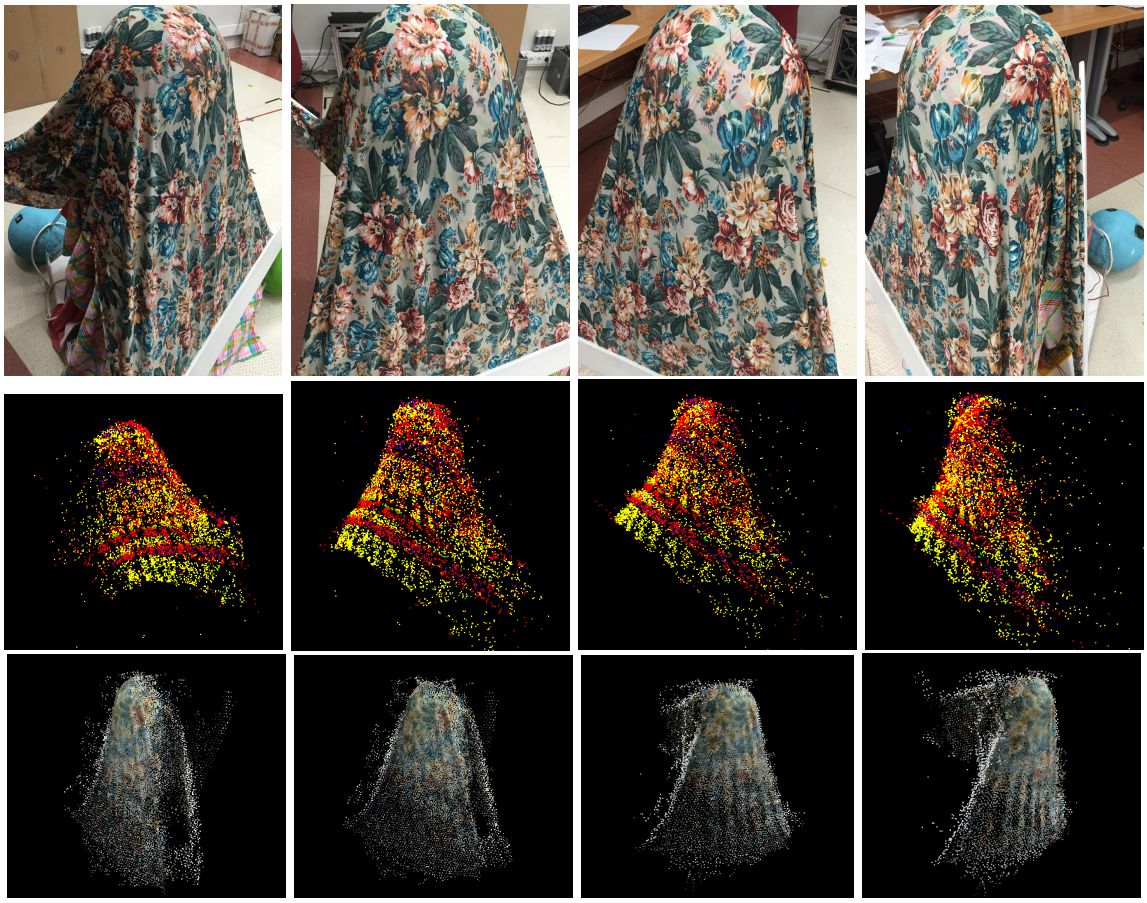

Figura 5.4: Objeto 4: Arriba: Imágenes RGB. Medio: Reconstrucción sparse. Abajo: Reconstrucción densa

