UNIVERSIDAD DE LAS PALMAS
DE GRAN CANARIA

ESCUELA DE INGENIERÍA DE TELECOMUNICACIÓN Y ELECTRÓNICA

PROYECTO FIN DE CARRERA

ESCÁNER 3D DE BAJO COSTE Y ALTA PRECISIÓN APLICADO A LA ADAPTACIÓN Y DISEÑO DE PRENDAS DE VESTIR

Autor: Cristina Girón Domínguez

Tutor: Dr. D. Gustavo Marrero Callicó

Titulación: Ingeniero de Telecomunicación

Fecha: julio 2018

UNIVERSIDAD DE LAS PALMAS
DE GRAN CANARIA

ESCUELA DE INGENIERÍA DE TELECOMUNICACIÓN Y ELECTRÓNICA

PROYECTO FIN DE CARRERA

ESCÁNER 3D DE BAJO COSTE Y ALTA PRECISIÓN APLICADO A LA ADAPTACIÓN Y DISEÑO DE PRENDAS DE VESTIR

HOJA DE FIRMAS

Alumna:

Fdo.: Cristina Girón Domínguez

Tutor:

Fdo.: Dr. D. Gustavo Marrero Callicó

Titulación: Ingeniero de Telecomunicación

Fecha: julio 2018

UNIVERSIDAD DE LAS PALMAS
DE GRAN CANARIA

ESCUELA DE INGENIERÍA DE TELECOMUNICACIÓN Y ELECTRÓNICA

PROYECTO FIN DE CARRERA

ESCÁNER 3D DE BAJO COSTE Y ALTA PRECISIÓN APLICADO A LA ADAPTACIÓN Y DISEÑO DE PRENDAS DE VESTIR

CALIFICACIÓN:

Presidente:

Fdo.:

Vocal:

Fdo.:

Secretario:

Fdo.:

Titulación: Ingeniero de Telecomunicación

Fecha: julio 2018

Agradecimientos

A mis padres, mi hermana y mis hermanos, por su ayuda, su apoyo incondicional y su paciencia infinita.

A mi tutor Gustavo, por apoyar este proyecto hasta el final.

A mis amigos, por sus palabras de ánimo siempre que necesitaba oírlas.

Índice de contenido

Índice de figuras

Índice de tablas

Capítulo 1.

Introducción

El presente proyecto de fin de carrera se realizará para la compañía adidas International Trading B.V. [adidasgroup, 2017] (de aquí en adelante referida como "la empresa"), situada en Herzogenaurach (Alemania), y en colaboración con la División de Diseño de Sistemas Integrados del Instituto Universitario de Microelectrónica Aplicada (IUMA) de la Universidad de Las Palmas de Gran Canaria (ULPGC).

Por ello, el material usado, producido o referenciado en este proyecto puede estar sujeto a cláusulas de confidencialidad, lo que se declara a los efectos oportunos, según se especifica en la siguiente reglamentación de la Escuela de Ingeniería de Telecomunicación y Electrónica (EITE): Art. 12 de la titulación de Ing. Telecomunicación y Art. 16 del reglamento en vigor.

1.1 Introducción

Los modelos 3D son cada vez más populares y si bien hoy en día los dispositivos de escaneado 3D se están convirtiendo en una fuente importante de información y captura de datos en muchas áreas de aplicación, el modelado basado en imágenes sigue siendo el enfoque más completo, portátil, flexible y económico ampliamente utilizado.

Actualmente existe en el mercado una gran variedad de herramientas basadas en fotogrametría digital, pero en nuestro caso nos basaremos en una solución sencilla, utilizando una cámara digital para la captura de imágenes y herramientas software herramientas *Open Source* (código abierto: software distribuido y desarrollado libremente) cuyo objetivo sea el de crear un modelo 3D de manera rápida y precisa a partir de un conjunto de imágenes, permitiendo la ubicación espacial del objeto a partir de un sistema de nube de puntos.

En líneas generales, se captura con una cámara digital convencional una secuencia de vistas sucesivas desde distintos ángulos de visión alrededor del objeto de estudio o escena.

Posteriormente, gracias a los parámetros intrínsecos de la cámara y mediante la importación del conjunto de vistas a distintos softwares específicos, se generan una serie de puntos característicos y correspondencias. El proceso de detección de puntos característicos y sus correspondencias es una etapa básica para la reconstrucción de puntos 3D, debido a que de ello depende por un lado la cantidad de puntos extraídos que determinarán el objeto de estudio o escena y, por otro lado, su resultado final en la aplicación. Este tipo de herramientas utilizan algoritmos basados en fotogrametría, como la tecnología *Estructura a partir del Movimiento* (SfM: Structure from Motion) [F. Dellaert y otros, 2000], que se basa en la reconstrucción tridimensional a partir de imágenes 2D procedentes de un sensor movimiento.

La motivación original del trabajo se encuentra en la aplicación de estas tecnologías para el desarrollo de una herramienta de trabajo para diseñadores, ya que en la actualidad una de las tendencias que existe en la industria de la moda es la de trabajar en la fase de diseño con virtualización y modelos 3D. La evolución y crecimiento de la industria de la moda en los últimos años requiere que las empresas sean creativas y competitivas a través del diseño y la producción. Este tipo de tecnologías se utiliza a menudo en diversas aplicaciones en arquitectura y bellas artes [I. Navarro y otros, 2016] [L. Ling, y otros, 2017], utilizándose también en animación, entretenimiento e incluso en el ámbito académico [J. D. Camba, y otros, 2016] y médico [Z. Xie y otros, 2010]. Por tanto, no es de extrañar que este tipo de tecnología también haya dado el salto en los últimos años al campo de la moda.

El uso de este tipo de tecnología conlleva numerosas ventajas a la hora de promocionar y vender prendas de vestir, pudiendo utilizar la representación de un modelo 3D para realizar cualquier tipo de promoción del producto incluso antes de que éste se haya producido, en lugar de esperar a obtener una copia física de su diseño. Al obtener una imagen renderizada del modelo 3D, se puede además decidir si su realización es físicamente posible, pudiendo evaluar la calidad del producto resaltando cualquier problema de diseño imprevisto. A veces, ciertos diseños de moda pueden verse bien sobre el papel, pero son difíciles de ejecutar físicamente. El uso de un modelo 3D virtual facilita al diseñador descubrir cualquier tipo de complicación antes de que el diseño entre en producción, permitiendo modificarlo con antelación y solventar cualquier tipo de problema futuro. Además, tecnologías como la impresión 3D permiten al diseñador validar su diseño a través del

análisis de prototipos virtuales, de los resultados de una simulación y pudiendo crear una propuesta física en un tiempo récord y a un coste muy bajo, reduciendo el número de prototipos físicos.

1.2 Antecedentes

Existen diversas fases a la hora de diseñar y fabricar una prenda de vestir (Figura 1). En primer lugar, nos encontramos una fase de diseño de concepto. Partiendo de un análisis de mercado, se estudia la viabilidad comercial, económica y técnica y se definen las características del producto. A continuación, se encuentra la fase de diseño en detalle en la que se definen aspectos sobre la realización del producto, la estética, los materiales que van a utilizarse, etc. En esta fase se puede hacer uso de herramientas como maquetas o prototipos. La última fase de desarrollo se centra en la verificación y evaluación del producto, trasladando de manera fiable el proyecto técnico a la fábrica.

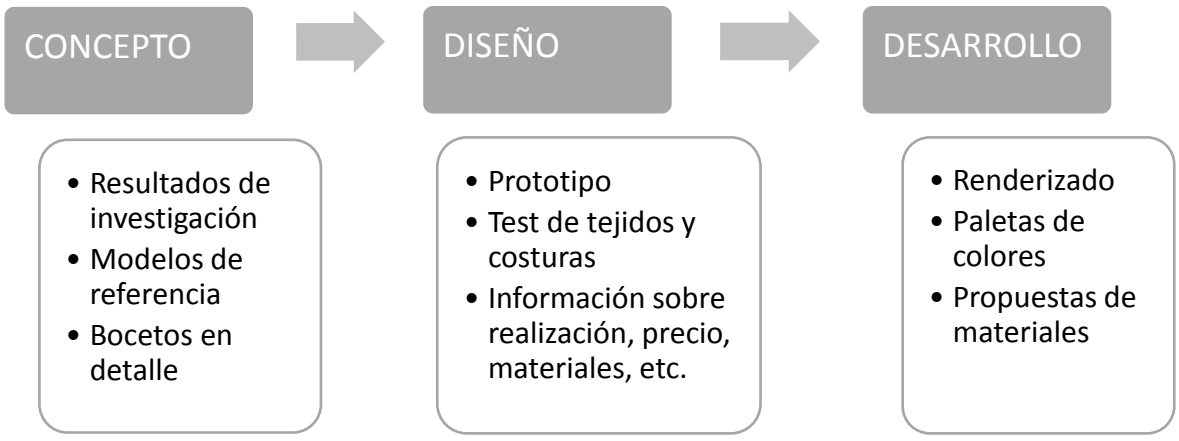

Figura 1. Fases del diseño de una prenda de vestir en la empresa

La creación de un sketch o esbozo es una de las primeras etapas dentro de la fase conceptual. En esta etapa únicamente está involucrado el equipo de diseño interno, cuyo propósito general es la creación y obtención de los primeros bocetos de los futuros productos clave de una colección. En el desarrollo de esta etapa, los diseñadores utilizan habitualmente herramientas simples de trabajo (papel, lápiz, lápiz, marcador, aerógrafo,

etc.) para crear bocetos a mano, partiendo como punto de referencia de muestras de temporadas anteriores y obteniendo modelos o prototipos bastante toscos (Figura 2).

Figura 2. Herramientas de trabajo de los diseñadores en la fase conceptual

Como se comentó anteriormente, una vez finalizada la fase centrada en el concepto comenzaría la fase de diseño, en la que están involucrados los diseñadores, el equipo de marketing y el equipo de desarrollo. El propósito de esta etapa es verificar si los diseños y la construcción son viables, antes de entrar en detalles de diseño y hacer muestras reales. Por lo tanto, podemos destacar la gran importancia de esta fase y la necesidad de herramientas de trabajo apropiadas. Entre las herramientas de trabajo actuales que se utilizan en esta etapa, encontramos modelos o muestras de anteriores colecciones (si estuvieran disponibles) y pliegos de condiciones y necesidades, para definir y describir la funcionalidad del producto. Los diseñadores también trabajan con herramientas software de diseño gráfico, como *Adobe Illustrator* o *Photoshop*, para obtener si fuera posible una representación renderizada de los modelos (Figura 3).

Figura 3. Herramientas de trabajo de los diseñadores y quipos de marketing y desarrollo

La etapa final del proceso de diseño se centra en las fases de desarrollo y producción, con la participación de diseñadores, equipos de desarrollo y marketing. En el desarrollo de esta etapa las herramientas de trabajo utilizadas son los primeros prototipos del producto, realizándose cada modelo en únicamente una o dos muestras de color.

Figura 4. Prototipos de calzado en una muestra de color (izquierda) o dos muestras (derecha) de la fase de desarrollo

Con el objetivo de mejorar y acortar el proceso de diseño se hace cada vez más necesario incorporar nuevas técnicas y herramientas software desde las primeras etapas del proceso. Además de las herramientas de diseño gráfico utilizadas por los diseñadores mencionadas

anteriormente, en los últimos años han aparecido otro tipo de técnicas que facilitan el proceso de diseño de prendas de vestir, como programas de diseño vectorial o diseño de prototipos de prendas en 3D.

La herramienta software llamada ReCap™ [Autodesk, 2018] (anteriormente conocida como Autodesk® 123D™ Catch) de Autodesk Labs supone un posible caso de uso por parte de un diseñador ya que permite tomar fotografías de un objeto, por ejemplo, de una prenda de ropa o calzado, y obtener un modelo del mismo en tres dimensiones (Figura 5). Esta herramienta se basa en la fotogrametría, una técnica que obtiene las propiedades geométricas de un objeto y su situación espacial a partir de una serie de imágenes fotográficas. Al trabajar con una fotografía podemos obtener información sobre la geometría del objeto (información bidimensional), pero si trabajamos con dos fotografías o más, a partir de la zona común (el solape) podemos obtener una visión estereoscópica del mismo (información tridimensional).

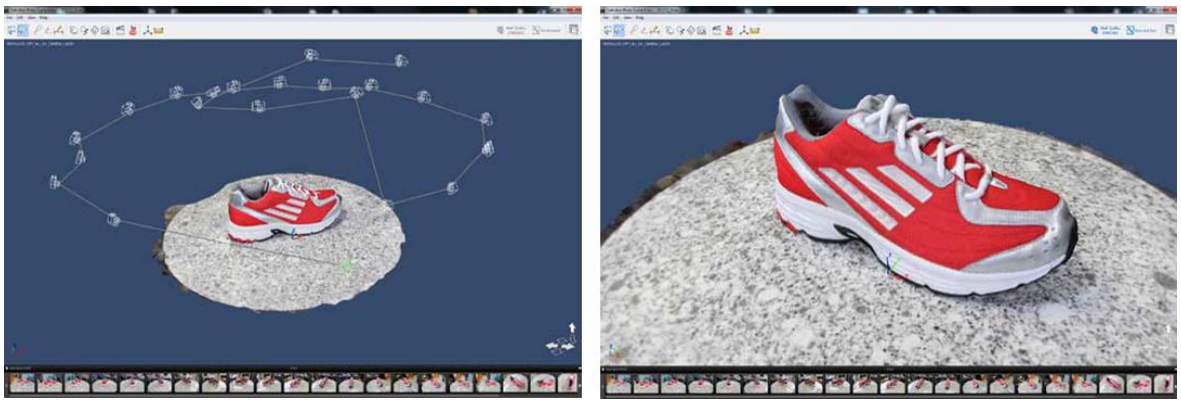

Figura 5. Escena creada a partir de las imágenes tomadas de una zapatilla de la empresa. Posiciones de la cámara respecto a la zapatilla (izquierda) y vista en detalle del modelo 3D (derecha)

Al trabajar con la herramienta, se parte de la toma de diversas fotografías, entre 40 y 50, realizadas desde varios puntos de vista, incluyendo la vista en alzado. Para evitar la falta de continuidad visual entre fotografías adyacentes, se utiliza un sistema de marcas para varios puntos del fondo, con un mínimo número de puntos de registro visibles desde cualquier posición.

Para obtener el modelo digital se necesita crear la malla de triángulos 3D correspondiente. Esta malla representa la superficie de un objeto utilizando facetas planas, compuestas por

un conjunto de triángulos conectados por sus vértices. Además del mallado del prototipo se obtiene su mapa de textura. Un mapa de textura o mapa UV es una forma de asignar la parte de una imagen a un polígono en el modelo 3D. Cada vértice del polígono es asignado a un par de coordenadas 2D que definen qué parte de la imagen es mapeada. Estas coordenadas 2D se llaman coordenadas UV (alternativa a las coordenadas XY, que junto con Z son usadas en el modelo 3D). En resumen, el mapeado UV es una manera de mapear texturas de tipo imagen sobre modelos tridimensionales. Estos mapas se ajustan perfectamente al modelo.

1.2.1 Trabajo realizado y problemas encontrados

Durante la etapa de prueba de la herramienta anteriormente mencionada se detectaron varios problemas. En primer lugar, el mapa de textura se genera automáticamente con una resolución de 4K (4096 x 4096 píxeles). Este mapa se divide en varias partes y se aplica automáticamente al objeto (Figura 6). De esta forma no se puede controlar la edición del mapa UV y existe por tanto un problema para volver a aplicarlo, como podría realizarse con la mayor parte del software 3D existente en el mercado. Aun así, el Departamento de Diseño en la empresa puede importar el objeto por completo a Adobe Photoshop y pintar directamente sobre la superficie 3D. Sobre un mapa de textura profesional sería más fácil hacer correcciones de color o diseño sobre las partes seleccionadas.

Figura 6. Mapa de textura (izquierda) y mallado (derecha) del modelo 3D

El segundo problema reside en que la herramienta genera automáticamente la estructura de base con la malla de triángulos (Figura 6). Esta malla se aplica aleatoriamente, por lo que no existe una manera de controlarla. Una vez aplicado el mapa de textura no se puede volver a generar la malla sobre el objeto, produciéndose entonces una pérdida de datos. Además de no tener acceso a la configuración del modelo, sólo se puede descargar el modelo en un formato texturizado (.obj), siendo Autodesk propietario de los derechos de dicho modelo.

1.3 Objetivos

El principal objetivo del presente proyecto fin de carrera es desarrollar un modelo matemático necesario para conseguir un mapa de textura y una malla 3D de una manera precisa y exacta, obteniendo de esta forma un modelo 3D de un objeto cualquiera a partir de fotografías 2D. Más allá de este objetivo, también se pretende crear un caso de uso útil para el Departamento de Diseño de la empresa, en el que los diseñadores podrían utilizar esta herramienta para obtener un modelo 3D más preciso, con una percepción más real sobre una muestra de una temporada anterior. Sobre estos modelos 3D, los diseñadores pueden obtener desde el principio resultados finales en diferentes pruebas de color, creando un nuevo punto de referencia para futuros diseños. Al utilizar este tipo de herramientas, el diseñador podría obtener diferentes modelos de un mismo producto a partir de un único prototipo 3D, permitiendo una visualización del resultado final del diseño para compartirlo con el resto de los departamentos involucrados en esta etapa, facilitando el trabajo en equipo. Además, a partir de los modelos 3D finales, se podrán obtener diferentes impresiones de nuevos cambios en los diseños.

Por otro lado, se debe tener en cuenta que la mayor parte de las herramientas utilizadas en el ámbito de la virtualización y la computación gráfica son soluciones propietarias que tiene un alto coste inicial al requerir del pago de licencias de software. Por tanto, uno de los intereses del proyecto reside en el desarrollo de una solución basada en el uso de herramientas *Open Source*. Esta máxima se mantendrá en la medida de lo posible, teniendo en cuenta que no siempre existe flexibilidad y facilidad de interacción entre herramientas de software libre.

1.4 Metodología

En la realización de este Proyecto Fin de Carrera se han llevado a cabo una serie de tareas que podemos agrupar en diferentes etapas, desde una fase inicial de documentación hasta la fase de evaluación del sistema. Éstas y otras etapas de planificación se detallan a continuación:

1ª Etapa. Este primer paso consistió en la búsqueda y lectura de información de diferentes estudios realizados hasta el presente sobre la generación de modelos 3D a partir de imágenes 2D, y los diferentes métodos y algoritmos utilizados hasta el momento, con el propósito de adquirir los conocimientos básicos necesarios para la realización del PFC.

2ª Etapa. En este siguiente paso se realizó un estudio de las diferentes herramientas presentes en el mercado de virtualización y computación gráfica, así como la selección de herramientas necesarias para el desarrollo del proyecto.

3ª Etapa. En esta fase se implementó el método propuesto para obtener de forma precisa mediante diferentes algoritmos el mallado 3D, el mapa de textura y finalmente el modelo 3D, a partir de la adquisición y registro de diferentes conjuntos de datos (imágenes).

4ª Etapa. Esta cuarta etapa, que junto con la anterior forman el núcleo principal del trabajo, consistió en la fase de pruebas y simulación del método diseñado y su funcionamiento, realizando las correcciones necesarias.

5ª Etapa. La última etapa consistió en elaborar y redactar toda la información generada en este documento.

1.5 Organización de la memoria

La redacción de la presente memoria se estructura en dos bloques principales formados por 5 capítulos y un anexo, seguidos de la correspondiente bibliografía, pliego de condiciones y hojas de presupuesto. A continuación, se disponen algunas pinceladas de estos apartados.

- **Capítulo 1. Introducción**. Introducción del tema tratado, justificando su necesidad y describiendo sus antecedentes. Se detallan los antecedentes. Se fijan los objetivos y se especifica la estructura del sistema desarrollado
- **Capítulo 2**. **Estado del arte.** Estudio de la literatura existente y de las herramientas utilizadas en el ámbito de la fotogrametría para el modelado 3D.
- **Capítulo 3**. **Diseño del escáner 3D.** Descripción de la solución propuesta, así como los algoritmos y herramientas utilizadas.
- **Capítulo 4**. **Evaluación de la implementación.** Descripción de la metodología y los experimentos realizados, así como los resultados obtenidos en cada uno de ellos.
- **Capítulo 5**. **Conclusiones y líneas futuras.** Conclusiones obtenidas a partir del trabajo realizado y posibles líneas futuras de trabajo.
- **Anexo A.** Se describe el software utilizado, así como su instalación y guía de uso.
- **Bibliografía**. Se detalla toda la bibliografía utilizada en este trabajo.
- **Pliego de condiciones.** Se muestran las herramientas software y hardware utilizadas.
- **Presupuesto.** Se detalla el presupuesto necesario para la realización de este Proyecto Fin de Carrera.

