Universidad de Valladolid

ESCUELA DE INGENIERÍAS INDUSTRIALES

UNIVERSIDAD DE VALLADOLID

ESCUELA DE INGENIERIAS INDUSTRIALES

Grado en Ingeniería Mecánica

Desarrollo y validación de un sistema de modelado 3D de software abierto

Autor:

Alonso López, Eric

Tutor:

Mahíllo Isla, Raúl

Ingeniería de los Procesos de Fabricación

Valladolid, Septiembre 2014.

RESUMEN

Desde el punto de vista ingenieril el proceso que se explica en el siguiente proyecto trata de crear un modelo 3D a partir de fotos del elemento real. Para ello utilizaremos dos programas de software abierto: Visual SfM y Meshlab.

Las posibilidades de las nuevas herramientas basadas en técnicas de fotogrametría, nos aportan un excelente medio para la documentación gráfica 3D. Por su accesibilidad desde el punto de vista económico, así como facilidad de implementación o manejo, constituyen una alternativa a métodos de modelado 3D basados en tecnología láser o luz estructurada.

Y gracias a las nuevas impresoras 3D de software libre podemos imprimir los modelos obtenidos, construyendo un proceso de prototipado 3D. Teniendo, en definitiva, un desarrollo CAD/CAM sencillo y económico.

Palabras clave: software abierto o libre, fotogrametría, nube de puntos, reconstrucción superficial, impresión 3D.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

1. INTRODUCCIÓN

1.1. Introducción

Este documento expone las pautas necesarias para el correcto desarrollo de un modelo 3D partiendo de imágenes del modelo real. Desde la iluminación y captura de fotos, hasta la impresión final.

En las últimas décadas, los avances en visión artificial han lanzado al mercado nuevas propuestas de modelado 3D como el *structure from motion* (SfM) [1]. Este sistema se basa en el fenómeno por el cual la visión del ser humano puede reconstruir estructuras tridimensionales a partir de imágenes 2D.

Aunque ya existían trabajos en torno al SfM en los años 80, será en esta última década, gracias en parte al aumento de potencia de cálculo de los procesadores, cuando comience a surgir herramientas consistentes y capaces de resolver modelos 3D complejos con una cierta eficacia y agilidad. Una de estas herramientas es Visual SfM y será la que utilicemos en la mayor parte del trabajo. Este software ha sido desarrollado principalmente por Changchang Wu y Yasutaka Furukawa –ingeniero de Google Maps- [2].

Visual SfM contiene complejas herramientas de fotogrametría, con el fin de automatizar ciertas rutinas y liberar al usuario de localizar los puntos comunes entre imágenes, pudiendo utilizar grandes cantidades de imágenes para describir con el mayor detalle posible la geometría de objetos o superficies.

A partir de los resultados que nos proporciona Visual SfM, utilizando Meshlab, crearemos el modelo digital que finalmente imprimiremos en la impresora 3D analizando los factores fundamentales de todo el proceso de prototipado.

1.2. Objetivos

A continuación se enumeran los objetivos más importantes:

- Manipular herramientas de software libre.
- Explicar cómo capturar las fotos.
- Crear una nube densa a partir de fotografías.
- Reconstruir la superficie.
- Desarrollar un flujo de trabajo para el prototipado 3D.
- Conocer los factores más influyentes en el proceso y sus limitaciones.
- Imprimir el modelo 3D obtenido.
- Verificar el desarrollo del modelo y el resultado final.

2. ESPECIFICACIONES DEL EQUIPO

En este apartado se comentará el equipo utilizado durante la realización del proyecto:

2.1. Cámaras

Nexus

Figura 2.1: Cámara Nexus

Especificaciones técnicas:

- 8MP
- 3264x2448 pixels
- Autofocus
- Flash LED
- Foco táctil
- Estabilización óptica de imagen
- HDR (o HDRI, *High Dynamic Range Imaging*)

Destacando su facilidad de manejo, rapidez para tomar fotos y su autofocus, importante para las fotos más cercanas a los modelos.

Canon EOS 550D

Figura 2.2: Cámara EOS 550D

Especificaciones técnicas:

- Sensor CMOS APS-C de 18MP
- DIGIC 4
- ISO 100-6400, H:12800
- Sistema de medición iFCL
- Disparo en ráfaga a 3.7 fps

A pesar de su gran calidad fotográfica, no ofrecía un resultado muy diferente a la cámara *Nexus*, aumentando el tiempo de la toma de fotos y el procesado posterior debido a un mayor tamaño en las fotos. Este tipo de cámara es más útil si

queremos realizar un modelo muy grande, como por ejemplo una estatua o una fachada de un edificio.

2.2. CPU

En lo que respecta a la CPU, sus especificaciones técnicas básicas son:

- Sistema Operativo: Windows 7 Home Premium 64-bit
- Procesador: Intel(R) Core(TM) i7-2600K CPU @ 3.40GHz (8 CPUs), ~3.4GH
- Memoria RAM: 8192MB
- DirectX Version: DirectX 11
- Tarjeta gráfica: AMD Radeon HD 7900 Series

Comentar que los procesos calculados en la CPU nombrada, también han sido testeados en un portátil TOSHIBA SATELLITE A660-1EQ. Aunque los tiempos de procesado aumentaban significativamente los resultados eran idénticos.

Los programas utilizados para la nube de puntos y el tratamiento superficial son válidos para cualquier ordenador que cumpla las especificaciones mínimas de estos.

2.3. Impresora 3D

La impresora 3D utilizada es una Prusa i3, con un volumen de impresión de 20x20x20cm, se encuentra dentro de las máquinas auto-replicantes llamadas RepRap.

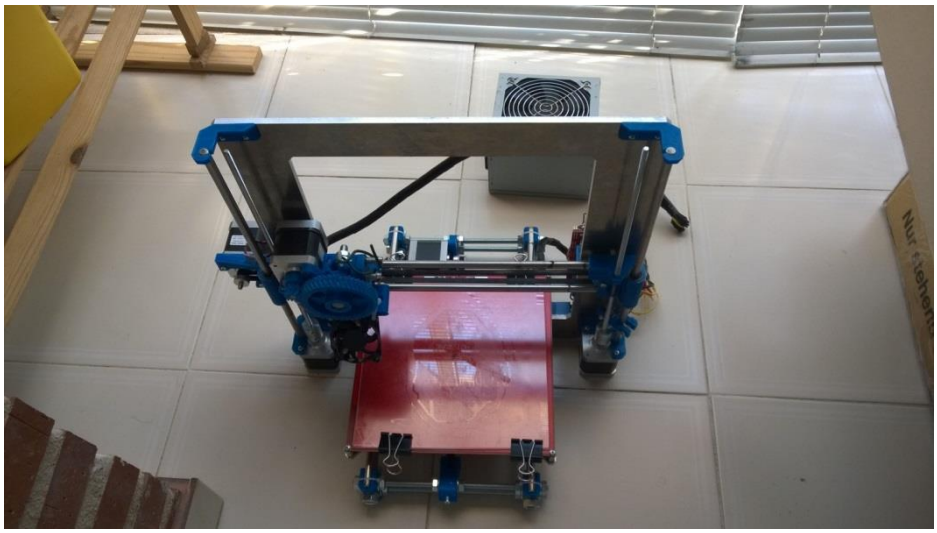

Figura 2.3: Impresora 3D Prusa i3

Se denominan RepRap a las impresoras 3D libres capaz de imprimir objetos normalmente de plástico. Donde muchas partes de la propia impresora están hechas de ese mismo plástico, como vemos en la Figura 2.3 los elementos de color azul. Por lo tanto pueden auto-replicarse haciendo un kit de sí mismas.