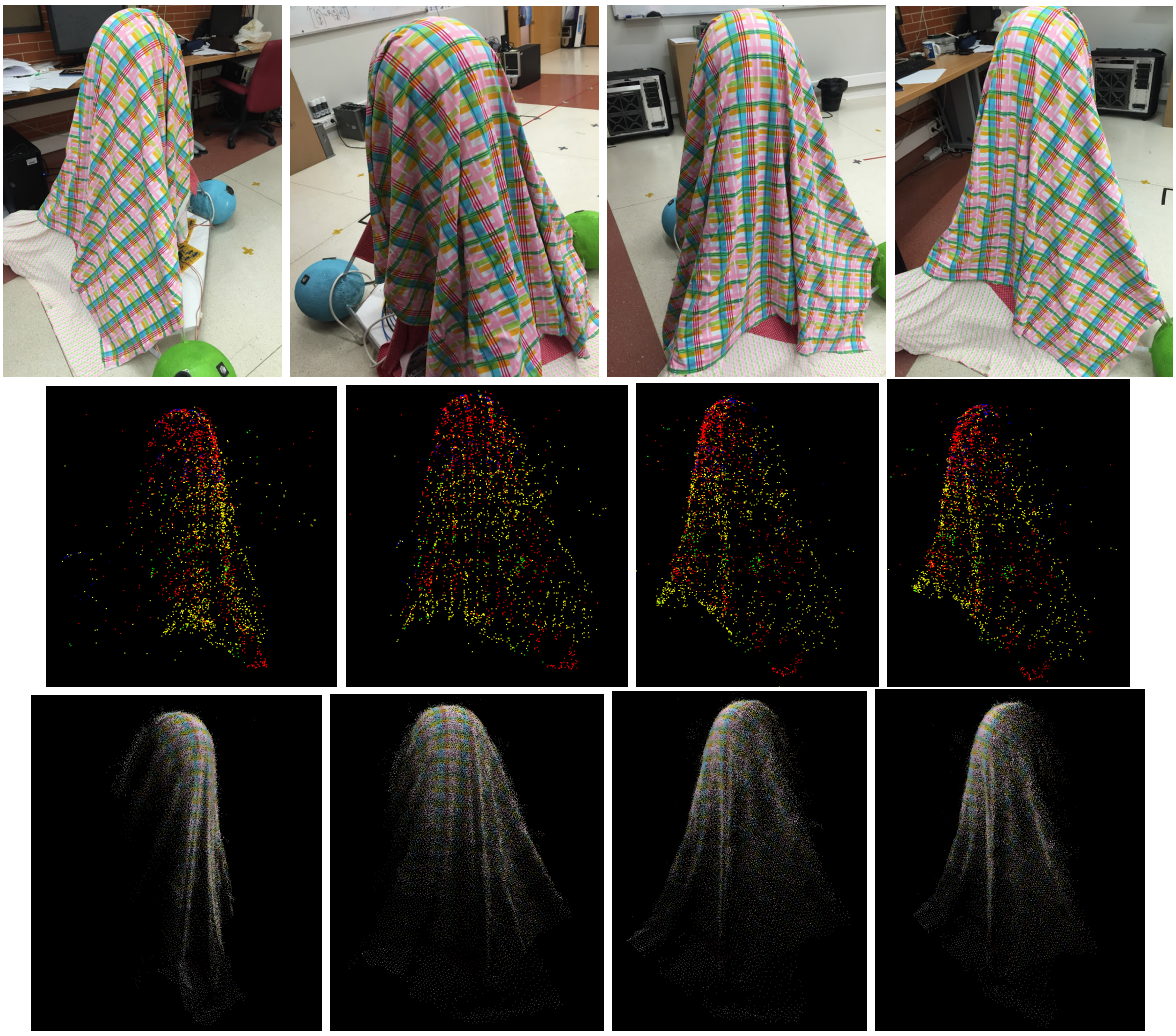

Figura 5.5: Objeto 5: Arriba: Imágenes RGB. Medio: Reconstrucción sparse. Abajo: Reconstrucción densa

La tasa de refresco de la cámara usada es de aproximadamente 30fps. Con un movimiento continuo de la cámara se tiene una tasa de refresco de la reconstrucción sparse para un objeto con características medias de aproximadamente 25fps. Para el caso de la reconstrucción densa la tasa de refresco se ve más afectada ya que se realiza un procesado por cada tres fotogramas válidos. Por lo tanto la reconstrucción densa del objeto se lleva a cabo con una tasa de refresco de aproximadamente 10fps.

Estos datos dependen en gran medida de la forma en la que el usuario realiza el movimiento de la cámara. Esto es, que si el usuario no mueve la cámara o la mueve de forma muy lenta la reconstrucción es muy lenta. Por otro lado si el usuario realiza un movimiento brusco de la cámara la reconstrucción no es válida ya que los fotogramas seleccionados para la reconstrucción densa se verían distorsionados por el movimiento.