Capítulo 2.

Estado del arte

2.1 Fotogrametría y herramientas de reconstrucción 3D

La fotogrametría se ha convertido en los últimos tiempos en una de las principales herramientas utilizadas en los campos de la arqueología, arte rupestre, conservación de bienes culturales y en general todas las disciplinas relacionadas con la protección, documentación, digitalización o virtualización del patrimonio, por ser una herramienta bastante asequible en cuanto a los costes, por la simpleza de su uso y por constituir una fuente de documentación bastante relevante. La fotogrametría es la técnica que obtiene las propiedades geométricas de un objeto y su ubicación espacial a partir de una serie de imágenes fotográficas. A partir de una sola imagen, se puede obtener información sobre la geometría del objeto (información bidimensional), pero a partir de dos o más imágenes y su área común (el solapamiento), podemos obtener una vista estereoscópica (información tridimensional). A partir de estas áreas de solapamiento podemos determinar puntos comunes entre imágenes, tomadas de forma controlada y con cámaras calibradas, con los que reproducir las vistas 3D.

El diseño 3D supone una disciplina exigente, tanto en cuanto a habilidades y conocimientos técnicos, como a las capacidades de hardware y software necesarias para llevarla a cabo. Además de la inversión necesaria para cumplir con los requerimientos de hardware en cuanto a procesador, tarjeta gráfica y memoria RAM, la gran mayoría de software 3D profesional supone el pago inicial de una licencia costosa para su uso.

Actualmente existe una gran variedad de programas de diseño 3D y software de modelado 3D en el mercado, utilizados en sectores tan variados como la impresión 3D, la animación, los juegos, la arquitectura y el diseño industrial, etc. Con el objetivo de buscar y seleccionar las herramientas necesarias para la realización del proyecto, se presenta a continuación una tabla comparativa de herramientas para fotogrametría, en base a su sistema operativo y al tipo de licencia necesaria:

SOFTWARE	SISTEMA OPERATIVO	LICENCIA
Visual SFM	Linux, OSX, Windows	Libre
Open MVS	Linux, OSX, Windows	Libre
PhotoModeler	Windows	Desde 3500 €
Cinema 4D	Windows, OSX	3570 €
3DSOM	Windows, OSX	995 $
Pix4D	Windows, OSX (Beta), Online	260 €/mes 2600 €/año 6500 € perpetua
PhotoScan	Linux, OSX, Windows	179 $ Standard 3499 $ Professional

Tabla 1. Comparativa de software comercial

Con el fin de simplificar la lista, mencionar que se han excluido los programas de diseño 3D empleados principalmente en los campos de animación y juegos, así como el software comercial utilizado con el uso de drones y en impresión 3D. Estas herramientas suelen estar diseñadas con interfaces de usuario más complejas y difíciles de entender, requiriendo además una gran cantidad de conocimientos y especializaciones muy particulares dentro del diseño 3D.

A partir del estudio de herramientas realizado, podemos concluir que nos encontramos por tanto con varias limitaciones a la hora de encontrar un programa comercial con el que obtener una única herramienta que cumpla con el objetivo descrito en el presente proyecto de utilizar en la medida de lo posible, herramientas *Open Source*. Las herramientas libres y de código abierto son herramientas con mucho potencial, que permiten al usuario no solo el acceso a los datos y al proceso de trabajo, sino también a sus códigos de programación. Este tipo de herramientas generan una nube de puntos 3D que normalmente han de ser visualizadas y convertidas en malla en otro software.

2.2 Sistemas sin sensores de profundidad

Los seres humanos pueden distinguir con facilidad la estructura tridimensional del mundo que nos rodea y sus propiedades, como lo son la forma y los patrones de luz y sombreado a través de la superficie de los objetos de una escena.

En campos como la física o la computación gráfica, se desarrollan técnicas matemáticas para recuperar y plasmar la apariencia y la forma tridimensional de los objetos en imágenes planas. En el campo de la visión artificial se trata de hacer lo contrario, es decir, describir el mundo que vemos en una o más imágenes pudiendo reconstruir las propiedades anteriormente mencionadas: la forma, la iluminación y la distribución de color [R. Szeliski y otros, 2010]. Se trata de una tarea sumamente sencilla para el ser humano, pero propensa a errores en los algoritmos de visión artificial. Además del procesado de la imagen, es necesario obtener información sobre la profundidad de la escena, mediante diversos sensores específicos. A continuación, se presentan dos técnicas que no utilizan sensores de profundidad, si no que captan la información utilizando únicamente imágenes.

Visión estéreo

La visión estéreo se basa en la recuperación de la estructura 3D de una escena utilizando dos o más imágenes [StereoCamera, 2004], cada una de ellas adquirida desde un punto de vista diferente en el espacio. Las características principales de este sistema serían las siguientes:

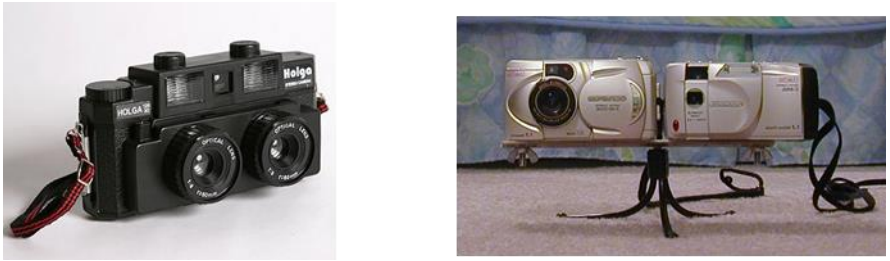

Figura 7. Modelos de cámaras estéreo

- Existen 2 cámaras para medir las distancias desde ellas a un punto de la escena (similar a la percepción de profundidad de la vista del ser humano).
- Las cámaras están alineadas con una distancia de separación entre ellas conocida.
- Cada cámara captura una imagen, y estas imágenes se analizan en busca de puntos o características comunes.
- Es necesario aplicar un método de triangulación para obtener la posición relativa de estos puntos que coinciden en las imágenes.

Structure from Motion

El proceso de reconstrucción de una escena representada por un conjunto de imágenes 2D recibe el nombre de Estructura a partir del Movimiento (SfM: Structure from Motion) [F. Dellaert y otros, 2000]. Es una técnica que, aunque inspirada en la fotogrametría (concepto anteriormente mencionado en el primer punto del presente capítulo) proviene del mundo de la visión artificial y permite realizar modelos 3D a partir de colecciones de imágenes no estructuradas tomadas mediante una cámara convencional cualquiera, por lo que resulta una estrategia más versátil. Actualmente la fotogrametría y SfM se entremezclan en los diversos programas que soportan este tipo de tareas, reduciendo todo a simples procesos de fotogrametría, cuando deberíamos hablar de SfM.

En el campo de visión artificial, SfM se refiere al proceso de encontrar la estructura tridimensional mediante el análisis del movimiento de un objeto a lo largo del tiempo. En este proceso nos encontramos con un problema similar al de encontrar la estructura a partir de la visión estéreo, ya que en ambos casos es necesario encontrar la correspondencia entre las imágenes 2D y la reconstrucción del objeto 3D.

El planteamiento de este problema sería el siguiente: dado un conjunto de imágenes de una escena estática con puntos 2D que se corresponden (en la Figura 8 se muestran como puntos codificados por colores), es necesario encontrar un conjunto P de puntos 3D, una rotación R y una posición t de las cámaras que explican dichas correspondencias. En otras palabras, cuando proyectamos un punto en cualquiera de las cámaras, el error de reproyección entre los puntos 2D proyectados y observados debe ser mínimo.

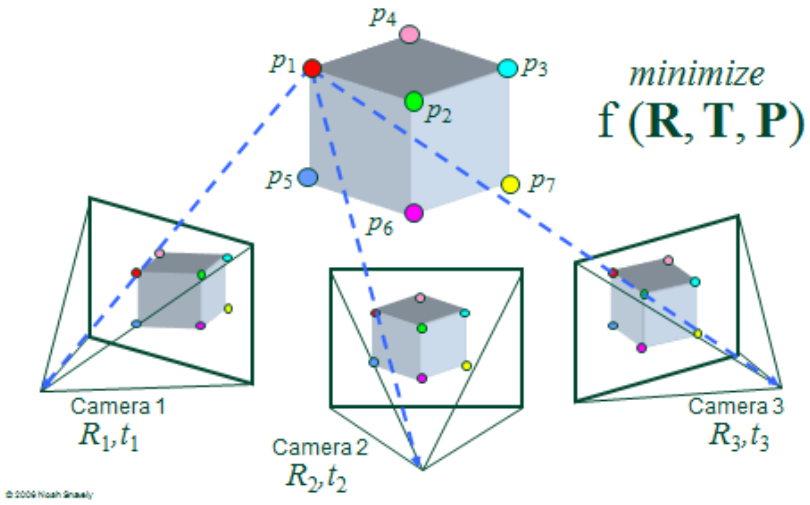

Figura 8. Proceso SfM [S. Agarwal y otros, 2009].

Para solucionar este problema, necesitamos por tanto una solución SfM que implique los siguientes pasos:

1) calibración de las cámaras,
2) aplicación de una solución para la estructura de la escena y el movimiento de las cámaras,
3) extracción de características y puntos comunes de las imágenes tomadas de una escena,
4) representación de una nube densa de puntos de la escena,
5) extrapolación de las propiedades geométricas y de textura de la escena.

La mayoría de las herramientas encontradas presentadas en el punto 2.1 del presente capítulo se basan en la funcionalidad de este algoritmo.

2.3 Detección de puntos característicos y correspondencias

La detección de puntos de interés o característicos tiene una gran importancia dentro del procesamiento de imágenes, al ser necesarios para realizar una búsqueda posterior de correspondencias entre imágenes que contienen información similar, pero que son presentadas de una forma diferente.

Podemos tomar como ejemplo las imágenes estéreo, es decir, múltiples vistas de una misma escena. Al tomar imágenes de un mismo escenario utilizando cámaras en diferentes posiciones, las imágenes obtenidas tendrán obviamente puntos en común, pero diferirán en ubicación, escala, orientación, etc. Esa es la razón por la cual es interesante desarrollar un software que sea capaz de encontrar y asociar estos puntos independientemente de cómo se presenten.

La propiedad más importante es su estabilidad a las interrupciones locales y globales. Estos pueden incluir rotaciones simples y transformaciones de traducción, ya sea más complejos, como variaciones y cambios de perspectiva y escala. Con el término cambio de escala, nos referimos a las variaciones en el tamaño de un objeto que aparece en la imagen debido, por ejemplo, a un movimiento longitudinal de la cámara. Para buscar este conjunto de puntos de interés, existen diferentes métodos.

Los sistemas de extracción de características deben ser robustos y capaces de funcionar bajo cualquier variación o transformación de la escena a reconocer. Entre tales variaciones podemos encontrar por ejemplo rotaciones y traslaciones, cambios en la iluminación, de perspectiva, de escala, etc. Cuando se habla de cambio de escala nos referimos a las variaciones en el tamaño de un objeto que aparece en la imagen debido, por ejemplo, a un movimiento longitudinal de la cámara.

El resultado de la extracción de características es una imagen simplificada que contiene únicamente las características principales de la imagen, necesarias para un tratamiento posterior.

Descriptores en el procesamiento de imágenes

Un descriptor, llamado también vector de características, es una herramienta de procesamiento de imágenes cuya función principal es extraer información y características de las imágenes. Los descriptores tienen también un papel importante en el reconocimiento de patrones, ya que son responsables de extraer las características locales, independientemente del patrón de referencia y de la región de la imagen, para luego hacer generar una coincidencia [S.A.J. Winder y otros, 2007].

Un descriptor debe cumplir las siguientes características [E. Vals y otros, 2015]:

- Tiene que ser robusto contra las posibles variaciones en la apariencia de los objetos.
- Debe ser lo suficientemente discriminativo, para poder distinguir los patrones de referencia almacenados en una imagen.
- Poseer determinadas invarianzas a la orientación de los diferentes objetos.
- Ser insensible a pequeñas imprecisiones al ubicar objetos en una imagen.

Todos estos factores pueden implementarse mediante diferentes características que definen el descriptor. Por esta razón, no existe un descriptor universal ya que, dependiendo del propósito de procesamiento para una imagen dada, responderán mejor uno u otros.

La búsqueda de correspondencias o *matching* es la etapa posterior a la búsqueda de puntos característicos. De esta forma, se pueden comparar los descriptores entre pares de imágenes, encontrando coincidencias entre ellas.

El *factoring* es un proceso que permite determinar la matriz de rotación R y el vector de traslación t (parámetros extrínsecos de la cámara), a partir de la matriz esencial E [R. Tsai y otros, 1982] [J. Weng y otros, 1989].

2.4 Geometría de la formación de imágenes

2.4.1 Modelo de cámara pinhole

En el campo de la visión artificial, la cámara puede considerarse como un dispositivo que proyecta y mapea el mundo tridimensional en una imagen bidimensional. Existen varios modelos que recogen esta transformación, dependiendo de la formación de la imagen obtenida y las características de la cámara empleada.

Para resolver el problema mencionado en el apartado 2.2 sobre la reconstrucción de una escena representada por dos o más imágenes, tomaremos el modelo de cámara pinhole. Este modelo de cámara es muy sencillo y ampliamente utilizado, por no disponer de lentes sino de una única apertura y muy pequeña (centro). Se puede aproximar a una caja a prueba de luz con un pequeño orificio en un lado, siendo éste el único agujero que permite que entre luz en la caja. Cuando la luz de una imagen pasa a través de este orificio, se forma

una imagen inversa en el lado opuesto de la caja (Figura 9). El ojo humano funciona según este mismo principio.

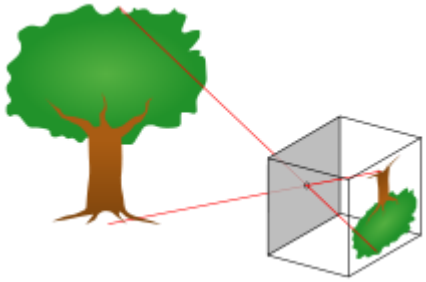

Figura 9. Diagrama de funcionamiento del modelo de cámara pinhole [Pinholecameramodel, 2018]

El modelo de cámara pinhole describe que cada punto del espacio tridimensional se proyecta a través de un rayo de luz que impacta en plano dando lugar a una imagen, atravesando el espacio en un único punto (apertura de la cámara), independientemente del punto de origen y del punto de impacto en la imagen [Orcero y otros, 2012]. En este modelo de cámara, cuanto más pequeño es el orificio a través del cual pasa la luz, más nítida y definida es la imagen que se forma dentro de la caja. Sin embargo, se debe tener en cuenta que la imagen se oscurece según el tamaño del agujero disminuye. Por lo tanto, se puede concluir que cuanto más definida sea la imagen, esta será también más opaca [R.J. Carrasco y otros, 2003].

Se debe tener en cuenta que al utilizar un modelo de cámara ideal es necesario añadir a nuestro modelo ciertos parámetros de distorsión que permitan corregirlo y aproximarlo al comportamiento real de la cámara.

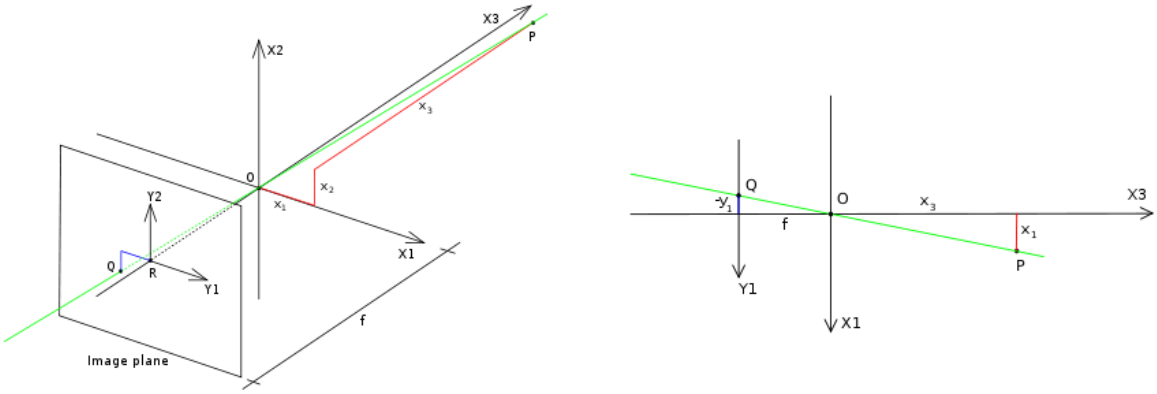

Figura 10. Geometría del modelo de cámara pinhole. Un punto P (x_1, x_2, x_3) se forma mediante proyección a un punto Q (y_1, y_2) en el plano imagen (izquierda). Modelo visto desde el eje X_2 (derecha) [Pinholecameramodel, 2018]

Según se observa en la imagen izquierda de la Figura 10, existe un sistema de coordenadas 2D en el plano de la imagen, con origen en *R* y con los ejes Y_1 e Y_2 paralelos a los ejes X_1 y X_2, respectivamente. Las coordenadas del punto *Q*, relativas a este sistema de coordenadas son (y_1, y_2). Suponemos que la apertura de la cámara pinhole, a través de la cual deben pasar todos los rayos de luz o líneas de proyección, es infinitamente pequeña, pudiendo considerarse como un punto. En la literatura este punto en el espacio tridimensional se conoce como el centro óptico de la cámara. Para comprender las dependencias entra las coordenadas (y_1, y_2) del punto *Q* y las coordenadas (x_1, x_2, x_3) del punto *P*, se presenta la imagen derecha de la Figura 10, con el modelo geométrico de la cámara visto desde el eje X_2.

Los elementos que forman parte de este modelo se definen a continuación:

- Un sistema de coordenadas ortogonales 3D con su origen en *O* (centro óptico o centro de proyección). En este punto también se encuentra la apertura de la cámara (el orificio), por la que pasan los rayos de luz. Los tres ejes del sistema de coordenadas se denominan X_1, X_2, X_3. El eje X_3 apunta en la dirección de visualización de la cámara y se denomina eje óptico, eje principal o rayo principal. El plano 3D que se cruza con los ejes X_1 y X_2 es el lado frontal de la cámara o plano principal.
- Un plano de imagen o plano focal, donde el mundo 3D se proyecta a través de la apertura de la cámara. El plano de la imagen es paralelo a los ejes X_1 y X_2 y se encuentra

a una distancia del origen O en la dirección negativa del eje X_3. Una implementación práctica de una cámara pinhole implica que el plano de la imagen se ubica de tal manera que interseccione el eje X_3 en la coordenada $-f$ donde $f > 0$. f también se conoce como la distancia focal (*fc*) de la cámara pinhole.

- Un punto *R* en la intersección del eje óptico y el plano de la imagen. Este punto se conoce como el punto principal o centro de la imagen.
- Un sistema coordenado en 2D en el plano de la imagen con origen en *R*. Los ejes del sistema de coordenadas se denominan Y_1, Y_2.
- Un punto *P* en algún lugar del espacio en las coordenadas (x_1, x_2, x_3) relativo a los ejes X_1, X_2, X_3.
- La línea de proyección del punto *P* en la cámara. En la imagen es la línea verde que pasa por el punto *P* y el punto *O*.
- Un punto *Q*, proyección del punto *P* en el plano de la imagen. Este punto viene dado por la intersección de la línea de proyección (verde) y el plano de la imagen.