El plástico utilizado fue PLA y los programas para la preparación del proceso de impresión fueron *Repetier host* y *Marlin,* ambos de software abierto.

3. FOTOGRAFÍA

La recreación 3D del modelo real para su posterior impresión parte de las fotografías tomadas a éste. Como es lógico es necesario tener un conocimiento básico de los factores fundamentales y el proceso fotográfico utilizado para obtener un modelo aceptable.

3.1. Factores fundamentales

Los factores fundamentales de una buena fotografía son la sensibilidad ISO, la apertura del diafragma y la velocidad de obturación. Estos tres factores se concentran en la denominada exposición.

La exposición es la acción de someter un elemento fotosensible a la acción de la luz. Por ello una buena exposición será lo que busquemos en todas nuestras fotos. En función del grado de exposición de una foto podremos hablar de tres situaciones: subexposición, exposición y sobreexposición.

Figura 3.1: Diferentes grados de exposición

Observando la Figura 3.1 se identifica los resultados de diferentes exposiciones para la misma foto.

Buscamos una fotografía de exposición correcta que recoja la cantidad de luz apropiada para representar fielmente la escena fotografiada.

Como hemos nombrado con anterioridad, los factores que determinan la exposición son la sensibilidad ISO, la apertura del diafragma y la velocidad de obturación (Figura 3.2).

- Sensibilidad ISO: Refleja la sensibilidad del sensor de nuestra cámara ante la luz que actúa sobre él. Una mayor sensibilidad hará que, a igual cantidad de luz y tiempo de incidencia, el sensor se haya excitado más y, por tanto, la fotografía tenga una mayor exposición
- Apertura del diafragma: Determina la cantidad de luz que se deja incidir sobre el sensor de nuestra cámara. Una mayor apertura supondrá una mayor cantidad de luz actuando sobre el sensor.
- Velocidad de obturación: También llamado tiempo de exposición. Marca el tiempo durante el cual la luz incide sobre el sensor. Un mayor tiempo y, por tanto, una menor velocidad darán lugar a que la luz incida durante un periodo más prolongado sobre el sensor.

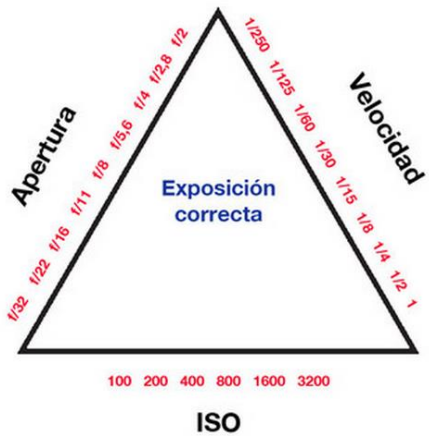

Figura 3.2: Factores para una correcta exposición

3.2. Proceso de captura

El factor más decisivo en este proceso es la luz. Es necesaria una buena iluminación de las superficies a fotografiar para que el programa Visual SfM capte el mayor número de puntos posibles, que explicaremos en el apartado 4.2.

La situación óptima para las condiciones lumínicas de nuestro modelo sería la exposición de luz solar en un día nublado, el cual no produce cambios de luz excesivos ni sombras. Pero como este proyecto quiere definir un sistema de prototipado 3D, no es aceptable depender de las condiciones meteorológicas.

Por todo ello, los modelos creados en este trabajo fueron expuestos a **luz blanca** por medio de dos flexos, permitiendo la eliminación parcial de sombras en las zonas a fotografiar. También es importante no sobreexponer al modelo, porque los reflejos blancos que crea la luz no podrán ser captados en Visual SfM. La utilización de flash no es posible, ya que los diferentes cambios de luz en cada fotografía no permiten un posicionamiento correcto de los puntos.

Una vez que tenemos la figura bien iluminada debemos configurar nuestra cámara a este tipo de iluminación:

- Para la sensibilidad escogeremos el **menor valor de ISO posible**, así reduciremos el ruido que pueda tener nuestro sensor para valores de ISO altos.
- Al reducir la sensibilidad, necesitaremos una **apertura de diafragma alta** para captar la mayor cantidad de luz. Aumentando éste valor reduciremos nuestra profundidad de campo.
- Y escogeremos una **velocidad de obturación baja**, dependiendo si hacemos las fotos manualmente o con trípode, en el segundo caso podremos reducirlo al mínimo valor de nuestra cámara.

La profundidad de campo es un término utilizado en fotografía para expresar el rango de distancias reproducidas con una nitidez aceptable en una foto. Por ello

dependerá en mayor medido de la distancia de enfoque que necesitemos. Reduciendo el ruido que obtendremos en la reconstrucción de la malla densa en Visual SfM (Apartado 4.4, d)).

También se puede reducir el ruido de fondo utilizando fondos unicolores. Afectando a la captación de puntos clave. En el trabajo se utilizaron cartulinas blancas, fáciles de posicionar y mover, para hacer las capturas de todo el modelo.

Finalmente, ya podemos capturar todas las imágenes de nuestra pieza que sean necesarias. El recorrido que tomemos dependerá en mayor medida de la forma del objeto a fotografiar y el resultado final que queramos obtener. Es importante planificar la sucesión de fotos porque una baja coincidencia entre estas resultará en una reconstrucción errónea, formando varios modelos diferentes.

Por ello, a continuación explicaremos unas pautas para el proceso de captura de fotos, diferenciando entre la reproducción de un modelo 3D completo o de una superficie.

- Para un **modelo 3D completo**, la **forma** más adecuada de nuestro recorrido alrededor de la pieza será en **espiral**. Variando entre 5º y 15º el ángulo entre fotos consecutivas. A menor grado mayor densidad de puntos, pero mayor número de fotos. Un alto número de fotos solo se traduce en un alto coste computacional.
- Para la **reconstrucción de una superficie**, la **forma** más adecuada de nuestro recorrido será en ***ZIG-ZAG***, ya sea horizontal o vertical. Pero siempre manteniendo una sucesión uniforme, sin grandes saltos entre fotos.

Una de las ventajas de trabajar con Visual SfM es que el patrón que hagamos a la hora de hacer el recorrido no tiene por qué ser muy regular, así nos podremos centrar en realizar una mayor densidad de fotos en las zonas más críticas de nuestro modelo.

Los recorridos utilizados para los modelos estudiados en el proyecto se pueden ver en las siguientes figuras.