A la vista de los resultados gráficos obtenidos, los resultados son bastante precisos. Sin embargo, existen bastantes puntos externos que no pertenecen al modelo que se han capturado del fondo. Para una mejora de la visualización por parte del especialista a los modelos resultantes se les debería de aplicar un filtrado de puntos. Este filtrado podría ser heurístico, partiendo de la forma del objeto o simplemente el cirujano podría hacer una segmentación manual rodeando en la pantalla la zona de interés de tal forma que los puntos que no se encuentren dentro de esa zona de interés serían eliminados, por poner un ejemplo.

En cuanto a la interfaz gráfica se refiere, es bastante fluida. Es capaz de almacenar y cargar los archivos de datos de nubes de puntos sin ninguna demora. Los indicadores de la interfaz funcionan de manera perfectamente fluida y sin errores apreciables.

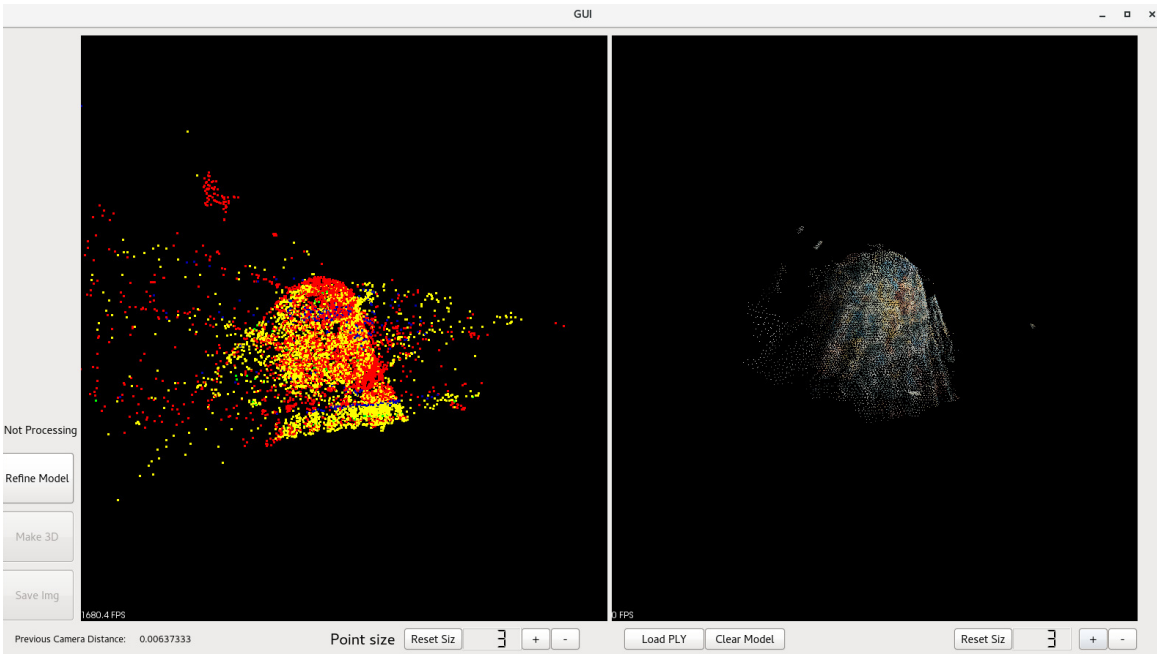

Figura 5.6: Interfaz de usuario en acción.

Capítulo 6

Conclusiones y líneas futuras

6.1 Introducción

Esta sección concluye este Trabajo Fin de Máster (TFM) y en ella se hace una reflexión sobre las conclusiones. Además de eso se detallan las mejoras que pueden ser aplicadas a la herramienta software del presente TFM así como las posibles futuras lineas de investigación relacionadas.

6.2 Conclusiones

En este TFM se ha desarrollado una herramienta software que ofrece una reconstrucción incremental en tiempo real enfocado a la reconstrucción gráfica de un órgano del cuerpo humano a partir de imágenes provenientes de una sola cámara no estéreo utilizada en operaciones de laparoscopia.

Se hizo un estudio de las técnicas de Cirugía Mínimamente Invasiva (MIS) presentes en el estado del arte y los trabajos relacionados con Realidad Aumentada (AR) en el ámbito de la medicina. También se estudiaron detenidamente las técnicas de reconstrucción 3D presentes así como las limitaciones de las mismas. Se hizo un diseño de la arquitectura software del programa a desarrollar implementando técnicas conocidas de reconstrucción 3D. Posteriormente se llevó a cabo el desarrollo de la herramienta implementando algunas herramientas de código abierto existentes en el estado de la técnica.