2.4.2 Matriz de proyección

En la imagen derecha de la Figura 10 se pueden reconocer dos triángulos similares, ambos con partes de la línea de proyección (verde) como sus hipotenusas, siendo los catetos del triángulo izquierdo $-y_1$ y f, y del triángulo rectángulo x_1 y x_3. Dado que los dos triángulos son similares, se deduce que:

$$\frac{-y_1}{f} = \frac{x_1}{x_3} \qquad \textit{Ecuación 1}$$

De forma similar, mirando en la dirección negativa del eje X_1 se obtiene:

$$\frac{-y_2}{f} = \frac{x_2}{x_3} \qquad \textit{Ecuación 2}$$

En resumen:

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = -\frac{f_{\,c}}{x_3}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \qquad \textit{Ecuación 3}$$

Esta expresión describe la relación entre las coordenadas 3D del punto P y sus coordenadas 2D dadas por el punto Q en el plano de la imagen. Se trata de una proyección en perspectiva

seguida de una rotación de 180° en el plano de la imagen, debido a la inversión de la imagen que realiza una cámara pinhole real (Figura 9). Además, el tamaño de los objetos proyectados depende de su distancia al punto focal, y el tamaño total de la imagen depende de la distancia f_c entre el plano de la imagen y el punto focal.

Por último, para obtener la imagen en su posición original y no invertida, uno de los métodos utilizados al trabajar con una cámara digital consiste en leer los píxeles en un determinado orden. Por lo tanto, la ecuación obtenida es la Ecuación 3, pero sin el signo negativo.

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \frac{f_c}{x_3} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \qquad \textit{Ecuación 4}$$

Otra forma de describir el modelo de la cámara es a través de una matriz. La matriz de la cámara (también conocida como matriz de proyección) es una matriz de 3x4 que permite transformar las coordenadas 3D de los puntos en el espacio en coordenadas 2D de la imagen (píxeles).

Sea x una representación de un punto 3D en coordenadas homogéneas y sea y una representación de la imagen de este punto en la cámara pinhole. Entonces tenemos que:

$$y \sim Px \qquad \textit{Ecuación 5}$$

Donde P es la matriz homogénea de proyección la cámara (3x4). Para pasar de la Ecuación 4 a este modelo de matriz, se deben usar coordenadas homogéneas. El vector bidimensional se convertirá en un vector tridimensional al agregar una coordenada extra con valor unitario. Además, la igualdad se cambia por una equivalencia a una multiplicación escalar distinta de cero:

$$\begin{bmatrix} y_1 \\ y_2 \\ 1 \end{bmatrix} \sim -\frac{f_c}{x_3} \begin{bmatrix} x_1 \\ x_2 \\ -\frac{x_3}{f_c} \end{bmatrix} \sim \begin{bmatrix} x_1 \\ x_2 \\ -\frac{x_3}{f_c} \end{bmatrix} \qquad \textit{Ecuación 6}$$

El vector que anteriormente era un vector 3D, se expresa de la misma manera en coordenadas homogéneas, obteniendo la siguiente expresión:

$$\begin{bmatrix} y_1 \\ y_2 \\ 1 \end{bmatrix} \sim \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{-1}{f_c} & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ 1 \end{bmatrix} \qquad \textit{Ecuación 7}$$

Donde P es la matriz homogénea de proyección de la cámara, dada por:

$$P = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{-1}{f_c} & 0 \end{bmatrix} \qquad \textit{Ecuación 8}$$

Pudiendo expresarse también de la siguiente manera:

$$P = \begin{bmatrix} f_c & 0 & 0 & 0 \\ 0 & f_c & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = [K|0] \qquad \textit{Ecuación 9}$$

2.5 Reconstrucción del modelo

2.5.1 Movimiento de la cámara

A la hora de implementar la reconstrucción del modelo 3D es necesario obtener el movimiento entre las cámaras, es decir, la rotación y traslación que reflejan el movimiento desde la primera cámara a la segunda, así como su calibración, reflejada en la matriz de proyección.

A partir de las correspondencias de puntos obtenidas entre las dos imágenes durante el proceso de emparejamiento, se va a obtener la posición y orientación relativa de las cámaras empleando la matriz fundamental y la matriz esencial.

Parámetros intrínsecos de la cámara

Como podemos ver del resultado de la Ecuación 9, la matriz de proyección P se construye a partir de una matriz K y un vector de valores nulos, donde K es la matriz de calibración de la cámara, formada por una serie de parámetro denominados parámetros intrínsecos. Hasta este momento se ha tomado como origen de coordenadas de la imagen su centro óptico (O), pero esto no tiene por qué ser así. En general, se toma como origen de

coordenadas y punto principal la esquina inferior izquierda de la imagen (c_x, c_y), resultando la matriz de calibración de la siguiente forma:

$$K = \begin{bmatrix} f_c & \gamma & c_x \\ 0 & f_c & c_y \\ 0 & 0 & 1 \end{bmatrix} \qquad \textit{Ecuación 10}$$

Como se comentó anteriormente, la matriz de calibración está formada por los parámetros intrínsecos de la cámara, que se calculan para compensar las distorsiones introducidas debido a la geometría interna y la óptica de la cámara. A continuación, se describen estos parámetros [M. Hervás y otros, 2015]:

- El centro óptico O, o centro de proyección, es el punto de la cámara por el que pasan los rayos de luz que se proyectan en el plano imagen.
- El punto principal c con coordenadas (c_x, c_y) que indican el desplazamiento del centro de coordenadas del plano imagen respecto al punto principal.
- El factor de escala γ que determina el grado de perpendicularidad de las paredes de los píxeles del sensor de la cámara. Al ser inversamente proporcional a la tangente del ángulo que forman los ejes X e Y, tendrá valor nulo si los píxeles son rectangulares, siendo esta característica propia de casi todos los sensores utilizados hoy en día.
- La distancia focal f_c, es la distancia entre el centro óptico O y el punto principal c.

Distorsión

La extracción de los parámetros intrínsecos de la cámara se realiza para compensar las distorsiones introducidas debido al sistema óptico. El uso de lentes produce deformaciones en las imágenes que se forman en el sensor. Para el cálculo de coeficientes de distorsión, se tienen en cuenta factores radiales y tangenciales [M. Hervás y otros, 2015].

La distorsión radial consiste en el desplazamiento de los píxeles de la imagen, de tal modo que las líneas situadas en los extremos del encuadre aparentarán salir hacia el exterior o el interior. Para la corrección de la distorsión radial de las coordenadas de un píxel de la imagen, se utiliza la siguiente fórmula:

$$x_{corrected} = x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \qquad \textit{Ecuación 11}$$

$$y_{corrected} = y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$

La distorsión tangencial se produce porque la lente no se encuentra perfectamente paralela al plano de imagen. Se puede corregir a través de las siguientes fórmulas:

$$x_{corrected} = x + [2p_1 xy + p_2(r^2 + 2x^2)]$$
$$y_{corrected} = y + [p_1(r^2 + 2y^2) + 2p_2 xy] \qquad \textit{Ecuación 12}$$

De tal forma que se obtienen cinco parámetros de distorsión, representados como una matriz fila de 5 columnas:

$$Distortion_{coefficients} = [k_1 \quad k_2 \quad p_1 \quad p_2 \quad k_3] \qquad \textit{Ecuación 13}$$

Parámetros extrínsecos de la cámara

A diferencia de los parámetros intrínsecos que describen los parámetros internos de la cámara (centro óptico, punto principal, factor de escala y distancia focal), los parámetros extrínsecos indican la posición y orientación externas de la cámara en el mundo real (3D).

Hasta este momento se ha considerado el centro óptico como origen de coordenadas del mundo. A efectos prácticos debemos considerar que la cámara se mueve o gira alrededor de una escena, para captarla correctamente, siendo necesario introducir una transformación entre el marco de referencia conocido del mundo real y el marco de referencia desconocido de la cámara. Por tanto, debemos modificar la Ecuación 5 presentada anteriormente para introducir una matriz de transformación $[R|t]$ que contiene los parámetros extrínsecos de la cámara que se describen a continuación:

$$y = K[R|t]x \qquad \textit{Ecuación 14}$$

- Vector de traslación t: desplazamiento entre las posiciones de origen relativas entre dos imágenes de referencia (entre el origen del sistema de coordenadas inicial hasta le origen del sistema de coordenadas final).
- Matriz de rotación R: representa un giro de la cámara (o de un objeto respecto a ella).

Geometría epipolar

Con la idea de obtener la información necesaria para llevar a cabo la búsqueda de puntos de interés a lo largo de las líneas entre dos imágenes, se debe estudiar la geometría de la visión estéreo, denominada geometría epipolar. A continuación, se presentan algunos elementos de la geometría epipolar entre dos vistas (Figura 11):

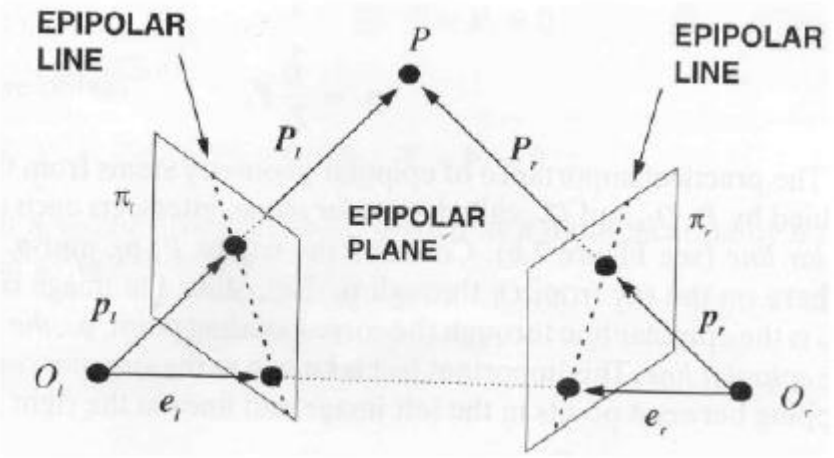

Figura 11. Geometría epipolar de la visión estéreo

- El epipolo: es el punto de intersección de la línea que une los dos centros de cámara (la línea de base) con el plano de la imagen.
- El plano epipolar: es un plano que contiene la línea de base (línea que une los dos centros de cámara y contiene a los epipolos). Existe una familia de planos epipolares (pencil of planes).
- La línea epipolar: es la intersección de un plano epipolar con el plano de la imagen. Todas las líneas epipolares se cruzan en el epipolo. Un plano epipolar intersecta los planos de imagen izquierda y derecha en líneas epipolares, y define la correspondencia entre las líneas.

Matriz fundamental y matriz esencial

La matriz fundamental es la representación algebraica de la geometría epipolar [J. Sánchez y otros, 2007]. Para cada punto en una imagen existe una línea en la otra imagen en la que

está incluido el punto de correspondencia. Luego existe una correspondencia entre un punto de una imagen y una línea en la otra. Esta información geométrica que relaciona dos puntos característicos diferentes de la misma escena se representa a través una matriz F conocida como matriz fundamental.

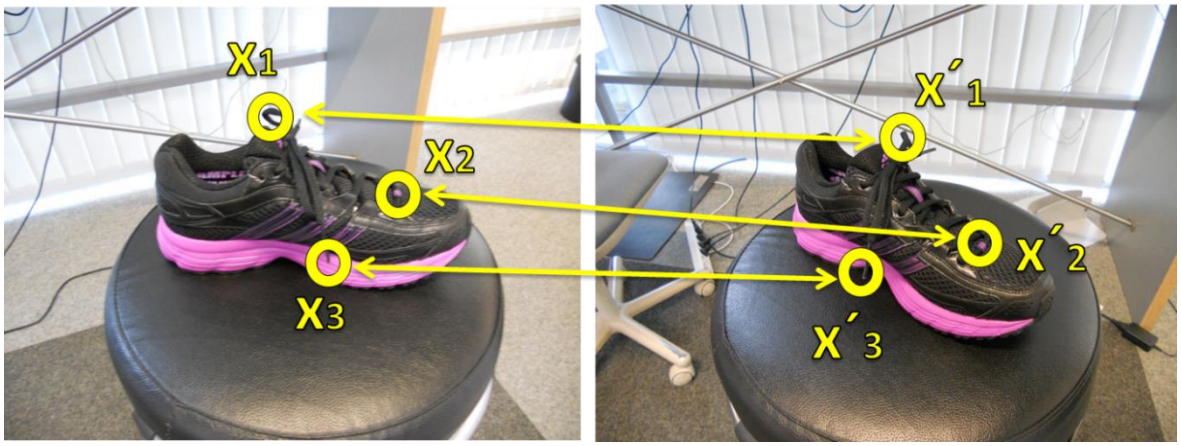

Figura 12. Correspondencia entre puntos característicos

Dado un par de imágenes, como se puede observar en la Figura 12, para cada punto x en una imagen, existe una línea epipolar correspondiente en la otra imagen. Cualquier punto x' en la segunda imagen que coincida con el punto x debe estar en la línea epipolar.

Como se mencionó anteriormente, la matriz fundamental F es una matriz de dimensión 3x3 que relaciona los puntos correspondientes en las imágenes estéreo. En la geometría epipolar, con coordenadas de imagen homogéneas x y x' (de los puntos correspondientes en un par de imágenes estéreo) Fx describe una línea (una línea epipolar) en la que debe estar el punto correspondiente x' en la otra imagen:

$$x'^{T} F x = 0 \qquad \textit{Ecuación 15}$$

La relación anterior que define la matriz fundamental fue publicada en 1992 por [R.I. Hartley y otros, 1992] y [O.D. Faugeras y otros, 1992]. Aunque la matriz esencial de [H. C. Longuet-Higgins y otros, 1981] satisface una relación similar, la matriz esencial es un objeto métrico perteneciente a las cámaras calibradas, mientras que la matriz fundamental describe la correspondencia en términos más generales y fundamentales de la geometría

proyectiva, como se observa a través de la relación entre una matriz fundamental F y su correspondiente matriz esencial E:

$$E = K'^{T} F K \qquad \textit{Ecuación 16}$$

donde K y K' son las matrices de calibración (matrices de parámetros intrínsecos) de las dos imágenes involucradas.

2.5.2 Triangulación

Según el principio de triangulación, al tomar dos fotografías del mismo objeto desde al menos dos posiciones, podemos rastrear la trayectoria de los rayos de luz que van desde el centro del objetivo de cada cámara hacia un punto marcado de interés en el objeto. El punto donde estos rayos de luz se cruzan matemáticamente son coordenadas tridimensionales de ese punto de interés, y ese punto de cruce estará determinado por el centro del objetivo de cada cámara y el punto marcado en el objeto. Al repetir este proceso, se determinan los puntos que definen la geometría del objeto.

Capítulo 3.

Diseño del escáner 3D

3.1 Introducción

La primera etapa de nuestro modelo 3D será la obtención del registro fotográfico de nuestro elemento de estudio. A continuación, el conjunto de fotografías obtenidas se exportará a un software basado en fotogrametría en el campo de la visión artificial. En base al estudio realizado expuesto anteriormente en el capítulo 1, utilizaremos la herramienta *VisualSfM* [C. Wu y otros, 2010] basada en *SfM*, en la mayor parte del trabajo. Como se comentó en el capítulo 2 sobre el estado del arte, la técnica *SfM* permite realizar modelos 3D a partir de colecciones de imágenes 2D no estructuradas.

Entre las ventajas que encontramos en el uso de este software con herramientas basadas en la fotogrametría se encuentra la automatización de ciertas tareas, como es la detección de puntos característicos entre imágenes sin necesidad de establecer puntos de referencia precisos, pudiendo utilizar un gran conjunto de vistas no estructuradas para describir con el mayor detalle el objeto de estudio o la escena. Los algoritmos utilizados en este programa permiten reubicar y ordenar en el espacio tridimensional el elemento en su entorno, obteniendo como resultado la estructura tridimensional de una nube de puntos. Esta nube de puntos debe ser lo suficientemente densa para ser exportada posteriormente a *MeshLab* [P. Cignoni y otros, 2008]. Con esta herramienta procesaremos dicha nube de puntos para generar la superficie de nuestro modelo y convertirla en una geometría tridimensional, creando finalmente el modelo en 3D.

Cumpliendo con uno de los objetivos del proyecto, todas las herramientas utilizadas son *Open Source*. En la siguiente figura se presenta el desglose por etapas en la reconstrucción del modelo 3D asociada cada una a las herramientas utilizadas.

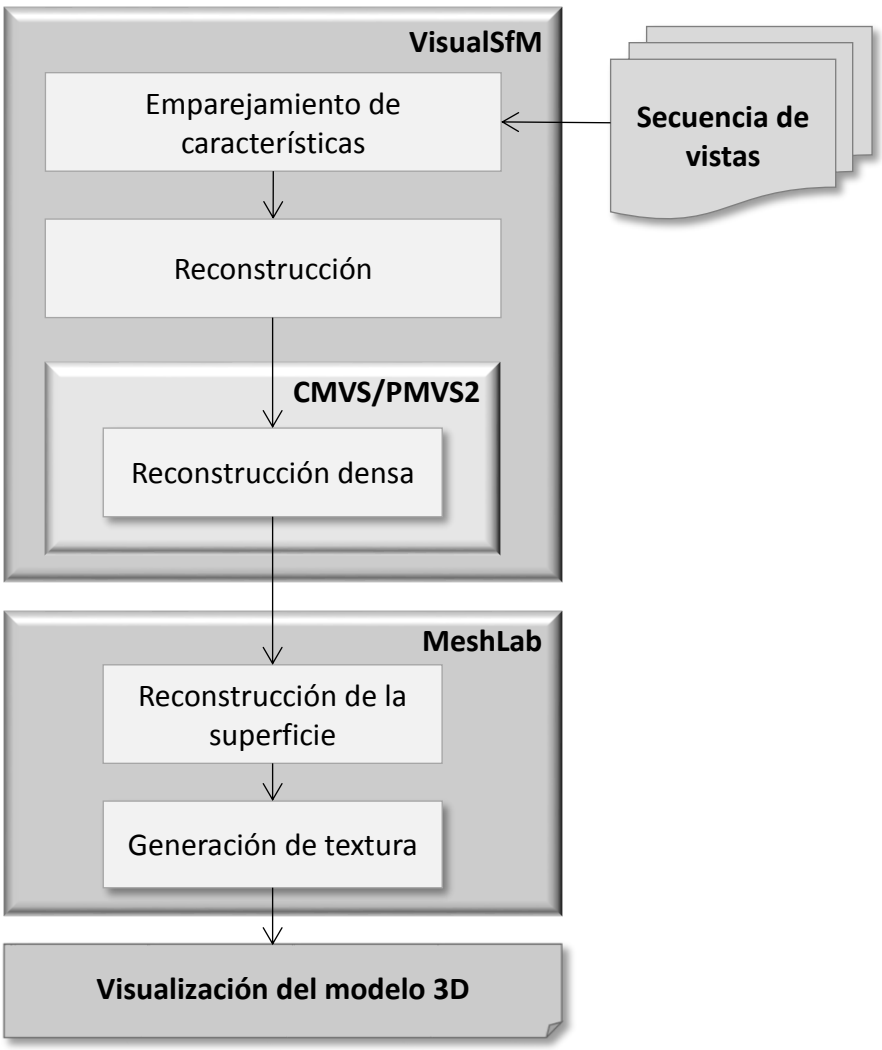

Figura 13. Etapas del diseño del escáner 3D

3.2 Secuencia de imágenes

Durante la primera etapa del proyecto, consistente en la recopilación de información acerca de la generación de modelos 3D, se valoró la opción de automatizar el proceso de captura de imágenes. Finalmente se descartó esta opción por considerar inviable crear un sistema de toma de fotografías que no creara sombras sobre la escena, afectando negativamente a la reconstrucción del modelo 3D. Si bien es cierto que actualmente existen procesos de luz estructurada o técnicas láser, éstos resultan demasiado costosos.

También se consideró el uso de una base giratoria para ubicar el objeto: de esta forma la línea de base B (distancia de la cámara al objeto) es fija y conocida de antemano, así como

el ángulo de rotación de la base. Al fijar la cámara en la misma posición y dirección, el objeto gira alrededor de un eje fijo, con una posición angular definida (Figura 15).