Figura 3.3: Distribución de fotos en los modelos 3D. Tiburón (Izquierda) y Búho (Derecha)

Figura 3.4: Distribución de fotos en el modelo superficial

En los inicios del proyecto quisimos automatizar éste proceso, pero es inviable crear un sistema que realice fotografías de manera adecuada sin crear sombras y, en definitiva, afectar negativamente a la reconstrucción 3D. Para ello, existen procesos de luz estructurada o técnicas laser mucho más costosos.

4. PROGRAMA Visual SfM

La fotogrametría es la disciplina que se encarga del estudio de las propiedades geométricas de objetos o escenas así como sus características espaciales a partir de fotografías. El principal atributo de estas técnicas, es que mientras la fotografía sólo representa atributos bidimensionales, la fotogrametría trabaja con información tridimensional, obtenida a partir de diversas imágenes bidimensionales solapadas. A través de estas zonas de solape se pueden determinar puntos comunes con los que se recrean las vistas 3D.

Para realizar los modelos 3D Visual SfM, como su nombre indica, utiliza la tecnología SfM que explicaremos en el apartado 4.2. En resumen, el objetivo de Visual SfM es crear una nube de puntos 3D lo suficientemente densa para poder formar, a partir de estos puntos, la superficie de nuestro modelo.

4.1. Instalación

A la hora de instalar Visual SfM deberemos ir a su página web [2] y seguir las instrucciones que nos indican. Nos centraremos en la instalación para Windows aunque también es posible instalarlo en Linux o Mac OSX.

Tendremos dos opciones: descargar el programa para Windows, ya sea de 32 o 64 bits, o también la versión que se aprovecha de la tecnología CUDA, la cual mejora los tiempos de procesado. Este tipo de tecnología solo viene implementado en las gráficas nVidia, si quisiéramos utilizarlo sin tener una tarjeta nVidia deberíamos descargarnos el emulador de CUDA y seguir las instrucciones de instalación y ejecutado descritas en [3].

Con este programa queremos conseguir una malla de puntos densa y para ello además de instalar el ejecutable VisualSFM.exe es necesario incorporar las

librerías CMVS/PMVS que se explican en el apartado 4.3. Estas librerías se deben colocar en la misma carpeta donde hayamos instalado el programa. La propia página [2] nos facilita estos archivos.

4.2. Tecnología SfM

El resultado de un proceso de fotogrametría o SfM es, en primera instancia, una nube de puntos discreta, la cual puede variar en densidad en función de las correspondencias detectadas entre imágenes. En los procesos tradicionales de ajuste manual de puntos comunes entre imágenes, estas nubes no estaban especialmente pobladas; sin embargo con SfM dichas nubes pueden llegar a tener miles de puntos. Estas nubes discretas, o poco densas, son el resultado de lo que se conoce como ***bundle adjustment***. Este concepto está íntimamente ligado a la fotogrametría y permite realizar la recolocación espacial tanto de una serie de puntos coincidentes entre imágenes como de las posiciones de las cámaras que tomaron dichas imágenes respecto a la escena.

Más formalmente, supongamos que n puntos 3D son vistos en m imágenes y $\mathrm{x_{ij}}$ es la proyección del i-ésimo punto sobre la imagen j. Denotamos como v_{ij} a la variable binaria cuyo valor será 1 cuando i es visible en la imagen j, y 0 en caso contrario. Cada imagen será parametrizada por un vector a_j y cada punto 3D por un vector b_i. *Bundle adjustment* minimiza el error de reproyección total con respecto a todos los puntos proyectados y los parámetros de las imágenes, ecuación (4.1). Donde $Q(a_j, b_i)$ es la proyección estimada del punto i en la imagen j y $d(Q(a_j, b_i), x_{ij})$ expresa la distancia Euclidea entre los puntos representados por los vectores $Q(a_j, b_i)$ y x_{ij}.

$$\min_{\boldsymbol{a_j}, \boldsymbol{b_i}} \sum_{i=1}^{n} \sum_{j=1}^{m} \boldsymbol{v_{ij}} \boldsymbol{d}(\boldsymbol{Q}(\boldsymbol{a_j}, \boldsymbol{b_i}), \boldsymbol{x_{ij}})^2 \qquad \textbf{(4.1)}$$

Dichos puntos pueden ser establecidos manualmente, o en el caso del SfM son detectados automáticamente mediante lo que se conoce como el proceso **SIFT o *scale-invariant feature transform***, publicado en 1999 por David Lowe, mediante el cual se detectan puntos o características comunes entre pares de imágenes, lo que nos permite cotejar un gran número de imágenes con el fin de extraer grandes cantidades de puntos comunes o puntos clave.

El algoritmo de extracción de características SIFT se basa en la idea de "espacio de escala". El **espacio de escala** de una imagen es una pirámide formada por todos los posibles reescalados de la imagen; y supone que, dada una característica, la respuesta del detector será máxima en un reescalado de entre todos los posibles, mientras que en los adyacentes en la pirámide se tendrá una menor respuesta. Opera realizando cuatro etapas fundamentales para detectar y describir características locales o puntos clave, en una imagen:

1. **Detección del espacio de escala.** En primer lugar, se construye el espacio de escala aplicando una serie de suavizados Gaussianos sobre la imagen original y se agrupan por "octavas". Cada octava corresponde a reducir el tamaño de la imagen a la mitad o, en este caso, a doblar el radio de suavizado. Tras esto se restan las imágenes adyacentes de cada escala obteniendo una diferencia de Gaussianas como muestra la Figura 4.1

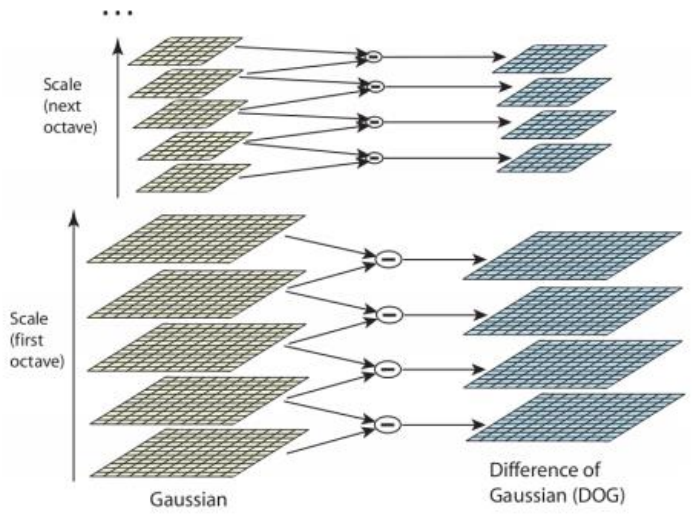

Figura 4.1: Combinación de imágenes suavizadas con suavizado Gaussiano para obtener la diferencia de Gaussianas en el algoritmo SIFT. En la escala superior, se produce un mayor suavizado

2. **Localización y filtrado de puntos clave.** Cada posible punto clave es encontrado como el punto de máxima respuesta (máximo o mínimo) de la diferencia de Gaussianas, comparándolo con sus vecinos en la imagen y con los puntos vecinos de las imágenes adyacentes del espacio de escala, como se muestra en la Figura 4.2.