Como conclusión, en este TFM se ha desarrollado satisfactoriamente un prototipo de una herramienta de reconstrucción incremental en tiempo real que ofrece al cirujano la posibilidad de supervisar el estado de la reconstrucción 3D a medida que realiza el movimiento exploratorio con la cámara de laparoscopia. Sin embargo, esta herramienta presenta varias limitaciones como son: la herramienta puede llegar a tener en cuenta puntos residuales no deseados en la reconstrucción correspondientes al fondo de la imagen. Otra limitación del sistema es que el órgano debe presentar cierta textura para una correcta detección de características de la parte de Structure-from-Motion (SfM) del sistema.

En definitiva, esta herramienta supone dar un paso más en el avance de la utilización de AR en operaciones de laparoscopia mejorando los resultados en quirófano. Existen artículos científicos [5] que demuestran que los sistemas de AR propuestos mejoran sustancialmente la presión por parte del especialista.

6.3 Líneas futuras

En esta sección se enumeran las líneas futuras que derivan de este TFM. Dichas lineas futuras y mejoras se detallan a continuación:

- Inclusión de una herramienta de segmentación del útero. Dicha herramienta puede ser manual; el cirujano dibuja en una pantalla táctil el contorno del útero o, idealmente, automática.
- Inclusión de mejores detectores de características. En el sistema propuesto las características son detectadas mediante detectores de características llamados FAST los cuales se caracterizan por su velocidad. Sin embargo, hay diversas texturas en las que las características no son bien detectadas, por ello, incluyendo otro tipo de detector de características como por ejemplo SURF se mejorarían los resultados obtenidos siendo los modelos mucho más densos.
- Adaptación e integración en GPU. Varios métodos del software desarrollado son procesados mediante la tarjeta gráfica o, lo que es lo mismo, en GPU. Sin embargo, el procesado de Patch-based Multi-view Stereo (PMVS) se realiza en la CPU. Por lo tanto, se mejoraría el tiempo de refresco del modelo denso si este proceso del sistema se realizara en GPU.
- Integración con los trabajos desarrollados en [2] de tal forma que la información del proceso preoperatorio se viera fusionado con el obtenido de la reconstrucción tridimensional.
- Mejora de la estética de la interfaz gráfica. QtCreator ofrece multitud de widgets que pueden ser incluidos en el desarrollo de la interfaz gráfica de usuario para una mejora en la estética de la interfaz gráfica.

Bibliografía

[1] T. Collins, D. Pizarro, A. Bartoli, M. Canis, and N. Bourdel, "Realtime wide-baseline registration of the uterus in laparoscopic videos using multiple texture maps," in *Augmented Reality Environments for Medical Imaging and Computer-Assisted Interventions*. Springer, 2013, pp. 162–171.

[2] ——, "Computer-assisted laparoscopic myomectomy by augmenting the uterus with pre-operative mri data," in *Mixed and Augmented Reality (ISMAR), 2014 IEEE International Symposium on*. IEEE, 2014, pp. 243–248.

[3] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on*. IEEE, 2007, pp. 225–234.

[4] Y. Furukawa and J. Ponce, "Accurate, dense, and robust multiview stereopsis," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 8, pp. 1362–1376, 2010.

[5] N. Bourdel, T. Collins, D. Pizarro, A. Bartoli, D. Da Ines, B. Perreira, and M. Canis, "Augmented reality in gynecologic surgery: evaluation of potential benefits for myomectomy in an experimental uterine model," *Surgical endoscopy*, pp. 1–6, 2016.

[6] A. Bartoli, T. Collins, N. Bourdel, and M. Canis, "Computer assisted minimally invasive surgery: Is medical computer vision the answer to improving laparosurgery?" *Medical hypotheses*, vol. 79, no. 6, pp. 858–863, 2012.

[7] D. J. Mirota, M. Ishii, and G. D. Hager, "Vision-based navigation in image-guided interventions," *Annual review of biomedical engineering*, vol. 13, pp. 297–319, 2011.

[8] L.-M. Su, B. P. Vagvolgyi, R. Agarwal, C. E. Reiley, R. H. Taylor, and G. D. Hager, "Augmented reality during robot-assisted laparoscopic partial nephrectomy: toward real-time 3d-ct to stereoscopic video registration," *Urology*, vol. 73, no. 4, pp. 896–900, 2009.