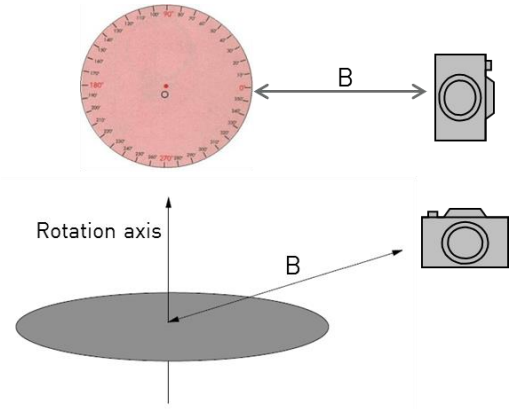

Figura 14. Plataforma giratoria

Esta opción quedó descartada por no resultar práctica a la hora de tener que monitorizar y automatizar la base, considerándose mucho más sencillo realizar la captura de imágenes por el propio usuario desplazándose alrededor del objeto.

Una de las ventajas de trabajar con *VisualSfM* es que no precisa de un conjunto de fotografías estructuradas, no es necesario seguir un patrón regular a la hora de hacer el recorrido alrededor del objeto de estudio y de su vista alzada. De esta forma la captura de imágenes resulta más sencilla y rápida, pudiendo centrarnos particularmente en obtener una mayor densidad de fotos de las áreas o zonas más críticas de nuestro modelo desde diferentes ángulos y perspectivas.

Para obtener un modelo 3D completo fijaremos el recorrido alrededor de éste en espiral, variando el ángulo de posición de la cámara entra vistas consecutivas entre 5 y 10 grados. Cuanto menor es el grado de variación entre tomas, mayor será la densidad de puntos obtenida, pero teniendo en cuenta que será necesario tomar un mayor número de fotos. Esto último se traduce en un mayor coste computacional. Un ejemplo de recorrido utilizado para los modelos estudiados en el proyecto se puede ver en la siguiente figura:

Figura 15. Recorrido seguido en la captura de imágenes.

En cuanto a la calibración de la cámara, no será necesario por tanto calcular los valores de las posiciones de las cámaras (imágenes) y la distancia de cada una respecto al objeto. La mayoría de las cámaras convencionales exportan este tipo de características físicas del momento de la captura, que se almacenan conjuntamente con el archivo imagen. *VisualSfM* emplea la información almacenada en formato EXIF, que almacena información adicional en forma de metadatos, como es la distancia focal (fc), complementando los datos del archivo imagen.

A la hora de realizar el registro de secuencias deberemos tener en cuenta las siguientes consideraciones:

- Número de vistas: necesitaremos un menor o mayor número de fotografías dependiendo del tamaño y la complejidad del objeto de estudio y del ángulo a cubrir. Se debe tener en cuenta que al aumentar el número de imágenes aumenta también el

espacio necesario para almacenarlas y el tiempo de cómputo para procesarlas, por lo que se debe llegar a una relación de compromiso en este aspecto. Entre 40 y 50 imágenes se considera suficiente.

- Iluminación: se trata de un aspecto importante, ya que las imágenes con mala iluminación, poco contraste o tomadas a contraluz pueden resultar contraproducentes, resultando en posiciones erróneas de las imágenes o en la no detección de puntos a la hora de obtener el modelo en tres dimensiones. Por ello se recomienda utilizar luz natural.
- Imágenes inadecuadas: para cada secuencia de vistas se deben seleccionar las imágenes más adecuadas y descartar aquellas que están borrosas, movidas o desenfocadas.

Los modelos que serán objeto de estudio también deberán cumplir una serie de requisitos:

- Brillo: evitar en la medida de lo posible objetos brillantes o que puedan reflejar partes del entorno
- Características: los objetos deben tener detalles claros e identificables. Si la superficie del objeto tiene un color o textura uniformes es probable que no se encuentren suficientes características entre imágenes contiguas o solapadas que sirvan de referencia a los algoritmos para entender que se trata del mismo objeto.

3.3 Emparejamiento de características y reconstrucción con VisualSfM

El método denominado *"Scale Invariant Feature Transform, SIFT"* utilizado en el caso de la técnica *SfM*, es un algoritmo para detectar y extraer vectores de características que contengan la información del entorno de diferentes puntos de interés. Además, es una herramienta útil que puede contribuir al reconocimiento de diferentes patrones de referencia en una escena determinada. Este algoritmo fue publicado por David Lowe en 1999 [D.G. Lowe y otros, 1999] y se basa principalmente en la apariencia de los objetos, específicamente en aquellos puntos clave de la imagen que son invariables a la escala de la imagen y a la rotación de los objetos.

Para llevar a cabo la identificación de los objetos en la imagen mediante el algoritmo SIFT, Lowe presenta un método cuyo cálculo consta de las siguientes etapas [D.G. Lowe y otros, 2004]:

• **Detección de extremos en el espacio-escala** (Scale-space extrema detection)

El primer paso del algoritmo se centra en localizar en toda la imagen y en todas las escalas consideradas diferentes puntos de interés que presentan una gran invarianza a la escala y a la orientación. Estos puntos se determinan usando un algoritmo de *Diferencia de Gaussianas (DoG)*. El espacio-escala de una imagen I, consiste en una familia de imágenes derivadas L_σ que se obtienen por la por la convolución de una gaussiana de desviación típica σ y escala variable G_σ, con la imagen de entrada I:

$$L_\sigma = G_\sigma * I \qquad \textit{Ecuación 17}$$

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \qquad \textit{Ecuación 18}$$

donde (*) denota la operación de convolución en las coordenadas *x* e *y*, y donde

$$G_\sigma = G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} \times e^{\frac{-(x^2+y^2)}{2\sigma^2}} \qquad \textit{Ecuación 19}$$

- **Localización de puntos característicos** (Keypoint localization)

Como resultado de la primera etapa se ubican numerosos puntos como candidatos, siendo muchos de ellos inestables. Por esta razón, esta segunda etapa realiza un modelado detallado de la escala y la orientación de cada una de las regiones, para descartar aquellos puntos que no proporcionan información importante y que son sensibles al ruido.

Para detectar puntos característicos con una ubicación estable, se utilizan los valores extremos del espacio-escala de la *Diferencia de Gaussianas* de la imagen $D(x, y, \sigma)$, tal y como se comentó anteriormente, pudiendo calcularse como la diferencia de dos escalas consecutivas separadas por un factor multiplicativo constante, k:

$$D(x, y, \sigma) = \big(G(x, y, k\sigma) - G(x, y, \sigma)\big) * I(x, y) = L(x, y, k\sigma) - L(x, y, \sigma) \qquad \textit{Ecuación 20}$$

$$D_\sigma = L_{k\sigma} - L_\sigma \qquad \textit{Ecuación 21}$$

La *Diferencia de Gaussianas* se utiliza debido a su buena aproximación al *Laplaciano de la Gaussiana normalizado a la escala* ($\sigma^2\nabla^2 G$). Debido a que este cálculo puede ser computacionalmente complejo, se realiza la aproximación a través de $D(x, y, \sigma)$.

$$\sigma\nabla^2 G = \frac{\partial G}{\partial \sigma} \approx \frac{G(x,y,k\sigma) - G(x,y,\sigma)}{k\sigma - \sigma} \qquad \textit{Ecuación 22}$$

$$G(x,y,k\sigma) - G(x,y,\sigma) \approx (k-1)\sigma^2\nabla^2 G \qquad \textit{Ecuación 23}$$

$$D_\sigma \approx (k-1)\sigma^2\nabla^2 G \times I \qquad \textit{Ecuación 24}$$

- **Asignación de la orientación** (Orientation assignment)

En la tercera etapa del cálculo del método se asigna una o más orientaciones a cada uno de los puntos característicos finales en función de las propiedades locales de la imagen. Esta asignación se basa en la magnitud y la orientación del gradiente de cada punto característico. De este modo se logra que cualquier transformación de estos valores sea completamente invariante a la rotación del objeto.

Para una imagen $L(x,y,\sigma)$, la magnitud del gradiente $m(x,y,\sigma)$, y su orientación $\theta(x,y,\sigma)$, son procesadas usando las ecuaciones:

$$m = \sqrt{\big(L(x+1,y,\sigma) - L(x-1,y,\sigma)\big)^2 + \big(L(x,y+1,\sigma) - L(x,y-1,\sigma)\big)^2} \qquad \textit{Ecuación 25}$$

$$\theta(x,y) = tan^{-1}\left(\frac{L(x,y+1,\sigma) - L(x,y-1,\sigma)}{L(x+1,y,\sigma) - L(x-1,y,\sigma)}\right) \qquad \textit{Ecuación 26}$$

- **Generación del descriptor del punto característico** (Keypoint descriptor)

En el cuarto y último paso se obtiene el descriptor de cada punto característico, que resulta de la recopilación de toda la información obtenida de una región particular de la imagen representada en un vector características. El descriptor contiene información local del gradiente de la región donde está localizado el punto característico, así como información relativa a la orientación de los píxeles vecinos.

Dos imágenes quedan definidas por los conjuntos de descriptores $\{d_i\}_{i=1}^{L}$ y $\{d_j\}_{j=1}^{N}$, donde L y N representan el número de puntos de interés. Para calcular el grado de similitud entre dos descriptores d_i y d_j se calcula la norma Euclídea entre los componentes de ambos vectores. Si la distancia es menor a 1.5 (valor por defecto de la propuesta de Lowe) se considera que los dos descriptores coinciden.

$$\left\|\{d_i\}_{i=1}^{L} - \{d_j\}_{j=1}^{N}\right\|^2 \leq 1.5 \qquad \textit{Ecuación 27}$$

Aunque la reconstrucción de la nube de puntos mediante este método nos ofrece información sobre la geometría de objeto de estudio, ésta puede ser insuficiente para evaluar con mayor detalle la superficie y la geometría, siendo necesario una segunda reconstrucción.

3.4 Reconstrucción de la nube de puntos densa con CMVS/PMVS

VisualSfM está integrado con otro tipo de herramientas de reconstrucción avanzadas como son CMVS [Y. Furukawa y otros, 2010] y PMVS2 [Y. Furukawa y otros, 2010], ambas técnicas del tipo *MultiView Stereo* (MVS) que toman como entrada el modelo *SfM*. Debido a que este tipo de técnicas obtiene las reconstrucciones gracias a las características geométricas de la visión estereoscópica, es necesario pasarles como entrada la reconstrucción previa de *SfM*. Como resultado se obtiene la reconstrucción de una nube de puntos mucho más densa y con más detalle que la obtenida de *SfM*. Estas nubes de puntos densas son un conjunto de puntos descritos en un sistema de coordenadas tridimensionales tipo XYZ, donde cada punto contiene información espacial además de una una descripción colorimétrica en el modelo RGB.

La técnica PMVS se basa en unos rectángulos llamados parches con un centro $c(p)$ en cada punto de la reconstrucción de la nube y con un vector normal $n(p)$ orientado hacia la cámara, como puede observarse en la siguiente figura [E. Alonso y otros, 2014]:

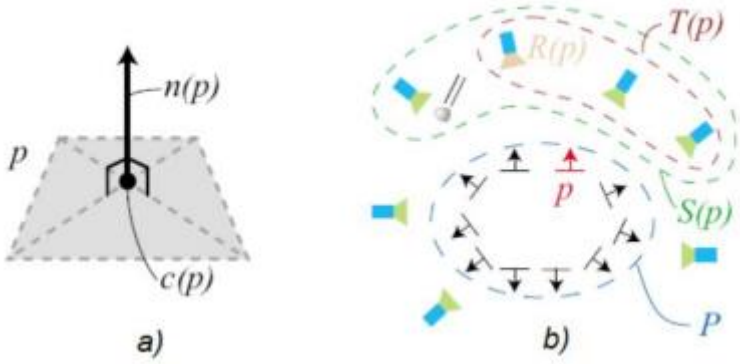

Figura 16. Representación gráfica de un parche (izquierda) y esquema gráfico del modelo

Cada parche p se asocia a la imagen $R(p)$, detectándose en las imágenes en las que se ve y añadiéndose a las imágenes desde las que debería ser visible (aunque no sea reconocible en la imagen).

Una vez se crea el primer modelo se realiza la iteración del algoritmo, añadiéndose nuevos parches a los ya creados, hasta que queda representada la superficie en su totalidad. En último lugar se realiza un filtrado para detectar y aceptar aquellos parches que se encuentran en la superficie estimada $U(p)$ y eliminado lo que se encuentran fuera de ella, como se observa en la imagen siguiente:

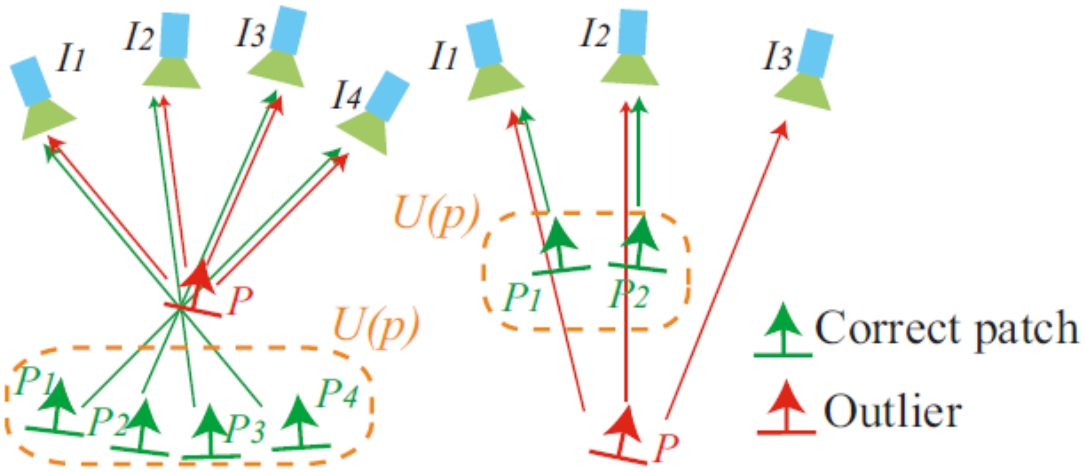

Figura 17. Filtrado de parches

3.5 Reconstrucción de la superficie y generación de textura con MeshLab

La reconstrucción de la superficie se basa en la conversión de la nube de puntos a un modelo tridimensional. Con el uso de la herramienta MeshLab realizamos la reconstrucción de la superficie por medio de mallas poligonales, aplicando la técnica conocida como triangulación. Entre las posibilidades que se ofrecen podemos destacar cuatro métodos: el Algoritmo de Pivoteo de Bola, el Algoritmo de los Cubos Móviles, la Triangulación de

Delaunay y la Reconstrucción de la Superficie por Poisson, siendo este último el método utilizado en los casos prácticos expuestos en el capítulo 5 del presente proyecto.

Uno de los primeros algoritmos de reconstrucción 3D en aparecer fue el Algoritmo de los Cubos Móviles [W. E. Lorensen y otros, 1987], convirtiéndose en una referencia para métodos más recientes. El objetivo de este algoritmo es crear modelos triangulares de densidad constante para el uso de datos médicos. El algoritmo escanea los datos con cortes y usa una función implícita para estimar los vértices de los triángulos mediante el uso de interpolación lineal. Una vez se completa la triangulación, se determina la dirección de los datos originales por medio del gradiente para completar el procesamiento del modelo.

Otro método de referencia es el Algoritmo de Pivoteo de Bola [F. Bernardini y otros, 1999], que utiliza un marco paramétrico para crear la malla resultante utilizando el enfoque de la Triangulación de Delaunay. Para un conjunto de puntos, la Triangulación de Delaunay mantiene que la circunferencia circunscrita de cada triángulo no contiene ningún vértice de otro triángulo en su interior. Partiendo de esta idea, el Algoritmo de Pivoteo de Bola hace girar una bola alrededor de los diferentes bordes de los triángulos y construye de forma iterativa la triangulación.

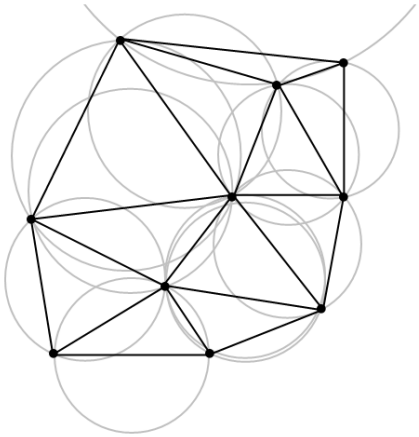

Figura 18. Triangulación de Delaunay

La Reconstrucción de la Superficie por Poisson [M. Kazhdan y otros, 2006] es un método más reciente. Este método requiere una nube de puntos orientados y calcula la superficie reconstruida utilizando una función indicador, definida como 1 dentro del modelo y 0 fuera

de la superficie. Para obtener la superficie reconstruida se extrae una isosuperficie (superficie con valor constante). El algoritmo considera todos los puntos de la nube al mismo tiempo, haciendo que la reconstrucción sea robusta frente a datos ruidosos.

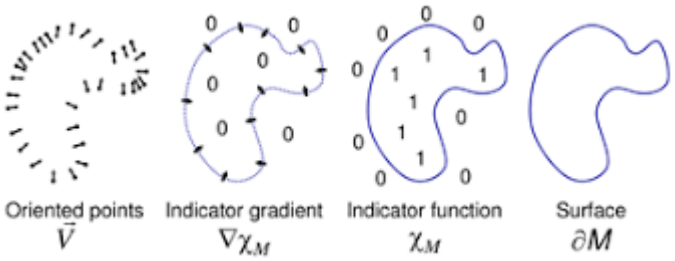

Figura 19. Reconstrucción de la Superficie por Poisson

Algunos años después los mismos autores mejoraron este mismo algoritmo [M. Kazhdan y otros, 2013] al reducir el tiempo de cálculo y crear una mayor calidad de las reconstrucciones. La diferencia entre ambos algoritmos depende de la función indicador, definida como $\frac{1}{2}$ dentro del modelo y $-\frac{1}{2}$ fuera de la superficie, por lo que la isosuperficie pasa cerca de los puntos de la nube.

Para la aplicación del algoritmo se divide la superficie en celdas tridimensionales en base a un árbol octal (octree), una estructura tipo árbol de datos donde cada nodo tiene 8 “hijos” u octantes, dividiéndose a su vez nuevamente cada nodo en 8 nuevos octantes. El nivel de profundidad del octree, se relaciona con el número de veces en que se ha dividido cada celda [I. S. Alameda y otros, 2013]. Cuanto mayor sea este valor, mayor será la exactitud de la reconstrucción.

Capítulo 4.

Evaluación de la implementación

4.1 Detalles sobre la implementación

Para la implementación del modelo de escáner se utilizaron dos tipos de cámaras fotográficas a la hora de capturar las secuencias de imágenes, modelos Canon EOS 550D y Nikon, y un ordenador personal con memoria RAM de 8GHz (memoria mínima para el uso de las herramientas seleccionadas). Las especificaciones técnicas del equipo utilizado se encuentran descritas en la sección de Pliego de Condiciones. La instalación y uso en detalle de las herramientas utilizadas se explicará en el Anexo A del presente proyecto.

El primer paso en nuestro proceso de trabajo será el de capturar la secuencia de imágenes del modelo de estudio y cargarla en la aplicación *VisualSfM.* Seguidamente debe realizarse el emparejamiento de imágenes, con el fin de detectar los puntos comunes entre las diversas fotografías y encontrar las coincidencias entre pares de vistas. Una vez realizado el emparejamiento, se realizará la primera reconstrucción de la nube de puntos. Si no se obtuviese un resultado óptimo, será necesario volver a comenzar con el primer paso del proceso y obtener una nueva secuencia de vistas. Si el resultado es aceptable, continuaríamos con el cálculo de una segunda reconstrucción 3D densa, utilizando el algoritmo CMVS/PMVS.

Una vez realizada la reconstrucción de la nube de puntos con *VisualSfM*, pasaremos a exportarla a *MeshLab*. Es probable que en el modelo aparezcan puntos o vértices no deseados, por no corresponderse con el objeto de estudio sino con su entorno o con el fondo de la escena y son considerados como ruido. En nuestro modelo de escáner el usuario es el encargado de identificar dichos puntos que no son necesarios para eliminarlos manualmente. Una vez hecho esto importaremos el mallado que nos proporciona también *VisualSfM.* Al igual que con la nube de puntos será necesario eliminar áreas de la malla que no son necesarias.