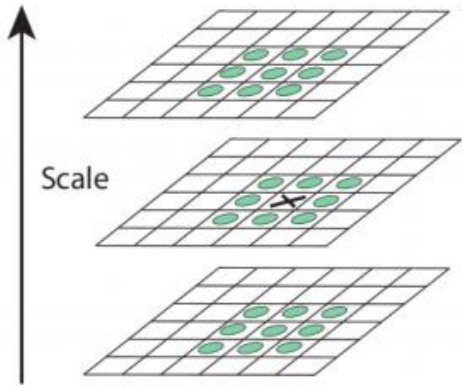

Figura 4.2: El punto marcado con la X se compara con cada uno de sus adyacentes, marcados con un círculo, buscando el punto en el que la diferencia de Gaussianas es máxima.

A continuación, es necesario filtrar para eliminar los puntos de bajo contraste y los que se encuentran en bordes, ya que la diferencia de Gaussianas produce una fuerte respuesta en torno a los bordes de la imagen. El resultado de este proceso puede verse en la Figura 4.3.

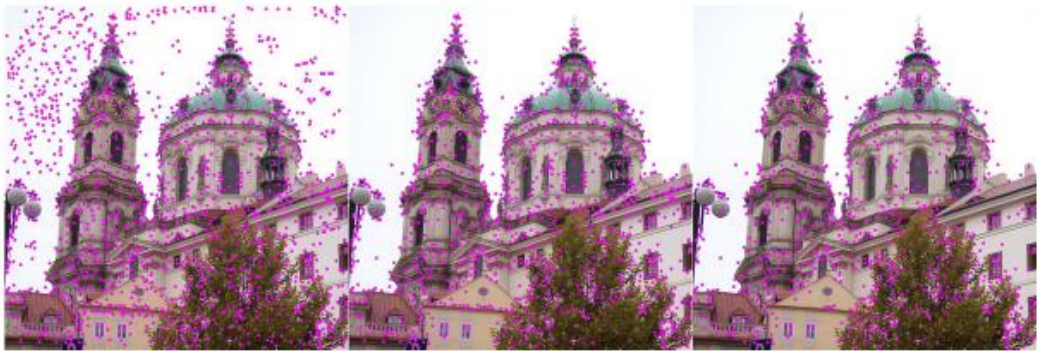

Figura 4.3: Puntos clave detectados en varias etapas por el algoritmo SIFT. De izquierda a derecha: todos los puntos detectados, tras eliminar los de bajo contraste y eliminando bordes.

3. **Asignación de orientaciones.** Para dotar a los *keypoints* de independencia a la rotación, se aplica un post-proceso, consistente en calcular la mejor aproximación del vector gradiente a una de las direcciones discretizadas. Así, se compara la magnitud y dirección de dicho vector con varias direcciones discretizadas, habitualmente 36 direcciones con 10 grados de amplitud cada una, y se asigna la orientación (u orientaciones) que tienen una respuesta máxima.

4. **Cálculo del descriptor.** Ahora, para cada *keypoint* localizado, se calcula un descriptor que lo identifique unívocamente y que sea invariante a la posición, escala y rotación y a otras características indeseables, como la iluminación. Para ello, se crean histogramas de la orientación de los vectores gradiente cerca del punto, en regiones de $4x4$ píxeles, con 8 cubetas cada histograma. Esto se hace en 16 histogramas alrededor del *keypoint*, teniendo en total 128 datos que son los que formarán el vector descriptor. Para conseguir invarianza a la iluminación, cada histograma es normalizado a la unidad, se le aplica un

umbral de 0,2 y se vuelve a normalizar. En la Figura 4.4 puede verse un ejemplo de este proceso, con 4 histogramas alrededor del *keypoint.*

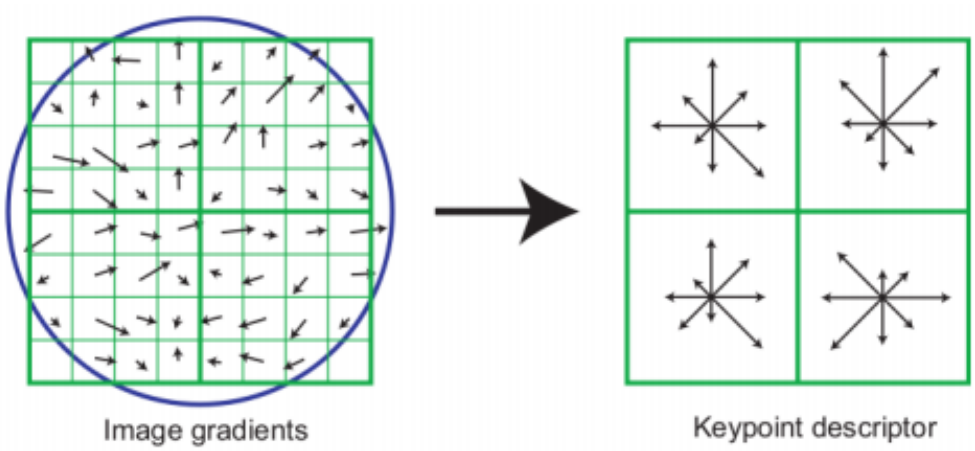

Figura 4.4: Los gradientes en torno a un punto son usados por SIFT para calcular el descriptor como un histograma normalizado de gradientes.

Para mayor información el artículo [4] escrito por el propio autor del algoritmo profundiza más en su funcionamiento.

Aunque estas nubes discretas puedan arrojar información sobre una geometría son insuficientes para evaluar en detalle un objeto o escena, por eso el paso final de estos procesos de SfM suele ser una nube de puntos densa con millones de puntos que describen con más detalle las superficies y geometría de objetos.

4.3. Densificado

Multi-view stereo (MVS) [5] es el proceso clave para el modelado 3D por fotografías. Su campo de aplicaciones va desde la reconstrucción de modelos realistas de objetos para la industria del cine, televisión y videojuegos hasta la metrología científica e ingenieril. De acuerdo con un estudio [6] los algoritmos MVS permiten una precisión superior de 1/200 (1mm de 20cm del objeto) a partir de un

conjunto de 640×480 imágenes (baja resolución). Pueden ser clasificados en cuatro clases:

I. Aproximaciones basadas en voxel, requiere conocer un cuadro delimitador que contenga la imagen, y su precisión es limitada por la resolución de la malla voxel [7].
II. Algoritmos basados en mallas poligonales deformables, requieren una malla inicial para aplicar el proceso de optimización, esto delimita su aplicación [8].
III. Aproximaciones basadas en múltiples planos de profundidad, son más flexibles pero requieren juntar las diferentes profundidades en un único modelo 3D [9].
IV. Método por parches (PMVS) que representa superficies de la imagen por medio de pequeños planos [10]

Las librerías nombradas en el apartado 4.1 contienen los algoritmos CMVS/PMVS (*Clustering Views for Multi-view Stereo/Patch-based Multi-view Stereo Software*) que permitirán densificar la malla de puntos obtenida a través de las técnicas descritas en el apartado 4.2. Por lo tanto, Visual SfM utiliza un densificado por parches mediante el algoritmo realizado por Yasutaka Furukawa, en el artículo [5] viene descrito en profundidad, pero para saber cómo funciona de una manera general a continuación se explicará brevemente.