[9] T. Simpfendörfer, M. Baumhauer, M. Müller, C. N. Gutt, H.-P. Meinzer, J. J. Rassweiler, S. Guven, and D. Teber, "Augmented reality visualization during laparoscopic radical prostatectomy," *Journal of endourology*, vol. 25, no. 12, pp. 1841–1845, 2011.

[10] G. A. Puerto-Souza, J. A. Cadeddu, and G.-L. Mariottini, "Toward long-term and accurate augmented-reality for monocular endoscopic videos," *IEEE Transactions on Biomedical Engineering*, vol. 61, no. 10, pp. 2609–2620, 2014.

[11] A. Amir-Khalili, M. S. Nosrati, J.-M. Peyrat, G. Hamarneh, and R. Abugharbieh, "Uncertainty-encoded augmented reality for robot-assisted partial nephrectomy: A phantom study," in *Augmented Reality Environments for Medical Imaging and Computer-Assisted Interventions*. Springer, 2013, pp. 182–191.

[12] D. Cohen, E. Mayer, D. Chen, A. Anstee, J. Vale, G.-Z. Yang, A. Darzi *et al.*, "Augmented reality image guidance in minimally invasive prostatectomy," in *International Workshop on Prostate Cancer Imaging*. Springer, 2010, pp. 101–110.

[13] D. Stoyanov, M. V. Scarzanella, P. Pratt, and G.-Z. Yang, "Real-time stereo reconstruction in robotically assisted minimally invasive surgery," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2010, pp. 275–282.

[14] N. Haouchine, J. Dequidt, I. Peterlik, E. Kerrien, M.-O. Berger, and S. Cotin, "Image-guided simulation of heterogeneous tissue deformation for augmented reality during hepatic surgery," in *Mixed and Augmented Reality (ISMAR), 2013 IEEE International Symposium on*. IEEE, 2013, pp. 199–208.

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

[16] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[17] "Boujou," http://www.boujou.com.

[18] D. Nister and H. Stewenius, "Scalable recognition with a vocabulary tree," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 2. IEEE, 2006, pp. 2161–2168.

[19] N. Snavely, S. M. Seitz, and R. Szeliski, "Skeletal graphs for efficient structure from motion." in *CVPR*, vol. 1, 2008, p. 2.

[20] J.-M. Frahm, P. Fite-Georgel, D. Gallup, T. Johnson, R. Raguram, C. Wu, Y.-H. Jen, E. Dunn, B. Clipp, S. Lazebnik *et al.*, "Building rome on a cloudless day," in *European Conference on Computer Vision*. Springer, 2010, pp. 368–381.

[21] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski, "Building rome in a day," in *2009 IEEE 12th international conference on computer vision*. IEEE, 2009, pp. 72–79.

[22] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski, "A comparison and evaluation of multi-view stereo reconstruction algorithms," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 1. IEEE, 2006, pp. 519–528.

[23] C. Strecha, W. von Hansen, L. Van Gool, P. Fua, and U. Thoennessen, "On benchmarking camera calibration and multi-view stereo for high resolution imagery," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. Ieee, 2008, pp. 1–8.

[24] "Estereopsis," https://es.wikipedia.org/wiki/Estereopsis.

[25] D. Marr, T. Poggio, E. C. Hildreth, and W. E. L. Grimson, "A computational theory of human stereo vision," in *From the Retina to the Neocortex*. Springer, 1991, pp. 263–295.

[26] "Google maps," http://maps.google.com.

[27] "Microsoft. bing maps," http://www.bing.com/maps.

[28] "Apple maps," http://www.apple.com/ios/maps.

[29] Y. Furukawa, C. Hernández *et al.*, *Multi-view stereo: A tutorial*. Citeseer, 2015.

[30] F. Devernay and O. Faugeras, "Straight lines have to be straight," *Machine vision and applications*, vol. 13, no. 1, pp. 14–24, 2001.

[31] "Parallel tracking and mapping for small ar workspaces - source code," http://www.robots.ox.ac.uk/ gk/PTAM/.

[32] D. Nistér, "An efficient solution to the five-point relative pose problem," *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, no. 6, pp. 756–770, 2004.