Con el fin de obtener nuestro modelo 3D necesitamos una nube de puntos mucho más densa de la proporcionada por la reconstrucción SfM. La reconstrucción por *Poisson* es el método de triangulación elegido en *MeshLab* para calcular la reconstrucción de la superficie. Una vez calculada la triangulación, será necesario eliminar caras redundantes o erróneas. SI fuera necesario reducir el número de triángulos o vértices, se aplicará un algoritmo de optimización incremental para simplificar la geometría.

Finalmente, el último paso de nuestro proceso de trabajo es la parametrización de la superficie y la creación de la textura.

A continuación, se muestra el diagrama de flujo de trabajo que resume la metodología de trabajo aplicada a los casos prácticos que presentamos en la siguiente sección.

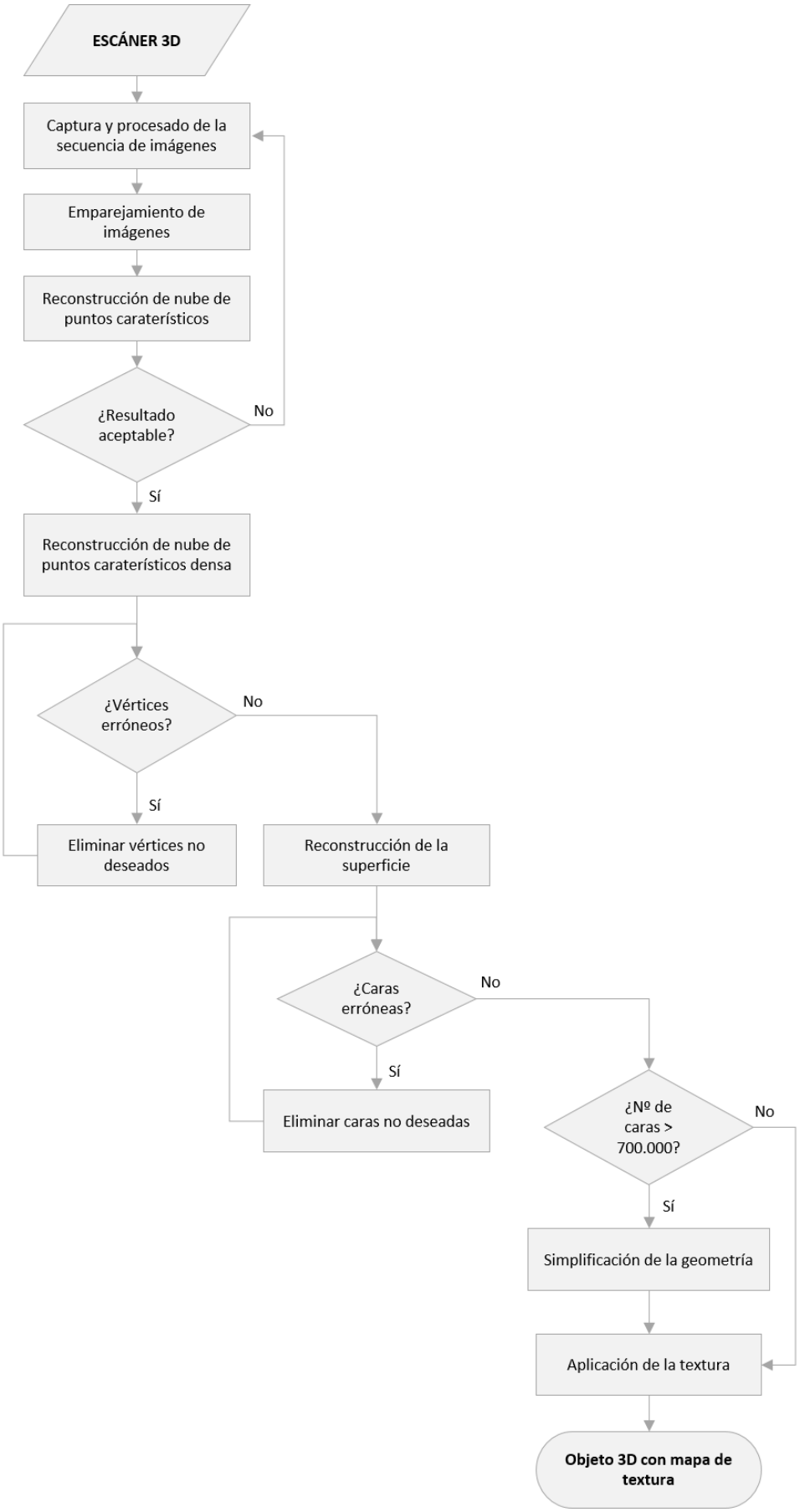

Figura 20. Diagrama de flujo del trabajo a implementar

4.2 Caso práctico 1

4.2.1 Descripción y captura de imágenes

En el primer test de pruebas el modelo elegido es un maniquí, conocido también como dummy. El set consta de 38 fotografías en alta resolución (3000 x 4000 píxeles) realizadas alrededor del objeto.

Figura 21. Miniaturas de la secuencia de imágenes del maniquí importadas a VisualSfM

4.2.2 Reconstrucción de la nube de puntos

Una vez cargadas las imágenes en la aplicación *VisualSfM*, el siguiente paso será el emparejamiento de las imágenes. Aunque el número de fotografías no es elevado y algunas de las imágenes no están centradas en el maniquí, el número de emparejamientos resultante (703 pares) es bastante elevado.

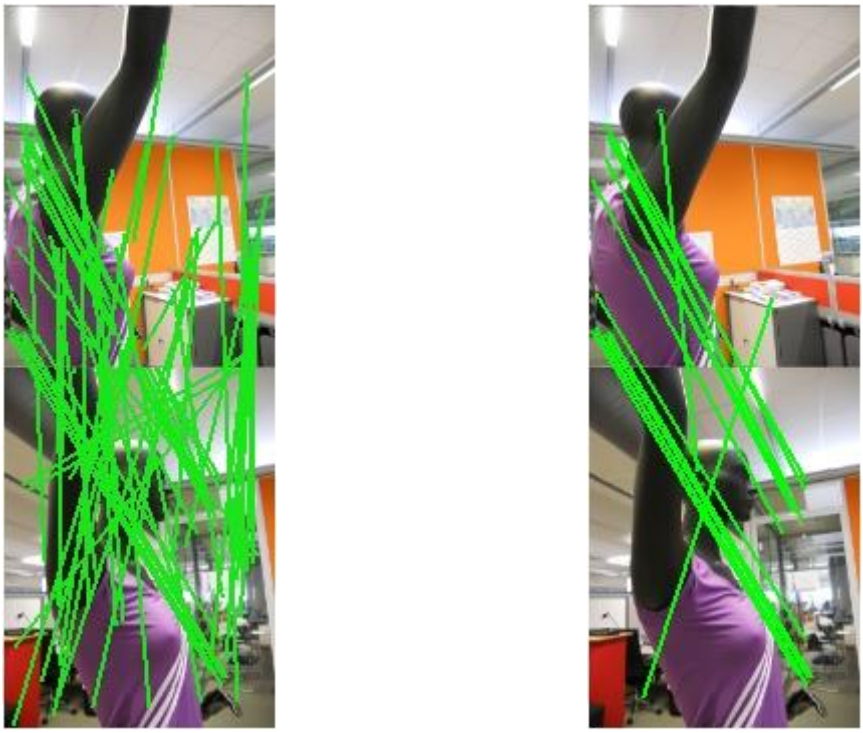

Figura 22. Ejemplo de detección de puntos característicos en un par de imágenes (izquierda) y emparejamientos o inliers en el mismo par (derecha)

A continuación, se realiza la primera reconstrucción 3D (Figura 22) en la que se aplican los procedimientos descritos anteriormente en la sección 3.3.1. y de la que se obtienen 1490 puntos característicos.

Figura 23. Primera reconstrucción de puntos 3D

Por último, se realiza la reconstrucción densa. Después de aplicar el algoritmo CMVS/PMVS se calculan 234.534 vértices. Como podemos observar en la Figura 23, existen vacíos importantes en nuestro modelo, que cuenta con unas zonas más densas que otras. Debido al tamaño del objeto de estudio (maniquí) encontramos mucha información no deseada del fondo de la escena, considerada como ruido, que procederemos a eliminar en la siguiente etapa.

Figura 24. Primeros resultados obtenidos con VisualSfM

4.2.3 Reconstrucción de la superficie

Una vez realizada la importación de la nube de puntos y la malla obtenidas con *VisualSfM* y después de realizar la limpieza de puntos y vértices, realizaremos la reconstrucción de la superficie por medio del filtro de *Poisson*. Como no se trata de una nube de puntos muy densa, se ha optado por seleccionar el valor máximo de profundidad (*Reconstruction Depth*), que suele estar comprendido entre 5 y 10.

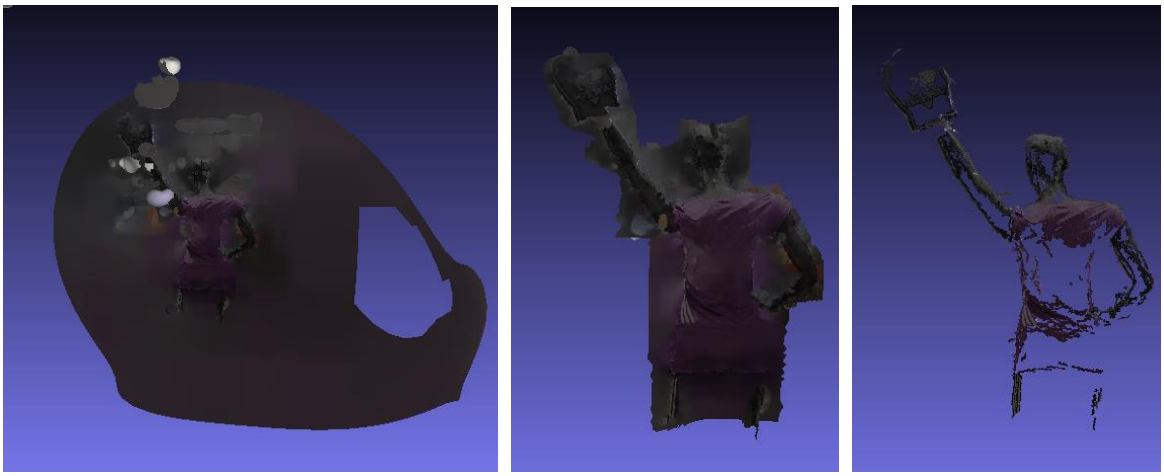

Figura 25. Resultados tras la aplicación del filtro (izquierda), después de eliminar las caras del mallado no deseadas (centro) y nubes de puntos útiles tras la reconstrucción.

Una vez realizada la triangulación obtenemos 138.553 caras (con un número de puntos útiles de 262.264) que supone un número bajo, así que no será necesario realizar ningún tipo de optimización. La reconstrucción de este modelo no es posible, debido a que se han obtenido un número muy limitado de vértices útiles de la nube de puntos tridimensional y por tanto no se tiene la suficiente información para representar el volumen del objeto.

Figura 26. Resultado de la triangulación (izquierda) y del modelo 3D tras la aplicación de la textura (derecha)

4.3 Caso práctico 2

4.3.1 Descripción y captura de imágenes

En el segundo test de pruebas el modelo elegido es un zapato deportivo. El set consta de 24 fotografías en alta resolución (5184x3456 píxeles) realizadas alrededor del objeto.

Figura 27. Miniaturas de la secuencia de imágenes del zapato importadas a VisualSfM

4.3.2 Reconstrucción de la nube de puntos

Al igual que en el caso práctico anterior, comenzaremos el procedimiento realizando el emparejamiento de imágenes, para detectar las características comunes entre pares de fotos (*matches*) y sus coincidencias (*inliers*).

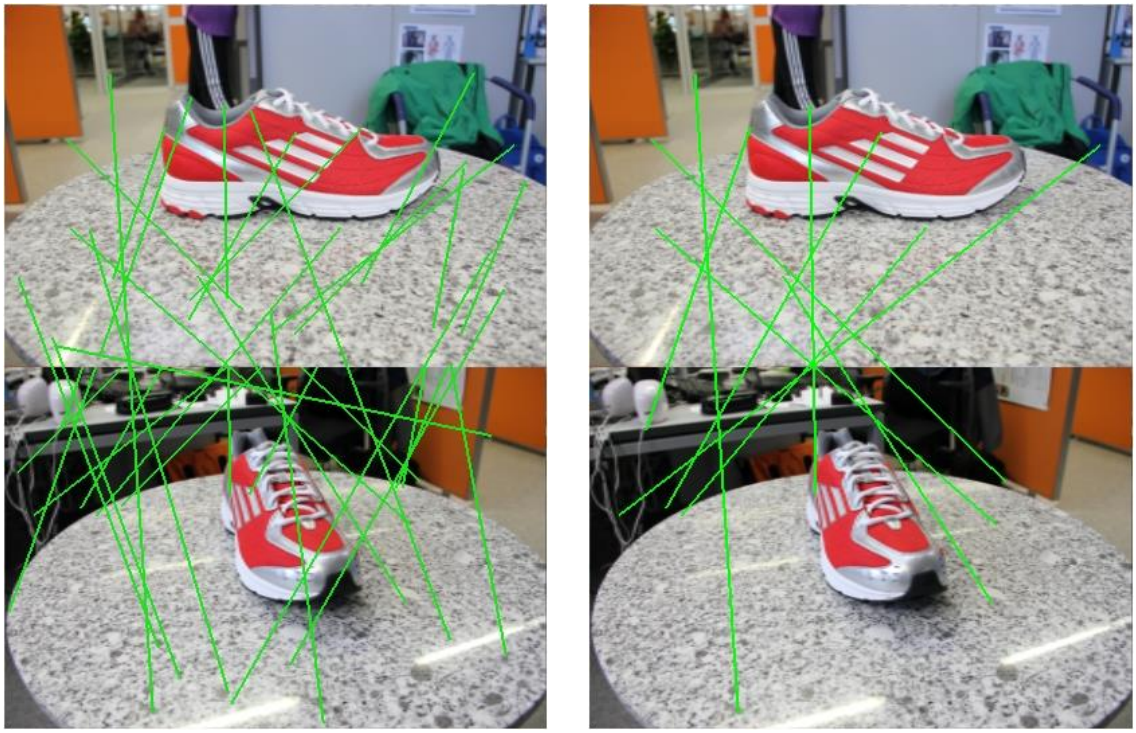

Figura 28. Ejemplo de detección de puntos característicos en un par de imágenes (izquierda) y emparejamientos o inliers en el mismo par (derecha)

En este caso el número de pares calculado es de 276. En la siguiente figura podemos observar el resultado de la primera reconstrucción de la nube de puntos general vista a partir de la secuencia completa de imágenes y la nube de puntos vistos por más de 3 cámaras, es decir, puntos comunes que encontramos en más de 3 imágenes. En este proceso da como resultado el cálculo de 12.810 puntos.

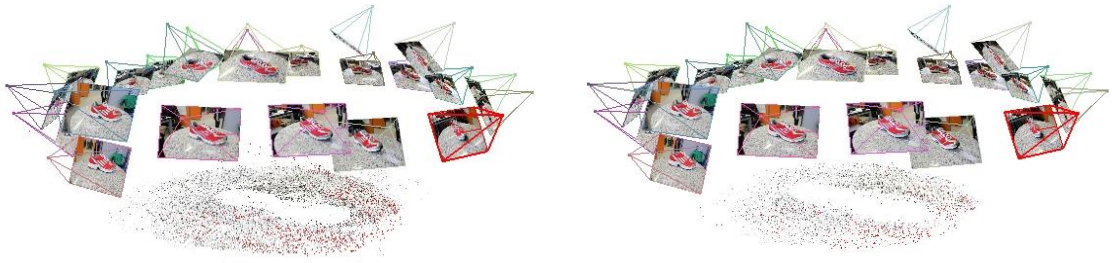

Figura 29. Reconstrucción de la nube de puntos vistos por todas las cámaras (izquierda) y puntos vistos por más de 3 cámaras

Tal y como se detalló en el diagrama de flujo de trabajo del apartado 4.1, el siguiente paso es la reconstrucción 3D densa, basada en el algoritmo CMVS/PMVS. Dependiendo del número de fotos y de la calidad, será necesario un mayor tiempo durante el proceso de cálculo. Para este caso práctico se obtuvo la nube densa que se presenta en la Figura 31, con 1.430.761 vértices calculados. Al igual que en el caso práctico anterior de la escena del maniquí, se obtienen puntos cercanos al objeto no deseados, que procederemos a eliminar a continuación con *MeshLab,* una vez hayamos exportado la nube de puntos.

Figura 30. Resultado de la reconstrucción densa de la nube de puntos

4.3.3 Reconstrucción de la superficie

Una vez realizada la importación de la nube de puntos a *MeshLab,* procederemos a eliminar los puntos indeseables considerados como ruido, utilizando las opciones disponibles de la herramienta. Una vez importada la malla, realizaremos el mismo proceso de limpieza. A continuación, se presentan imágenes de ambos procesos y del resultado final. En ambos casos se decide no descartar la base en la que se posa el objeto, para no sufrir pérdidas importantes de información de la zona inferior del zapato.

Si en el paso anterior obteníamos como resultado una nube de 12.810 puntos y una malla densa de 1.430.761 vértices, después de este proceso de limpieza nuestro modelo contará con 12.397 puntos y 1.418.364 vértices. En la imagen del resultado final observamos como aparecen algunas zonas con menos densidad que otras.

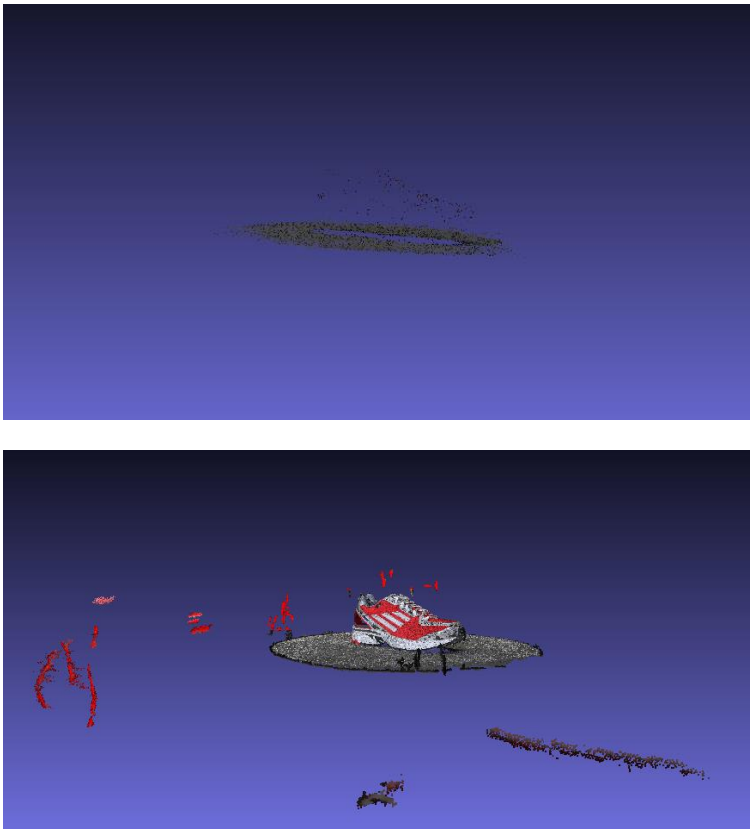

Figura 31. Importación de la nube de puntos (arriba) y malla (abajo) a MeshLab y eliminación de vértices indeseados

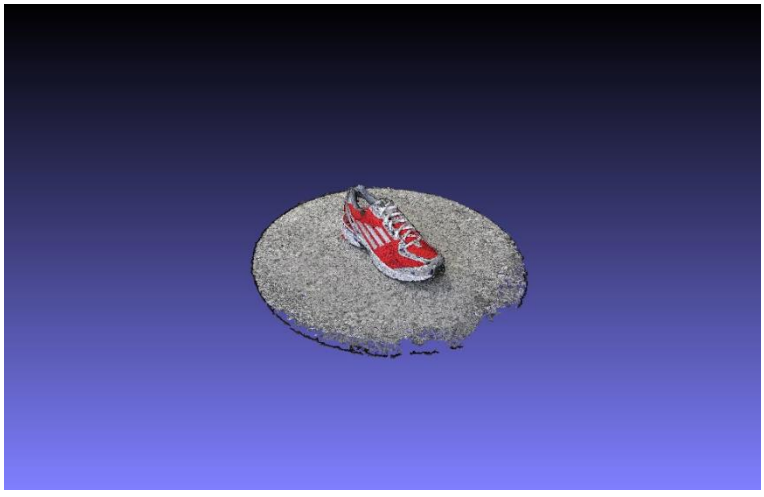

Figura 32. Resultado de la limpieza de vértices del mallado

El siguiente paso del proceso es la reconstrucción superficial para transformar la nube de puntos en un modelo tridimensional. En este caso optamos por seleccionar los valores por defecto de la reconstrucción de *Poisson*, con un valor máximo de profundidad de 8.