El método PMVS es un sistema basado en parches, los cuales son unos rectángulos centrados en cada punto de la reconstrucción, y con vector normal orientado hacia la cámara Figura 4.5 a). Cada parche p, se asocia a la imagen $R(p)$. También se añade a las imágenes desde las que debería ser visibles, pero puede que no sean reconocibles $S(p)$, y se detecta en las imágenes desde las que realmente se ve $T(p)$ Figura 4.5 b).

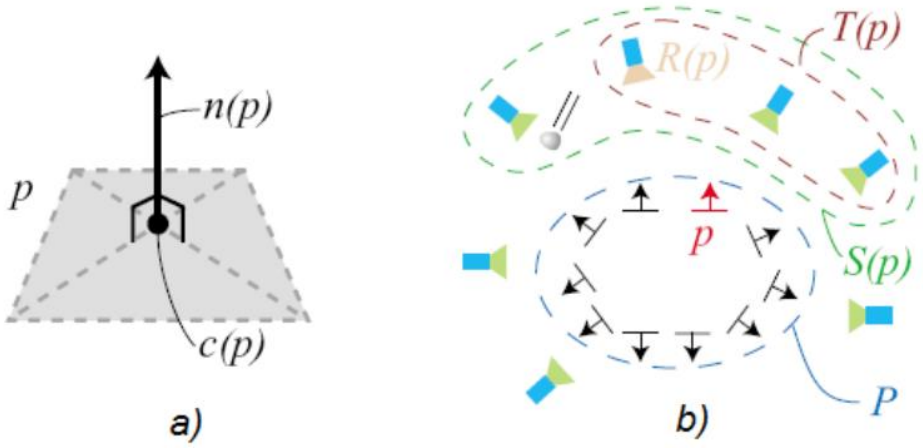

Figura 4.5: a) Representación gráfica de un parche $\boldsymbol{p}$, vector normal $\boldsymbol{n}(\boldsymbol{p})$, centro del parche $\boldsymbol{c}(\boldsymbol{p})$
b) Esquema gráfico del modelo por parches .

Al crearse el primer modelo de parches, posteriormente permite la iteración del algoritmo produciendo la expansión de la superficie representada. Se añaden vecinos a los parches que se han creado, hasta que se cubren las superficies visibles en la escena.

Finalmente se hace un filtrado en el que según las superficies estimadas, se detecta si los parches añadidos se encuentran dentro de ella o no, aceptando todo aquel que se encuentre dentro de la superficie estimada $U(p)$, y eliminando los que caigan fuera de ella Figura 4.6.

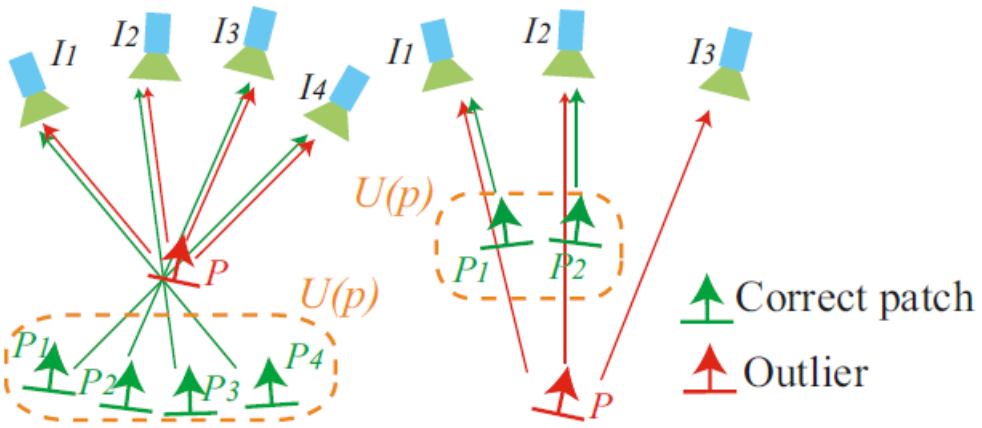

Figura 4.6: Filtrado de parches

Cuando hablamos de nubes de puntos densas nos referimos a un conjunto de vértices descritos en un sistema de coordenadas tridimensionales tipo XYZ. Donde además de la información espacial, se acompaña cada vértice o punto de una descripción colorimétrica en el modelo RGB.

Esta combinación de información geométrica o espacial con datos colorimétricos resulta especialmente interesante a la hora de recopilar información descriptiva de una obra o escena, pero conservar el color de nuestro modelo no es el objetivo de este trabajo.

4.4. Walkthrough

Al ejecutar VisualSfM veremos una interfaz gráfica típica de la mayoría de programas Figura 4.7, y a su vez, tendremos una ventana complementaria para visualizar las tareas que vayamos realizando. La ventana principal se divide en tres zonas: barra de menús, barra de herramientas y espacio de trabajo.

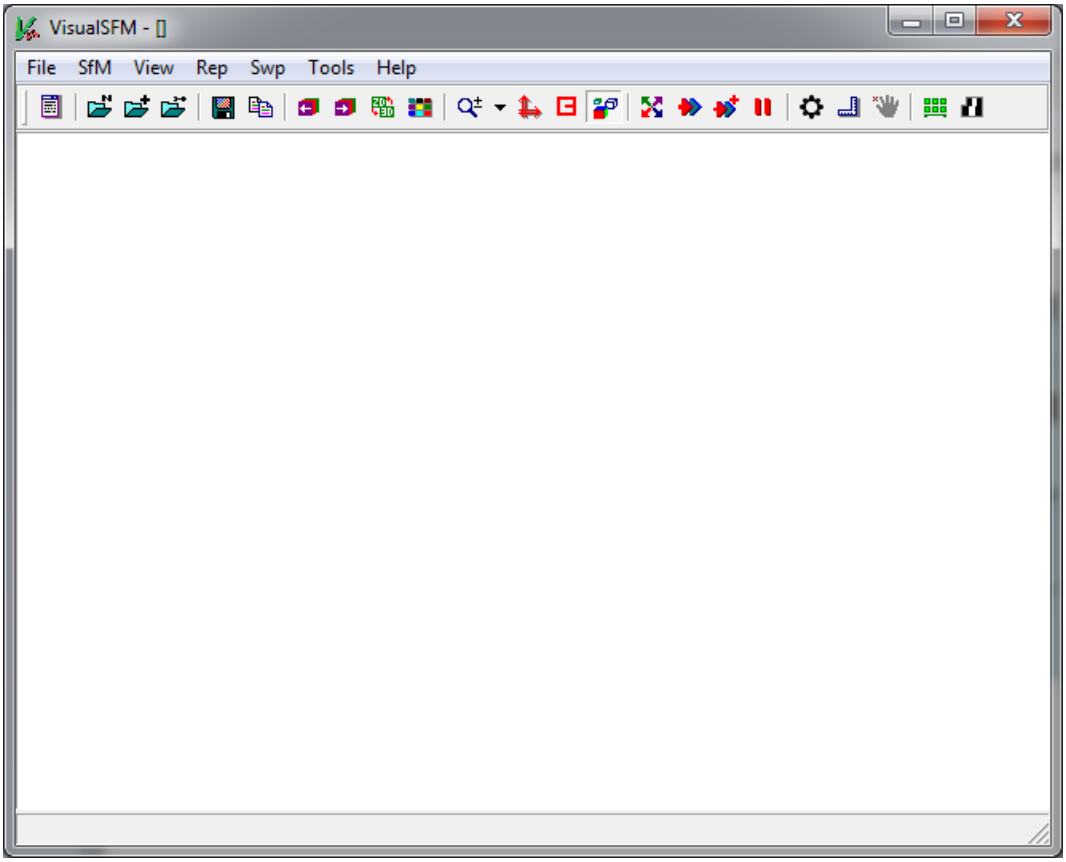

Figura 4.7: Ventana principal

Los pasos que debemos seguir para llegar a una nube de puntos densa son los siguientes:

a) Abrir fotos: Esta tarea se puede realizar de dos formas, seleccionando los iconos siguientes:

Es necesario un archivo .txt que contenga todos los nombres de las fotos que queremos abrir en la carpeta donde las hayamos guardado.
Barra de herramientas > SfM > Load NView Match

Simplemente seleccionamos las fotos desde la carpeta que las contenga.
Barra de herramientas > File > Open+ Multi Images

Al abrir las fotos no se mostrarán en la pantalla de trabajo, para ello es necesario pulsar el icono  , y aparecerán todas las imágenes cargadas Figura 4.8. Antes de seguir con el procedimiento es muy importante comprobar todas las fotos. Es necesario que estén correctamente expuestas para obtener el mejor resultado, como ya se ha explicado en el apartado 3.1, dando doble click en la foto que queremos inspeccionar y ampliándola con la ruleta del ratón.

Para eliminar las fotos indeseadas se seleccionaran en la lista de fotos (apareciendo un recuadro rojo alrededor de las seleccionadas) y pulsaremos la tecla suprimir.

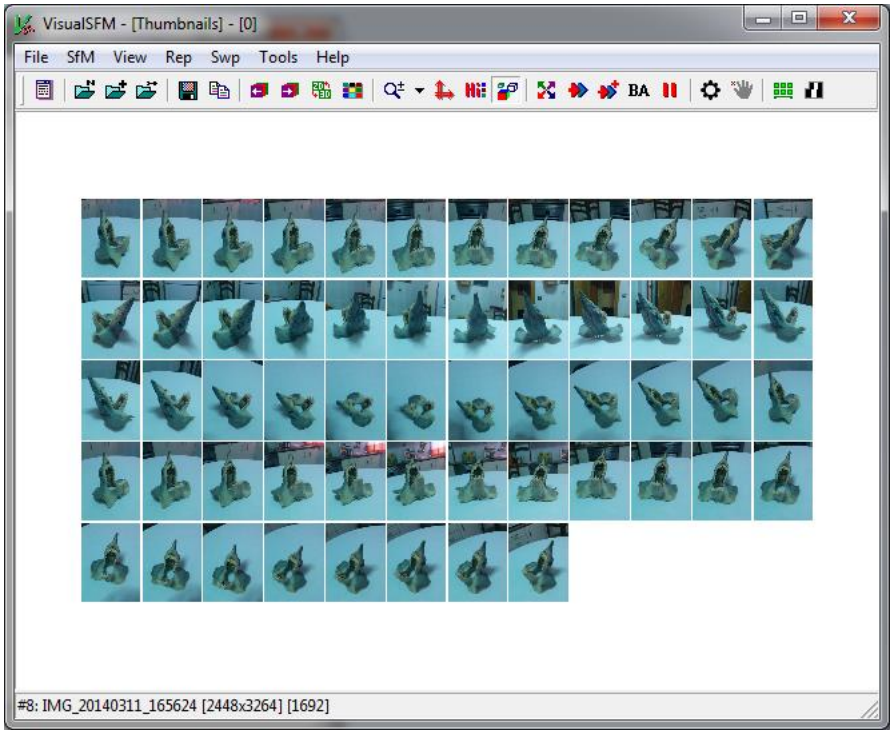

Figura 4.8: Visualización de imágenes

Como se aprecia en la Figura 4.8 la serie de fotografías se ha ordenado para disminuir el tiempo de emparejamiento, que será el siguiente paso, y de esta forma mejorar las coincidencias que toma el programa para obtener la nube de puntos discreta.

b) Emparejamiento de imágenes: En este paso se detectan las características comunes y las coincidencias entre pares de fotos. Para ello seleccionaremos el siguiente icono:

Barra de herramientas > SfM > Pairwise Matching > Compute Missing Match

En la ventana de visualización de tareas se mostrará algo parecido a la Figura 4.9 donde, al iniciar el cálculo, nos indica el número de pares identificados, en este caso 171, y al finalizar el cálculo el número de pares calculado y el tiempo transcurrido en la operación.

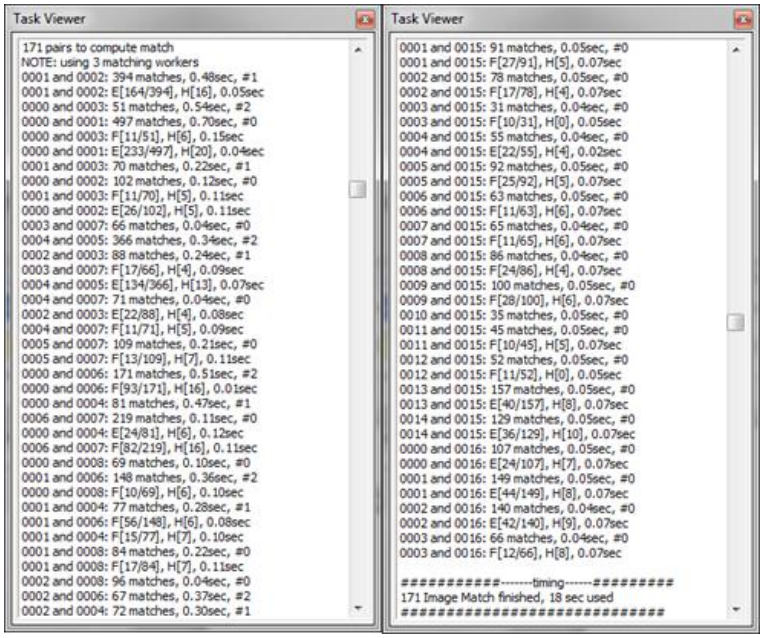

Figura 4.9: Visualización del cálculo en el emparejamiento imágenes inicio (Izquierda), final (Derecha)

c) <u>Reconstrucción 3D poco densa</u>: En este paso entra en juego los procedimientos descritos en el apartado 4.2. Para iniciar la reconstrucción a partir de las imágenes seleccionamos el icono:

 Barra de herramientas > SfM > Reconstruct Sparse

Este proceso necesitará un mayor tiempo de cálculo dependiendo del número de fotos y su calidad. En la Figura 4.10 se muestra el proceso de reconstrucción. Para minimizar o maximizar las imágenes que muestran las fotos y su orientación apretamos la tecla Ctrl y a su vez movemos la ruleta

del ratón. Si queremos visualizar el tiempo transcurrido pulsaremos la tecla 'X' como nos indica en la barra inferior de la Figura 4.10.