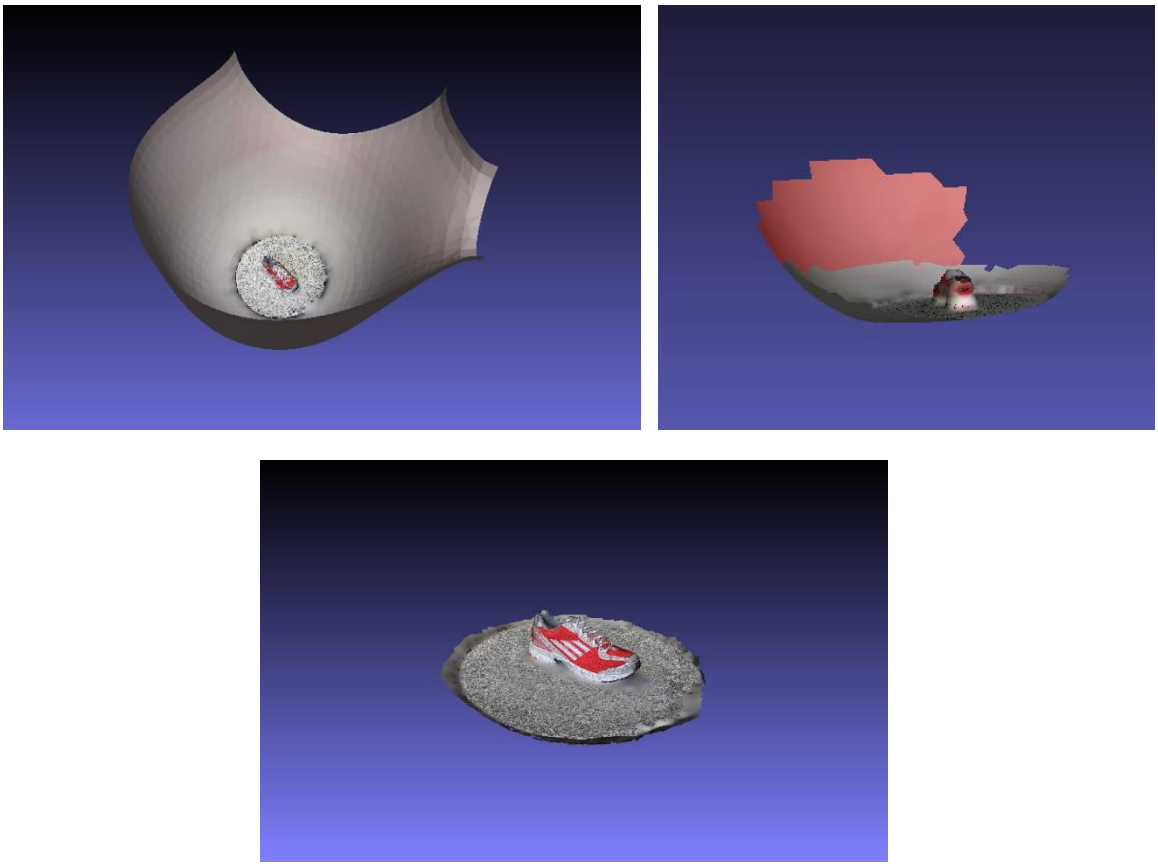

Figura 33. Resultado de la reconstrucción de Poisson

En la figura anterior se observa que el resultado es un modelo más continuo sin vacíos, ya que hemos aplicado un método de triangulación que se adapta bien a superficies irregulares. En la primera aproximación se han obtenido 18.806 caras que se corresponden con 1.441.465 vértices. Después del proceso de eliminación y limpieza, se obtiene como resultado final 14.773 caras, que se corresponden con 10.704 vértices, reduciéndose considerablemente su número.

Por último, se aplicó la textura al modelo. Como observamos a continuación en las imágenes del resultado final, el método proporciona la suficiente información del color, obteniendo un mapa de textura de 1024*1024 píxeles que se aplica correctamente al modelo 3D.

Figura 34. Resultado final del modelo 3D (arriba) con detalle de la triangulación (centro) y de su mapa de textura (abajo)

4.4 Caso práctico 3

4.4.1 Descripción y captura de imágenes

Al igual que para el segundo test de pruebas, en el último set de fotos el modelo elegido es un zapato deportivo. El set consta de 79 fotografías en alta resolución (5184 x 3456 píxeles) realizadas alrededor del objeto.

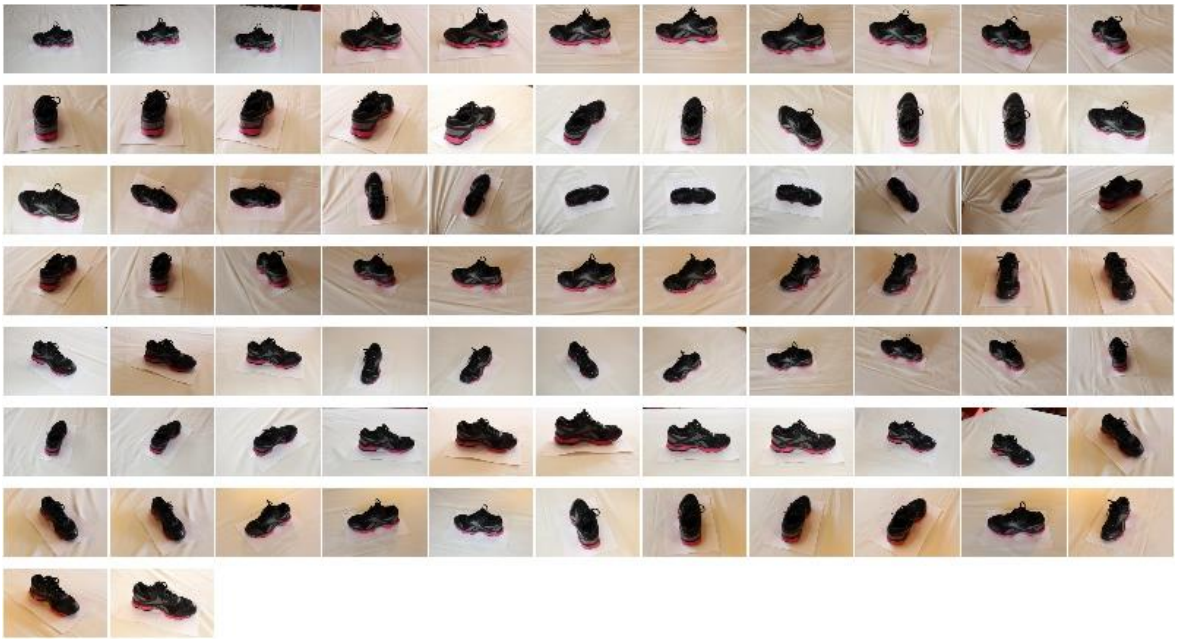

Figura 35. Miniaturas de la secuencia de imágenes del zapato importadas a VisualSfM

Con el fin de evitar el ruido que introduce la información no deseada del entorno y poder eliminar con mayor facilidad la base del objeto al contrario del caso práctico anterior, se ha situado el zapato sobre una superficie plana y blanca.

4.4.2 Reconstrucción de la nube de puntos

En cuanto al emparejamiento de imágenes, en este caso se obtuvieron 3.081 pares, un número elevado comparado con el caso práctico anterior. Esto es debido al incremento en el número de fotografías, ya que esta secuencia de vistas cuenta con más del triple de la

anterior. De esta forma se pueden realizar más emparejamientos por pares de fotografías sin que esto requiera un mayo tiempo de cómputo.

El siguiente paso es el de la reconstrucción de la nube 3D, realizada en 2 fases. En la siguiente figura se muestra el resultado obtenido para la nube de puntos tras la primera reconstrucción, con un número total de 11.449 puntos característicos y a continuación se presenta el resultado tras la reconstrucción densa, con un total de 571.302 vértices calculados.

Figura 36. Nube de puntos tras la primera reconstrucción (arriba) y tras la reconstrucción densa (abajo).

Llegados a este punto y en comparación con el caso anterior, podemos advertir que un mayor número de emparejamientos no implica un mayor número de puntos que conformen una nube tridimensional más densa.

4.4.3 Reconstrucción de la superficie

Una vez importado nuestro modelo y malla a *MeshLab*, eliminaremos manualmente los vértices no deseados. En la siguiente figura se muestran algunos ejemplos de este proceso (los vértices no deseados aparecen seleccionados en color rojo antes de ser eliminados).

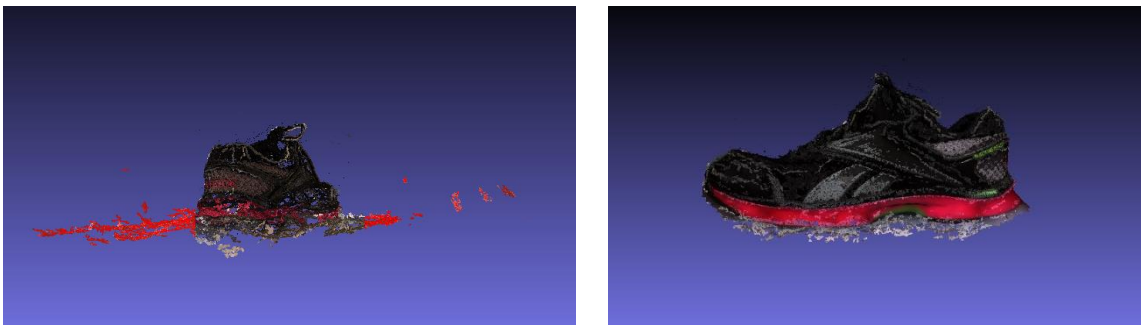

Figura 37. Importación de la malla y eliminación de vértices erróneos

En cuanto a la reconstrucción de la superficie por *Poisson*, se utilizaron los mismos valores por defecto para el cálculo de la triangulación. En este caso práctico se obtuvieron algunos problemas a la hora de eliminar las caras más cercanas a la base del zapato, debido a la forma cóncava de la malla de triangulación, hecho que quedará en evidencia al aplicar la textura al zapato para obtener el modelo 3D final.

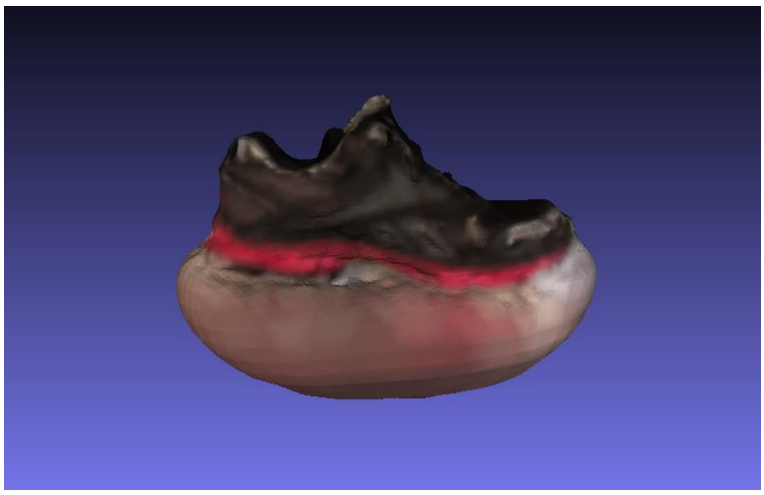

Figura 38. Resultado de la reconstrucción de Poisson

Como podemos observar en la siguiente figura del modelo 3D final, uno de los laterales de la base del zapato queda incompleta, al haber eliminado parte de la malla de triangulación en el paso anterior. A pesar de que ciertas zonas del zapato resultan más oscuras debido al poco contraste de algunas de las imágenes originales, el mapa de textura se aplica correctamente al modelo 3D, con una resolución de 1024*1024 píxeles.

Figura 39. Resultado final del modelo 3D y de su mapa de textura

4.4.4 Simulación con imágenes preprocesadas

Con el fin de evitar un modelo 3D con zonas poco iluminadas, se realizó una segunda prueba con la secuencia de imágenes presentadas anteriormente, preprocesando previamente las imágenes mediante un ajuste automático de los niveles de intensidad. Para ello desarrollamos una función de preprocesamiento, desarrollada en el programa y lenguaje de programación GNU Octave [GNU Octave, 2018], considerado el equivalente libre de Matlab.

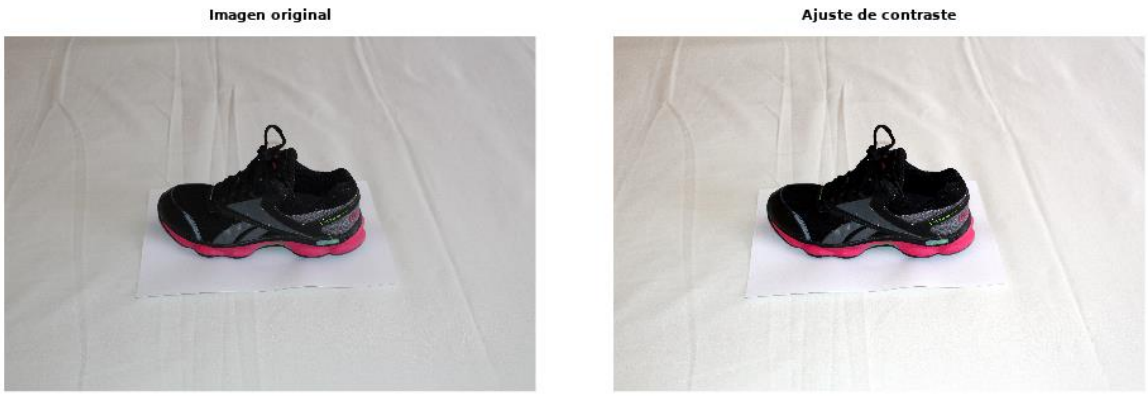

Figura 40. Ejemplo de preprocesamiento de la imagen por ajuste de contraste

Aunque en este caso se obtuvo el mismo número de emparejamientos (se utilizó la misma secuencia de vistas), sí se obtuvo un ligero incremento en el número de puntos característicos para ambas reconstrucciones de la nube de puntos tridimensional.

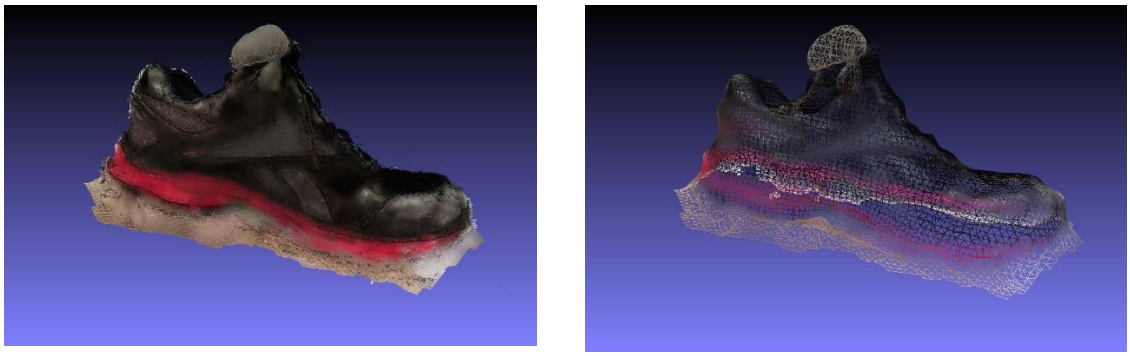

Figura 41. Vistas en detalle tras la aplicación de la reconstrucción de Poisson

Además, una vez realizada la reconstrucción de la superficie por *Poisson*, en esta simulación se ha evitado eliminar las caras de la malla de triangulación más cercanas a la base del objeto, con el fin de evitar una posible pérdida de información a la hora de aplicar la textura. El resultado final del modelo 3D presenta zonas de mayor brillo y luminosidad, debido al preprocesamiento de las imágenes.

Figura 42. Resultado final del modelo 3D (arriba) con detalle de la triangulación (centro) y de su mapa de textura (abajo)

4.5 Comparativa de resultados

En este apartado se presenta una tabla comparativa en la que se muestra, para cada caso de uso, los siguientes datos más característicos obtenidos durante el proceso de implementación del escáner 3D:

- Modelo de cámara utilizada para la captura de imágenes
- Número de imágenes empleadas para la reconstrucción del modelo.
- Resolución de las imágenes
- Tiempo de carga de las imágenes empleadas en la herramienta *VisualSfM*
- Número total de emparejamientos encontrados tras la ejecución del algoritmo de emparejamiento de pares de imágenes en *VisualSfM* y tiempo de cálculo asociado a este proceso
- Puntos característicos calculados en la primera reconstrucción en *VisualSfM* y tiempo de cálculo asociado a este proceso
- Puntos de la reconstrucción de la nube 3D densa en *VisualSfM* y tiempo de cálculo asociado a este proceso
- Puntos útiles y nº de caras tras el cálculo de la triangulación y la reconstrucción por Poisson en *MeshLab*
- Tiempo total resultante de la aproximación del tiempo empleado para todo el proceso de modelado. No se tienen en cuenta los tiempos de cómputo de *MeshLab* por ser del orden de milisegundos y por lo tanto despreciable frente al resto de los tiempos.

Como observamos a partir de los resultados obtenidos, el tiempo de cómputo de la digitalización del modelo 3D dependerá sobre todo del número de imágenes capturadas y procesadas.

	DATASET 1	DATASET 2	DATASET 3 1er test	DATASET 3 2º test
Cámara	Nikon	Canon EOS 550D	Canon EOS 550D	Canon EOS 550D
Nº de imágenes	38	24	79	79
Tamaño imágenes	3000x4000 px	5184x3456 px	5184x3456 px	5184x3456 px
Tiempo de carga	18 s	20 s	61 s	48 s
Emparejamientos	703 pares	276 pares	3.081 pares	3.081 pares
	78 s	188 s	180 s	183 s
Puntos característicos	1.490	12.810	11.449	11.558
	83 s	33 s	39 s	36 s
Nube de puntos densa (vértices)	234.534	1.430.761	581.302	599.247
	9,38 min	30,76 min	36,35 min	47,23 min
Puntos útiles	262.264	10.704	13.883	13.028
Nº de caras	138.553	14.773	23.258	21.483
TIEMPO TOTAL	11,36 min	34,77 min	41,01 min	51,68 min

Tabla 2. Tabla comparativa de resultados

Capítulo 5.

Conclusiones y líneas futuras

5.1 Conclusiones

En este Proyecto Fin de carrera se ha aplicado una metodología para el tratamiento de nubes de puntos característicos obtenidos a partir de fotografías y la posterior creación de modelos tridimensionales a partir de las mismas.

Como se observa a raíz de las pruebas de simulación realizadas y los resultados obtenidos se han encontrado algunas limitaciones a la hora de trabajar con la herramienta *VisualSfM*. En líneas generales, de los resultados obtenidos podemos afirmar que la herramienta funcionará mejor en objetos muy texturizados, ya que en superficies con color y texturas uniformes no dará buenos resultados. Además, no se obtendrán buenos resultados si se detectan muy pocos puntos característicos.

La captura de la secuencia de vistas de la primera implementación se realizó durante la estancia en las oficinas de la empresa. Al ser el primer test no se tuvieron en cuenta los requisitos en cuanto al número total de imágenes y al registro desde diferentes perspectivas y ángulos, por lo que no se tiene suficiente información del objeto ni con el nivel de detalle requerido. Los resultados de esta primera simulación son bastante pobres, como cabría esperar, ya que se obtienen bastantes puntos en el espacio tridimensional que no se corresponden con el objeto fotografiado y el modelo no tiene la densidad de puntos necesaria para poder representar su volumen tridimensional.

En la segunda implementación, encontramos el inconveniente de la base en la que se encuentra el objeto, y por esta razón se decidió utilizar una superficie blanca plana en la tercera implementación. Comparando ambos casos, obtendremos un mayor número de emparejamientos al utilizar una secuencia con un número elevado de vistas. En cambio, tal y como quedó reflejado en la Tabla 2 de resultados, un mayor número de emparejamientos no implica un mayor número de puntos que conformen una nube tridimensional más densa, pero sí tendrá influencia en el número total de vértices y caras del modelo final. En

cuanto a la texturización, la elección en ambos casos de un mapa de textura de 1K de resolución (1024*1024 píxeles) proporciona un resultado adecuado.

Por otro lado, en el caso de la última implementación con imágenes procesadas, aunque un preprocesamiento previo de las imágenes por ajuste de contraste no interfiere en los valores calculados para el número total de vértices o caras de la malla de triangulación, sí se verá reflejado en la visualización del modelo 3D final, aportando zonas de mayor brillo y luminosidad. En este último caso, a pesar de que los resultados del modelo son aceptables, no se logra una alta precisión en el modelo 3D final debido a la introducción de ruido del entorno, en concreto de la base en la que se posa el objeto.

Una desventaja de nuestra implementación de escáner 3D reside en la eliminación de ruido del modelo, ya que descartar puntos y caras manualmente se convierte en una tarea bastante tediosa y complicada cuando se tratan zonas circulares, como puede ser la suela de un zapato.

Las principales ventajas de la implementación de este modelo de escáner 3D con respecto a otro tipo de dispositivos de escaneado tradicionales son la rapidez y la calidad en la extracción y procesado de las nubes de puntos, así como en la creación del modelo 3D. Además, el uso de herramientas *Open Source* supone un coste cero en licencias, frente al elevado coste que suponen los escáneres láser y otros dispositivos que podemos encontrar actualmente en el mercado.

5.2 Líneas de trabajo futuras

- Una posible línea de trabajo en la que podríamos centrar nuevas simulaciones estaría orientada hacia la obtención del objeto de estudio sin ninguna información del entorno, es decir, aplicando previamente técnicas de segmentación para eliminar el fondo de las imágenes.
- Otra posible línea de trabajo sería la mejora de la visualización del modelo final, pudiendo importar nuestro modelo texturizado (.obj) a otro tipo de herramientas de computación gráfica que aplican algoritmos de renderización. La renderización es la técnica que realiza el acabado final del modelo 3D de una forma detallada. Se trata de

un proceso en el que se agregan una serie de efectos como pueden ser sombras, transparencias, texturas, colores, tonalidades, reflejos, profundidades de campo, etc. *Blender* [Blender, 2018] es una herramienta Open Source que cumple con estas premisas, ya que permite crear una escena a partir de un objeto 3D otorgándole un mayor realismo aplicando este tipo de técnicas y efectos.

- Por otro lado, gracias a las nuevas impresoras 3D de software libre que existen actualmente, el siguiente paso a seguir para el caso de la obtención de modelos 3D de piezas de calzado, sería el de construir un procedimiento de prototipado 3D a partir de los modelos obtenidos.

Anexo A. Instalación y uso de software

A.1 Preprocesamiento de imágenes

Función preprocessing.m desarrollada en GNU Octave:

```
% preprocessing – Octave function for image processing that includes contrast adjustment
%                        and compression by a factor of k
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% ENTRADA:
%     directory -> (1x1) string folder that contains the images
% maxFrames -> (1x1) number of images to process
%                 k -> (1x1) factor de compresión
% SALIDA:
%            Files -> (1,1) struct that contains the images processed
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [Files] = preprocessing(directory, maxFrames, k)
files = dir(directory);
files = files(3:end);
mkdir(strcat(directory,"imagenesprocesadas/"));

% Process all the images
for i=1:1:min(size(files,1),maxFrames)

          % Read filename
          img1 = files(i).name;
          img1 = strcat(directory,img1);
          img2 = strcat(files(i).name);
          img2 = strcat(directory,"processed/",img2);

 im = imread(img1);
 %Compression by a factor of k
 im = imresize(IM, k, 'linear');

 % Contrast adjustment of the RGB image
 RGB1 = imadjust(im);
 % Contrast adjustment of the RGB image, specifying contrast limits
 % imadjust (I, [low_in; high_in],[low_out; high_out])
 % RGB2 = imadjust(im,[.2 .3 0; .6 .7 1],[]);

  figure ();
  subplot (1,2,1),imshow(im);title('Original image','Fontweight','bold');
  subplot (1,2,2),imshow(RGB1);title('Processed image','Fontweight','bold');

  % Save image
  fprintf('Image: %s processed\n', img1);
```

```
  imwrite(RGB1,img2);
    % Memory clean
  clear img1 img2 im RGB1;

end;

disp("Image processing finalized.\n")
Files = struct;
Files.files = files;
Files.dir = directory;
```

A.2 VisualSfM y CMVS/PMVS2

A.2.1 Instalación

Para instalar *VisualSfM* bastará con acceder a la página oficial de la herramienta [C. Wu y otros, 2010] y seleccionar el tipo de sistema operativo empleado para descargarnos automáticamente el ejecutable correspondiente. Continuaremos con la instalación de las librerías CMVS/PMVS2. Al igual que en el paso anterior, accederemos a la página oficial [Y. Furukawa y otros, 2010] para descargar dichas librerías, que deberán colocarse en la misma carpeta anterior en la que instalamos el programa *VisualSfM.* Si no realizamos este paso no nos aparecerá la opción de "Reconstrucción densa (CMVS)" en *VisualSfM.*

A.2.2 Manual de uso

El uso de VisualSfM en combinación con CMVS/PMVS2 es bastante sencillo, realizándose la reconstrucción de la nube de puntos densa en tan sólo 4 pasos, tal y como se detalla a continuación:

Figura 43. Barra de herramientas de VisualSfM [C. Wu y otros, 2010]

1) **Añadir imágenes.** En primer lugar, se añade la secuencia de vistas tomadas alrededor de un objeto o escena, teniendo en cuenta que las imágenes de alta resolución necesitarán un mayor tiempo de cómputo para ser cargadas.
2) **Emparejar las imágenes.** A continuación, se realiza la búsqueda y el emparejamiento de imágenes. Pulsando sobre este botón (SfM > Pairwise Matching > Compute Missing Matches) se realiza automáticamente el emparejamiento de las imágenes previamente cargadas. También existe la posibilidad de especificar una lista propia de puntos característicos que queramos importar (SfM > Pairwise Matching > Import Feature Matches) utilizando el siguiente formato en un archivo .sift: [Header][Location Data][Descriptor Data][EOF]

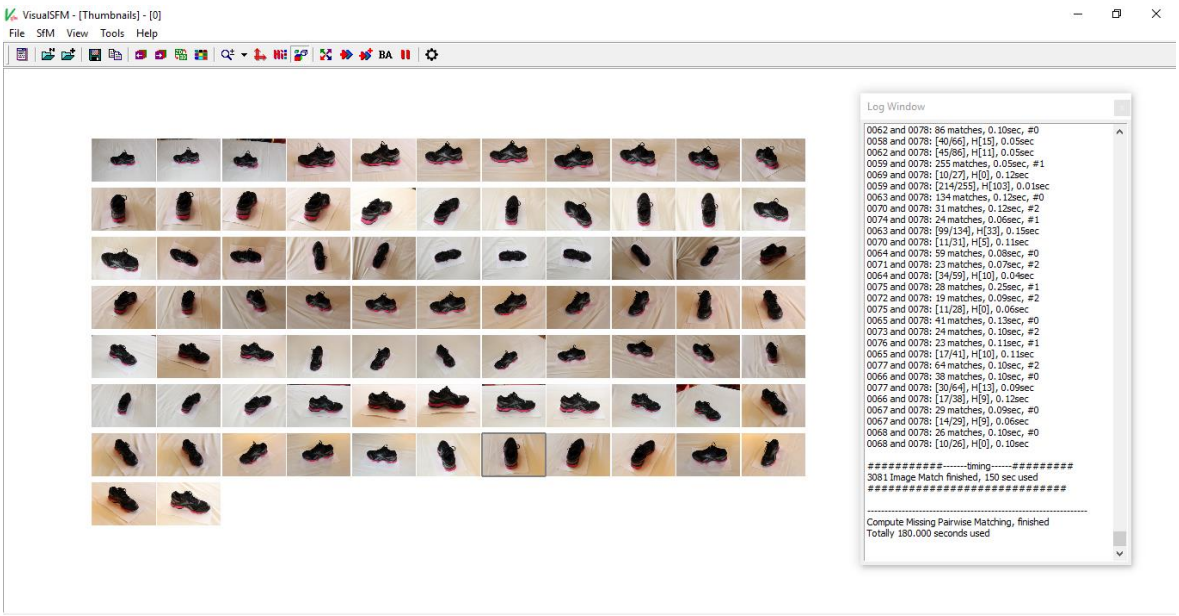

Figura 44. Ejemplo del proceso de emparejamiento de imágenes finalizado.

3) **Reconstrucción dispersa.** Una vez finalizado el proceso de emparejamiento, podemos visualizar ejemplos de características (View > Feature Matches) y emparejamientos (View > Inlier Matches) marcados sobre un par de fotografías. A continuación, se realiza la primera reconstrucción 3D. En este paso los puntos característicos emparejados previamente se calculan respecto a un sistema de coordenadas, pudiendo ser visualizados (View > N-View 3D Points).
4) **Reconstrucción densa.** Finalmente se calcula la reconstrucción densa (CMVS), reconstruyendo la textura. En este punto debemos indicar la ruta para guardar el

proyecto en una carpeta con formato “nombre.nvm.cmvs” y dos archivos en formato “nombre.nvm” y “nombre.0.ply”. El tiempo de cómputo de esta reconstrucción es superior al de la anterior, del orden de minutos o incluso horas, dependiendo del número de fotos, la resolución y de la capacidad del ordenador. Al finalizar, podemos observar el resultado final de nuestro modelo 3D (View > Dense 3D Points).

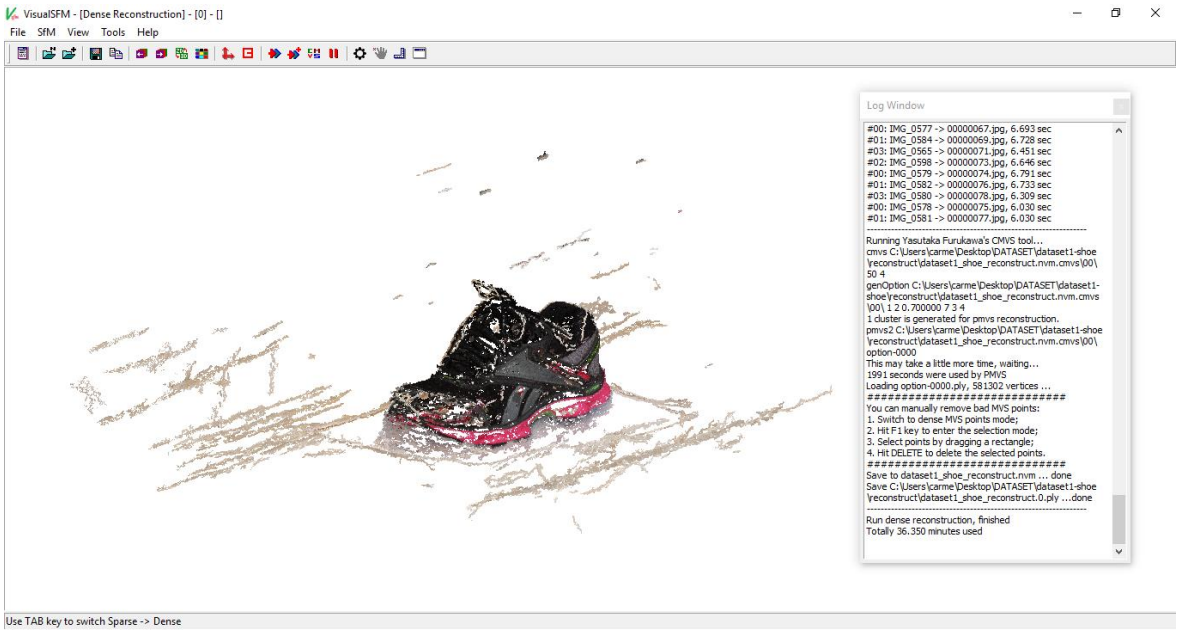

Figura 45. Ejemplo de modelo 3D tras finalizar el proceso de reconstrucción densa (CMVS)

Como se puede observar obtenemos una reconstrucción del modelo 3D con mucho ruido entorno al objeto que procede del fondo de las imágenes, siendo necesaria la exportación a *MeshLab* para limpiar el modelo y obtener una mejor recreación del mismo.

A.3 MeshLab

A.3.1 Instalación

Para instalar *MeshLab* se debe acceder a la página oficial de la herramienta [MeshLab, 2016]. Una vez seleccionado el tipo de sistema operativo empleado, el archivo ejecutable se descargará automáticamente en nuestro equipo.

A.3.2 Manual de uso

A continuación, se detallan los pasos a seguir para optimizar el modelo 3D importado:

1) **Importación de la nube de puntos y malla.** En primer lugar, abriremos la carpeta de nuestro proyecto (File > Open Project) y seleccionaremos el archivo "nombre.nvm" proporcionado por *VisualSfM.* En la parte inferior de la pantalla y sobre un panel de color morado, podemos observar las características del modelo:

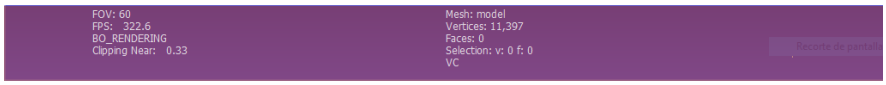

Figura 46. Características del modelo 3D

En la parte izquierda se encuentran los siguientes valores:

- FOV (Field Of View): ángulo de visión
- FPS (Frames Per Second): número de de fotogramas por segundo al trasladar la nube.

En la parte derecha del panel encontramos:

- Mesh: nombre del fichero
- Vertices: número de vértices o puntos
- Faces: número de caras de la triangulación. En este caso el valor es nulo, porque todavía no se ha aplicado ningún algoritmo de triangulación.

Una vez cargado, simplificaremos el modelo 3D eliminando los puntos considerados erróneos o que no pertenecen al modelo. A continuación importaremos la malla (File > Import Mesh) seleccionando el archivo "nombre.0.ply" . Al igual que en el paso anterior, seleccionaremos los vértices que no se corresponden con nuestro modelo para eliminarlos.

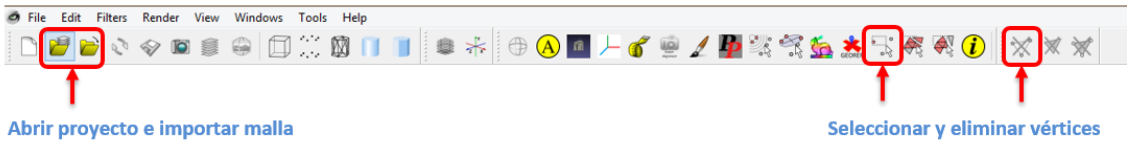

Figura 47. Importación de la nube de puntos y malla en MeshLab.

2) **Creación de la malla**. Para la reconstrucción superficial de la malla poligonal formada por triángulos, utilizaremos la triangulación por Reconstrucción de Poisson. Para ello aplicaremos el filtro "Screened Poisson Surface Reconstruction" (desde la opción del menú superior Filters > Remeshing, Simplification and Reconstruction). Según su descripción, este algoritmo reconstruye superficies herméticas a partir de conjuntos de puntos orientados. Para este tipo de filtro debe seleccionarse un valor para la profundidad (*Reconstruction Depth*), que suele estar comprendido entre 5 y 10. Para valores altos, la aproximación a la superficie será mayor, pero a costa de un mayor coste computacional. En las reconstrucciones presentadas en los casos prácticos se seleccionó un valor de 8, excepto para el caso práctico 1 en el que se seleccionó el valor máximo, por tratarse de una nube de puntos muy poco densa.

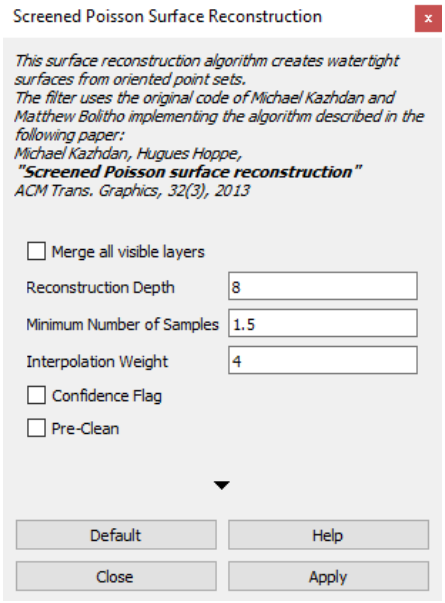

Figura 48. Valores de la reconstrucción de Poisson

Al igual que en el primer paso del proceso, se seleccionarán las caras erróneas y que no pertenecen al objeto, para ser eliminadas.

Figura 49. Eliminación de caras erróneas en MeshLab

En el caso de obtenerse un número elevado de caras (> 700.000) simplificaremos la geometría aplicando el algoritmo de optimización incremental "Quadric Edge Collapse Decimation" (desde la opción del menú superior Filters > Remeshing, Simplification and Reconstruction) para reducir el número de triángulos y de vértices.

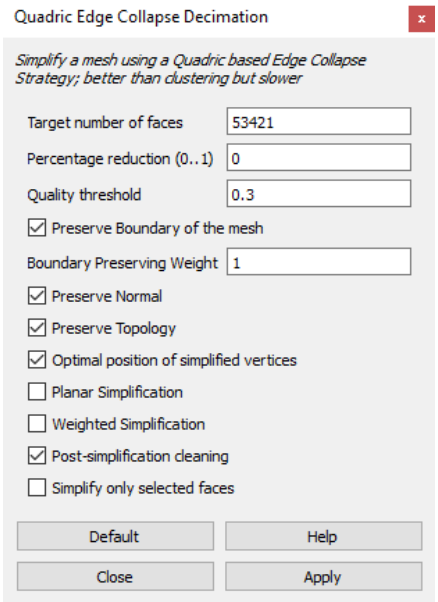

Figura 50. Ejemplo de simplificación de la geometría en MeshLab

Para conservar la topología del modelo y la orientación de los triángulos deben marcarse las opciones "Preserve Normal" y "Preserve Topology". Este proceso puede aplicarse tantas veces como sea necesario.

3) **Aplicación de la textura a la malla**. Una vez realizada la reconstrucción, se continúa con la texturización del objeto. Para ello seleccionaremos la opción "Parametrization Fron Registered Rasters + Texturing" (desde la opción del menú superior Filters > Texture).

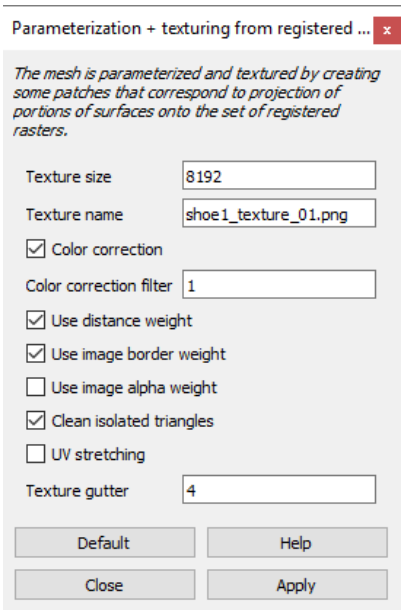

Figura 51. Aplicación de textura en MeshLab

En este punto debemos asignar un tamaño al mapa de textura y un nombre para guardarlo con formato de imagen. *MeshLab* propone una resolución de 8192*8192 píxeles para el mapa de textura, pero en nuestro caso comprobamos que un mapa de textura de 1024*1024 píxeles se aplica correctamente al modelo 3D. Para obtener el modelo texturizado y poder exportarlo a otro tipo de herramientas de visualización de modelos 3D, deberemos guardarlo previamente (File > Export Mesh As), en un fichero con formato “nombre.obj”.