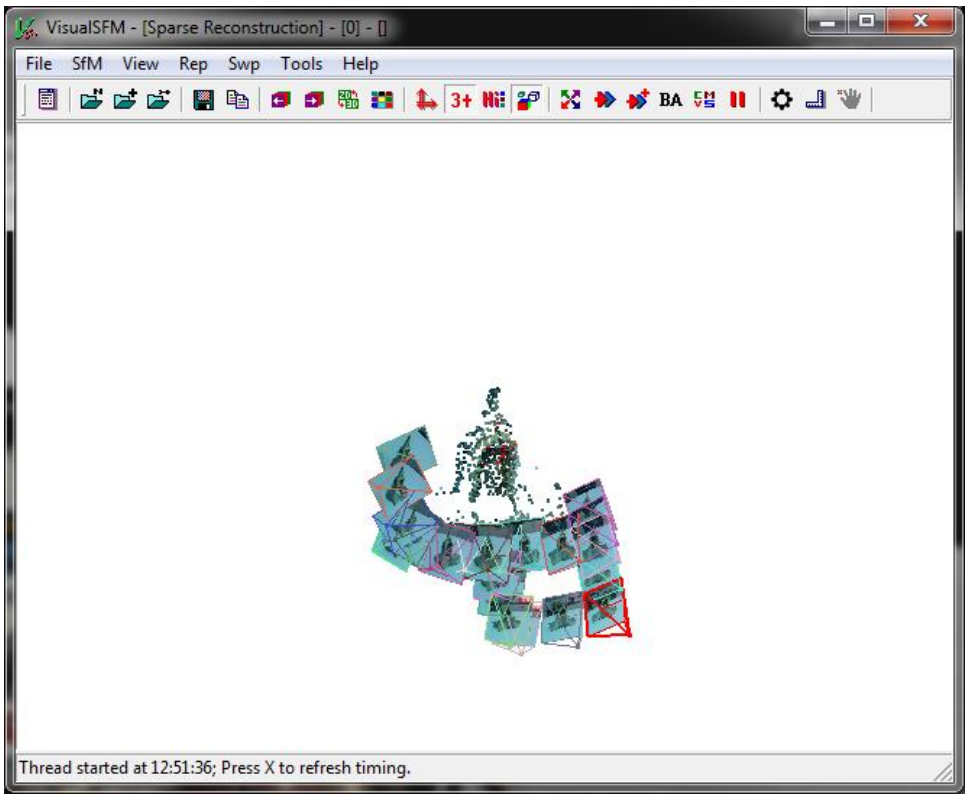

Figura 4.10: Proceso de reconstrucción 3D poco densa

d) <u>Reconstrucción 3D densa</u>**:** Como ya explicamos en el apartado 4.3 para poder llegar a un modelo 3D es necesario una nube de puntos mucho más densa que la que nos proporciona la reconstrucción SfM. Para ello el programa utiliza el algoritmo creado por Yasutaka Furukawa's CMVS/PMVS [5]. En este paso se creará una carpeta para guardar los parámetros del modelo denso como [nombre].nvm.cmvs y dos archivos fuera de esta carpeta, uno con la extensión [nombre].nvm y otro como **[nombre].0.ply.** Éste último será el que utilizaremos para la reconstrucción superficial en MeshLab.

Para mostrar los resultados obtenidos deberemos ir a *View>Dense 3D points*. Normalmente en la mayoría de modelos tendremos puntos no deseados que podremos eliminar con MeshLab, pero como se ha descrito en el apartado 3.2 para minimizarlos es necesario tener fondos unicolor o desenfocados, y si no es posible, se podría trabajar las fotos con algún programa de edición fotográfico, como por ejemplo GIMP. En el ejemplo que estamos tratando se obtuvo la nube densa que se ve en la Figura 4.11.

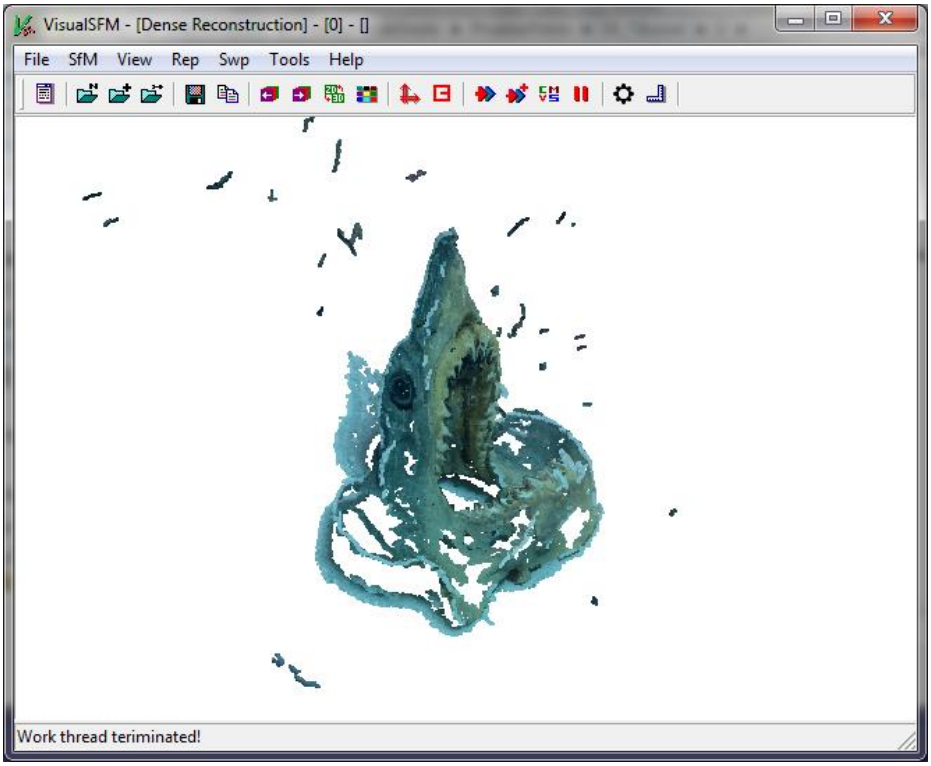

Figura 4.11: Primeros resultados obtenidos con VisualSfM

5. PROGRAMA MeshLab

MeshLab es un programa portable y de código abierto para la edición y procesado de modelos de superficies en forma de mallas triangulares 3D.