Bibliografía

[adidasgroup, 2017] Adidas oficial site AG. https://www.adidas-group.com/en

[Autodesk, 2018] Autodesk® portfolio solutions. Available at: https://www.autodesk.com/solutions/123d-apps

[Blender, 2018] Open Source 3D Creation Suite. Available at: https://www.blender.org/

[C. Wu y otros, 2010] "VisualSfM: A Visual Structure from Motion System". Available at: http://ccwu.me/vsfm/doc.html.

[D.G. Lowe y otros, 1999] Object recognition from local scale-invariant features. International Conference on Computer Vision, Corfu, Greece, pp. 1150-1157, 1999.

[D.G. Lowe y otros, 2004] Distinctive Image Features from Scale-Invariant Key-points. Available at: http://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf

[E. Alonso y otros, 2014] E. Alonso. "Desarrollo y validación de un sistema de modelado 3D de software abierto". 2014. Available at: https://uvadoc.uva.es/bitstream/10324/13086/1/TFG-I-155.pdf

[E. Vals y otros, 2015] E. Vals, "Seguimiento de patrones faciales por descriptores de forma". Available at: http://www.maia.ub.es/~sergio/linked/quique08.pdf

[F. Bernardini y otros, 1999] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, and G. Taubin, "The ball-pivoting algorithm for surface reconstruction," IEEE Trans. on Visualization and Computer Graphics, vol. 5, pp. 349-359, 1999.

[F. Dellaert y otros, 2000] F. Dellaert, S.M. Seitz, C.E. Thorpe, & S. Thrun, "Structure from Motion without Correspondence". CVPR, 2000.

[H. C. Longuet-Higgins y otros, 1981] H. C. Longuet-Higgins, "A computer algorithm for reconstructing a scene from two projections". Nature, vol. 293, pp. 133-135, 1981.

[I. Navarro y otros, 2016] I. Navarro, O. de Reina, A. Rodiera and D. Fonseca, "Indoor positioning systems: 3D virtual model visualization and design process of their assessment using mixed methods: Case study: World heritatge buildings and spatial skills for architecture students," 2016 11th Iberian Conference on Information Systems and Technologies (CISTI), pp. 1-6, 2016.

[I. S. Alameda y otros, 2013] I. S. Alameda, "Métodología para la Gestión y Explotación de Datos Espaciales Obtenidos con Sistemas de Captura Masiva de Puntos," Universitat Politécnica de Catalunya, Barcelona, 2013.

[J. D. Camba y otros, 2016] J. D. Camba, A. B. De Leon, J. de la Torre, J. L. Saorín and M. Contero, "Application of low-cost 3D scanning technologies to the development of educational augmented reality content," 2016 IEEE Frontiers in Education Conference (FIE), pp. 1-6, 2016.

[J. Sánchez y otros, 2007] J. Sánchez, "Vision tridimensional. Geometría epipolar y matriz fundamental", pp 16-18, 2007.

[J. Weng y otros, 1989] J. Weng, T. Huang, "Motion and structure from two perspective views: algorithms, error analysis, and error estimation". IEEE Transactions on Patterns Analysis and Machine Intelligence. Volume 11 Nº 5, pp. 451 – 476, 1989.

[L. Ling y otros, 2017] L. Ling, G. Juntao and D. Xi, "The design and implementation of the 3D virtual campus models," 2017 4th International Conference on Systems and Informatics (ICSAI), pp. 1747-1751, 2017.

[MeshLab, 2016] Página oficial de MeshLab. http://www.meshlab.net/

[M. Hervás y otros, 2015] M. Hervás, "Técnicas de vision por computador y realidad aumentada aplicadas a la gestión documental", TFG, pp. 17-24, 2015

[M. Kazhdan y otros, 2006] M. Kazhdan, M. Bolitho and H. Hoppe, "Poisson surface reconstruction," in In Eurographics Symposium on Geometry Processing, 2006.

[M. Kazhdan y otros, 2013] M. Kazhdan and H. Hoppe, "Screened Poisson surface reconstruction," ACM Trans. Graph 32, 3, no. 29, pp. 1-13, 2013.

[GNU Octave, 2018] Scientific Programming Language. Available at: https://www.gnu.org/software/octave/

[O.D. Faugeras y otros, 1992] O.D. Faugeras, "What Can Be Seen In Three Dimensions With an Uncalibrated Stereo Rig". Available at: http://image.diku.dk/imagecanon/material/Faugeras92.pdf

[S. D. Orcero y otros, 2012] S. D. Orcero "La matriz fundamental y la matriz esencial. Conceptos y aplicaciones". Available at: http://www.orcero.org/irbis/fundamental/slides.pdf

[P. Cignoni y otros, 2008] P. Cignoni, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, G. Ranzuglia "MeshLab: an Open-Source Mesh Processing Tool". Sixth Eurographics Italian Chapter Conference, pp. 129-136, 2008.

[Pinholecameramodel, 2018] Imágenes del módelo y la geometría de la cámara pinhole. Available at: http://en.wikipedia.org/wiki/Pinhole_camera_model

R.I. Hartley y otros, 1992] R.I. Hartley, "Estimation of relative camera positions for uncalibrated cameras". Available at: http://users.cecs.anu.edu.au/~hartley/Papers/eccv92/Higgins/higgins.pdf

[R.J. Carrasco y otros, 2003] R.J. Carrasco. "Modelo de cámara pinhole". Disponible en: http://catarina.udlap.mx/u_dl_a/tales/documentos/lis/carrasco_r_j/capitulo2

[R. Szeliski y otros, 2010] R. Szeliski, "Computer Vision: Algorithms and Applications", draft, pp. 1-28, 2010.

[R. Tsai y otros, 1982] R. Tsai, T. Huang, “Estimating three-dimensional motion parameters of a rigid planar match”. IEEE Transactions on Acoustics, Speech, and Signal Processing, ASSP30, pp. 525-534. 1982.

[S. Agarwal y otros, 2009] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz and R. Szeliski, "Building Rome in a day," 2009 IEEE 12th International Conference on Computer Vision, Kyoto, 2009, pp. 72-79, 2009.

[S.A.J. Winder y otros, 2007] S.A.J. Winder, M. Brown, “Learning Local Image Descriptors”. 2007 IEEE Conference on Computer Vision and Pattern Recognition, Minneapolis, MN, pp. 1-8, 2007.

[StereoCamera, 2004] Stereo Camera. Available at: http://www.cse.unr.edu/~bebis/CS791E/Notes/StereoCamera.pdf

[W. E. Lorensen y otros, 1987] W. E. Lorensen and H. E. Cline, “Marching cubes: A high resolution 3d surface construction algorithm,” In ACM SIGGRAPH Computer Graphics, vol. 21, pp. 163-169, 1987.

[Y. Furukawa y otros, 2010] Y. Furukawa. and J. Ponce. “Clustering View for MultiStereo (CMVS)”. Available at: http://www.di.ens.fr/cmvs/

[Y. Furukawa y otros, 2010] Y. Furukawa. and J. Ponce. “Patch-Based Multi-View Stereo Software (PMVS – Version 2)”. Available at: http://www.di.ens.fr/pmvs/

[Z. Xie y otros, 2010] Z. Xie and W. Jiang, "Research for 3D Digital Analysis Model Applied in Anatomic Medicine," 2010 International Conference on Biomedical Engineering and Computer Science, pp. 1-4, 2010.

Pliego de Condiciones

PL.1 Introducción

En los siguientes apartados se especifican las versiones y características del conjunto de herramientas software y equipos hardware necesarios para la realización y desarrollo del presente Proyecto Fin de Carrera.

PL.2 Equipos hardware

- PC de sobremesa con procesador Acer Intel(R) Core(TM) 2 Quad CPU Q9650 @ a 3.00 GHz y memoria RAM de 4GB, propiedad de la Escuela de Ingeniería de Telecomunicación de la ULPGC
- Portátil personal Aspire E5-573 Intel(R) Core(TM) i5-5200U CPU @ a 2.20 GHz y memoria RAM de 8GB
- Cámara de fotos modelo Canon EOS 550D, propiedad de la empresa
- Cámara de fotos personal modelo Nikon COOLPIX S2500
- Impresora HP LaserJet P1505n
- Memoria USB Pen-drive de 16GB

PL.2 Herramientas software

- **Autodesk® 123D™ Catch y ReCap™ de Autodesk Labs**: herramienta de modelado 3D utilizados para realizar pruebas de obtención de modelos 3D en la empresa
- **Linux Ubuntu 16.04.4 LTS**: sistema operativo bajo el que se realizaron las simulaciones finales
- **Windows 8.1**: sistema operativo empleado en la etapa de investigación y redacción del presente proyecto
- **VisualSfM y MeshLab**: programas utilizados en la realización de la implementación de la solución propuesta
- **Pack Microsoft Office 365 ProPlus**: Word (empleado para la redacción del proyecto), Excel (utilizado para la creación de las tablas de resultados), PowerPoint (usado para la

presentación e ilustración de diagramas e imágenes), Proyect Manager (empleado para la construcción de la planificación temporal).

- **TeamViewer**: aplicación gratuita usada para acceder remotamente al ordenador principal en el que fueron realizadas las simulaciones (www.teamviewer.com).

Presupuesto

Doña Cristina Girón Domínguez, autora del presente Proyecto Fin de Carrera, declara que:

El Proyecto Fin de Carrera con título "Escáner 3D de bajo coste y alta precisión aplicado a la adaptación y diseño de prendas de vestir", desarrollado para la compañía adidas International Trading B.V. en colaboración con la División de Diseño de Sistemas Integrados del Instituto Universitario de Microelectrónica Aplicada, de la Universidad de Las Palmas de Gran Canaria, en el período de un año, tiene un coste de desarrollo total de **65.420,04€** correspondiente a la suma de las cantidades consignadas a los apartados considerados a continuación.

La autora del proyecto

Cristina Girón Domínguez

Julio de 2018

PR.1 Desglose del presupuesto

Para la realización del presupuesto se han seguido las recomendaciones del Colegio Oficial de Ingenieros de Telecomunicación (COIT) sobre los baremos orientativos mínimos para trabajos profesionales en 2009. Actualmente, los colegios profesionales ya no pueden establecer baremos de honorarios orientativos ni de ningún otro tipo, según se estableció en el artículo 5 de la Ley 25/2009, de 22 de diciembre, de modificación de diversas leyes para su adaptación a la ley sobre el libre acceso a las actividades deservicio y su ejercicio.

El presupuesto se ha desglosado en distintos conceptos, donde se ha tenido en cuenta la duración total del PFC, la cual se estima en 12 meses:

- Recursos materiales.
- Trabajo tarifado por tiempo empleado.

- Costes de redacción del proyecto.
- Material fungible.
- Derechos de visado del COIT.
- Gastos de tramitación y envío.
- Aplicación de impuestos.

PR.2 Costes de los recursos materiales

Entre los recursos materiales utilizados para la realización de este proyecto, se incluyen las herramientas software de desarrollo de los algoritmos del sistema, los paquetes software usados para la redacción de la memoria, y el sistema operativo bajo el que se ejecutó el trabajo. Asimismo, se incluyen los equipos hardware usados para dar soporte a estas herramientas.

Se estipula el coste de amortización para un período de 3 años. Para ello, se utilizará un sistema de amortización lineal o constante, en el que se supone que el inmovilizado material se deprecia de forma constante a lo largo de su vida útil. La cuota de amortización anual se calcula usando la siguiente fórmula:

$$Cuota\ Anual = \frac{Valor\ de\ adquisición - Valor\ residual}{Número\ de\ años\ de\ vida\ útil} \qquad (PR.\ 1)$$

donde el valor residual es el valor teórico que se supone que tendrá el elemento después de su vida útil.

Los costes de amortización se calcularán para el primer año, teniendo en cuenta la duración del proyecto. De una forma similar se calcula la amortización, que se rige por la siguiente expresión:

$$Amortización = \frac{Cuota\ anual \times Tiempo\ de\ uso}{12\ meses}(€) \qquad (PR.\ 2)$$

PR.2.1 Recursos hardware

Teniendo en cuenta que la duración del PFC es de 12 meses y el cálculo del coste de amortización se establece en un período de 3 años, los costes de amortización se calcularán para el primer año. En la siguiente tabla se presentan los costes de amortización del material hardware empleado:

Equipo	Valor de adquisición	Valor residual (3 años)	Coste de amortización (1 año)
PC de sobremesa, procesador Acer Intel(R) Core (2 Quad CPU Q9650 @ 3.00 GHz, 4GB RAM)	1.750 €	0 €	583,33 €
Portátil personal Aspire E5-573 Intel(R) Core(TM) (i5-5200U CPU @ a 2.20 GHz, 8GB RAM)	465 €	0 €	155 €
Cámara de fotos Canon EOS 550D	324 €	0 €	108 €
Cámara de fotos personal Nikon COOLPIX S2500	79 €	0 €	26,33 €
Impresora HP LaserJet P1505n	710 €	0 €	236,66 €
Memoria USB Pen-drive de 16GB	12 €	0 €	4 €
		Total	1.113,32 €

Tabla 3. Costes de amortización de los recursos hardware

El coste total de los recursos hardware libre de impuestos asciende a *mil ciento trece* euros y *treinta y dos* céntimos (**1.113,32 €**).

PR.2.2 Recursos software

Para el cálculo de los costes de amortización de los recursos software se manejan las mismas premisas que en el apartado anterior. La siguiente muestra los elementos software empleados en la realización del proyecto, así como su valor de adquisición, valor residual y coste de amortización.

Concepto	Valor de adquisición	Valor residual (3 años)	Coste de amortización (1 año)
Autodesk® 123D™ Catch, ReCapTM de Autodesk Labs	0 €	0 €	0 €
Linux Ubuntu 16.04.4 LTS	0 €	0 €	0 €
Windows 8.1	102 €	0 €	34 €
VisualSfM	0 €	0 €	0 €
MeshLab	0 €	0 €	0 €
Pack Microsoft Office 365 ProPlus	155 €	0 €	51,66 €
TeamViewer	0 €	0 €	0 €
		Total	85,66 €

Tabla 4. Costes de amortización de los recursos software

Finalmente, el coste total del material software libre de impuestos asciende a *ochenta y cinco* euros y *sesenta y seis* céntimos (**85,66 €**).

PR.3 Trabajo tarifado por tiempo empleado

Siguiendo las recomendaciones del COIT, se obtiene una aproximación del importe de las horas empleadas en la realización del proyecto. Los honorarios totales se calculan en base a la siguiente expresión:

$$H = (C_t \times 74{,}88 \times H_n) + (C_t \times 96{,}72 \times H_e)\ (€) \qquad \textit{(PR. 3)}$$

donde:

- H son los honorarios totales por el tiempo dedicado.
- H_n son las horas normales trabajadas (dentro de la jornada laboral).
- H_e son las horas especiales.
- C_t es un factor de corrección función del número de horas trabajadas.

Se estima que para la realización del presente PFC se ha trabajado durante 12 meses, lo que corresponde a 1920 horas (considerando 20 días laborables al mes, con una dedicación de 8 horas por jornada laboral), todas en horario normal. Según el COIT el coeficiente corrector C_t tiene un valor variable en función del número de horas empleadas de acuerdo con la siguiente tabla:

Horas empleadas	Factor de corrección C_t
Hasta 36 horas	**1,00**
36 a 72 horas	**0,90**
72 a 108 horas	**0,80**
108 a 144 horas	**0,70**
144 a 180 horas	**0,65**
180 a 360 horas	**0,60**
360 a 540 horas	**0,55**
540 a 720 horas	**0,50**
720 a 1080 horas	**0,45**
Más de 1080 horas	**0,40**

Tabla 5. Coeficiente de corrección Ct en función de las horas trabajadas

Para el número de horas indicado el COIT especifica un coeficiente corrector de 0,4, por lo que el coste total de los honorarios asciende a:

$$H = (0{,}4 \times 74{,}88 \times 1920) + (0{,}4 \times 96{,}72 \times 0) = 57.600\ (€) \qquad \textit{(PR. 4)}$$

El trabajo tarifado por tiempo empleado asciende a la cantidad de *cincuenta y siete mil seiscientos* euros (**57.600 €**).

PR.4 Material fungible

En esta sección se consignan los costes asociados a los materiales utilizados en la realización del proyecto, como son: el papel de impresión, tóner de la impresora, CD-Rom, y otros, que suponen un valor de *ciento cincuenta* euros (**150 €**).

PR.5 Costes de redacción del proyecto

El COIT recomienda utilizar la siguiente expresión para determinar el coste asociado a la redacción de la memoria del proyecto:

$$R = 0{,}07 \times P \times C_h \qquad \textit{(PR. 5)}$$

donde:

- R son los honorarios por la redacción del proyecto
- P es el presupuesto del proyecto.
- C_h es el coeficiente de ponderación en función del presupuesto.

El valor del presupuesto P se calcula sumando los costes de las secciones anteriores correspondientes al trabajo tarifado por tiempo empleado y a los costes de los recursos materiales (tanto hardware como software). El coeficiente de ponderación a aplicar

definido por el COIT para el tramo comprendido entre 42.070,70 € y 63.106,05 €, es de 0,45. Por tanto, se obtiene:

$$R = 0{,}07 \times (1.113{,}32 + 85{,}66 + 57.600 + 150) \times 0{,}45 = 1.856{,}89\ € \qquad \textit{(PR. 6)}$$

El coste de la redacción del proyecto asciende a *mil ochocientos cincuenta y seis* euros y *ochenta y nueve* céntimos (**1.856,89 €**).

PR.6 Derechos de visado del COIT

El COIT establece que para la redacción de proyectos y trabajos en general, los derechos de visado se calculan en base a la siguiente expresión.

$$V = 0{,}006 \times P \times C_v \qquad \textit{(PR. 7)}$$

donde:

- V es el coste de visado del proyecto.
- P es el presupuesto.
- C_v es el coeficiente reductor en función del presupuesto.

El valor del presupuesto P se halla sumando los costes de las secciones anteriores correspondientes al trabajo tarifado por tiempo empleado, a los costes de los recursos materiales (tanto hardware como software) y a la redacción del proyecto. En este caso, el coeficiente reductor C es el asignado para el tramo entre 30.050 € y 60.101 €, cuyo valor es de 0,9. Por lo que aplicando la ecuación se obtiene:

$$V = 0{,}006 \times (1.113{,}32 + 85{,}66 + 57.600 + 150 + 1.856{,}89) \times 0{,}9 = 328{,}35\ € \qquad \textit{(PR. 8)}$$

Los costes por derechos de visado del presupuesto ascienden a *trescientos veintiocho* euros y *treinta y cinco* céntimos (**328,35 €**).

PR.7 Gastos de tramitación y envío

Los gastos de tramitación y envío son fijos y se estipulan en 6,01 € (*seis* euros con *un* céntimo).

PR.8 Aplicación de impuestos

El importe del presupuesto lleva implícito el importe del IGIC (impuesto general indirecto canario). Para esta actividad económica el IGIC graba con un 7 %. El presupuesto calculado hasta el momento asciende a 61.140,23 € por lo que el impuesto asciende a 4.279,81 € (*cuatro mil doscientos setenta y nueve* euros y *ochenta y un* céntimos).

En la siguiente tabla se muestra el presupuesto final de este PFC con los impuestos aplicados.

Concepto	Coste
Trabajo tarifado por tiempo empleado	57.600 €
Recursos hardware	1.113,32 €
Recursos software	85,66 €
Material fungible	150 €
Redacción del proyecto	1.856,89 €
Derechos de visado del COIT	328,35 €
Gastos de tramitación y envío	6,01 €
TOTAL (sin IGIC)	61.140,23 €
IGIC (7%)	4.279,81 €
TOTAL (con IGIC)	65.420,04

Tabla 6. Presupuesto total del proyecto

El importe total al que asciende el presupuesto es de **65.420,04 €** (*sesenta y cinco mil cuatrocientos veinte* euros y *cuatro* céntimos).

La autora del Proyecto

Cristina Girón Domínguez

Julio de 2018