Se ha diseñado para el procesado de nubes de puntos, proporcionando herramientas para la edición, limpieza, relleno, búsqueda, renderizado y conversión de este tipo de mallas.

Y se basa, prácticamente en su totalidad, por la *VCG library*, desarrollada por el *Visual Computing Lab* del *ISTI-CNR*, y está disponible para Windows, MacOSX y Linux.

MeshLab se creó en 2005 como parte del curso *FGT* del Departamento de Informática de la Universidad de Pisa, y está apoyado por el proyecto *3D-CoForm,* se puede descargar gratuitamente de la propia página [11].

En este apartado mostraremos como a partir de la exportación de una nube de puntos en VisualSfM creamos una superficie en MeshLab. Para ello debemos realizar los siguientes pasos.

5.1. Importación en Meshlab

Una vez abierto el programa, podemos seleccionar desde la barra de menús File>Import Mesh… , también en la barra de herramientas aparece el icono para importar la malla o directamente utilizando el comando Ctrl+I.

5.2. Reducción de ruido

En términos de tratamiento de nubes de puntos y modelado tridimensional, se considera ruido a aquellos puntos que se han registrado y que no pertenecen a la superficie que nos interesa. Normalmente son zonas cercanas al objeto que han sido detectadas por el programa y no son deseadas.

Existen algoritmos para la reducción del ruido que se basan, por ejemplo, en el hecho de que los puntos que tienen pocos o ningún punto a su alrededor se consideran ruido. Sin embargo, con Meshlab lo más sencillo es eliminar los puntos que el usuario crea convenientes.

El objetivo es crear un flujo de trabajo que pueda aplicarse a cualquier objeto o superficie a modelar, y obviamente, el usuario será el encargado de eliminar los puntos que el crea indeseables para ese modelo.

En el programa, utilizaremos las siguientes herramientas para la creación de la superficie final.

5.3. Planos tangentes y orientación de la superficie

A partir de una nube de puntos de distribución uniforme y sin ruido es posible determinar un conjunto de planos tangentes, que nos dan una primera aproximación lineal y local de la superficie, así como los vectores normales asociados a éstos, que proporcionan información sobre la orientación de dicha superficie.

Un plano tangente T_p asociado a un punto x_i de la nube X está representando por un punto o_i llamado centroide, y está asociado a su vez con un vector normal o perpendicular $\vec{n}_i$. Tanto el centroide como la normal al plano están determinados por los puntos vecinos de x_i.

Cada plano tangente se calcula a partir de los vecinos más adecuados de x_i determinados por mínimos cuadrados, y cada normal se calcula a partir de las direcciones principales del plano. Al determinar la normal se obtiene la orientación del plano.

Si suponemos dos puntos $x_i, x_j \in X$, y la nube tiene una distribución uniforme y suave (no hay variaciones excesivas en altura), entonces los planos tangentes $T_p(x_i) = (o_i, \vec{n}_i)$ y $T_p(x_j) = (o_j, \vec{n}_j)$ son prácticamente paralelos y

están orientados. Por lo tanto, el producto escalar $\vec{n}_i \times \vec{n}_j \approx \pm 1$. Ahora bien, todos los pares de planos tangentes deben cumplir esta propiedad para poder conseguir la orientación global de la superficie.

El problema de la orientación global se puede modelar en forma de grafo $G(V, E)$ contiene un vértice $i \in V$ de cada plano tangente $T_p(x_i)$ y las aristas $i, j \in E$ que conectan dos planos tangentes con centros o_i y o_j. Este grafo, llamado Grafo de Riemannian [12], se construye así para observar la proximidad geométrica de los centroides de los planos tangentes.

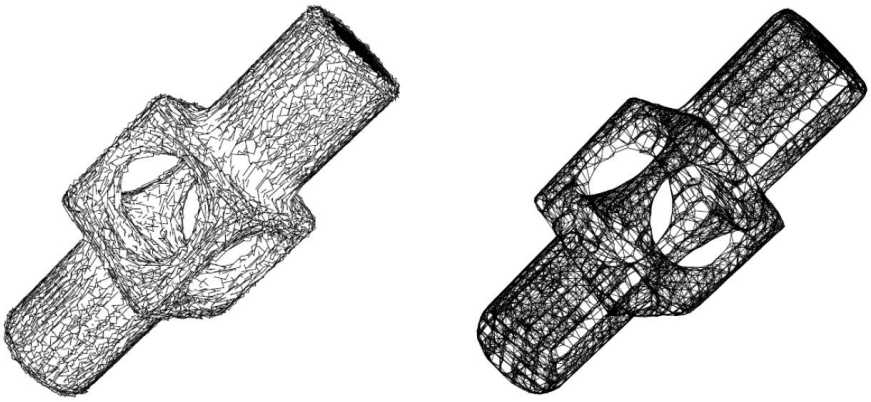

Figura 5.1: Planos tangentes (izquierda) y Grafo de Riemannian (derecha).

Para conseguir dicha orientación global se empieza por uno de los planos y se "propaga" la orientación a los planos adyacentes del grafo. Un orden de propagación favorable es aquel que favorece la propagación de $T_p(x_i)$ a $T_p(x_j)$ siempre y cuando los planos no orientados sean prácticamente paralelos. El método más común consiste en asignar a cada arista (i, j) el valor $1 - |\vec{n}_i \cdot \vec{n}_j|$ en el Grafo de Riemannian. Este valor, además de no ser negativo, tiene la propiedad de que es menor si los planos tangentes no orientados son prácticamente paralelos.

Una vez conseguida la orientación global es posible encontrar la distancia $d_u(p)$ desde un punto arbitrario $p \in R^3$ a un punto de la nube X: primero se encuentra el plano tangente $T_p(x_i)$ cuyo centroide es o_i, y luego se calcula la distancia entre el p y el punto $z \in X$, que en el caso de la Figura 5.2 es la proyección de p en el plano. Matemáticamente la distancia $d_u(p)$ se expresa en la ecuación (5.1).

$$d_u(p) = (p - o_i) \cdot \vec{n}_i \quad \textbf{(5.1)}$$

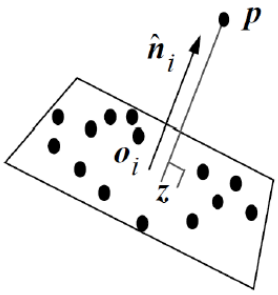

Figura 5.2: Centroide y normal asociados a un plano tangente y distancia de un punto exterior a la proyección de éste sobre el plano

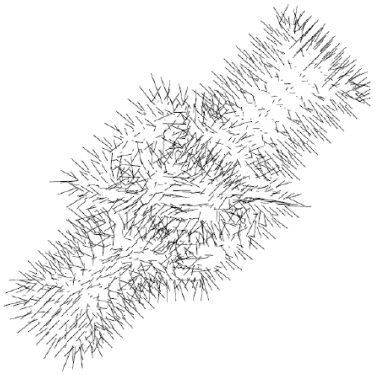

Figura 5.3: Vectores normales asociados a los vértices. La orientación de cada vector es la misma que la del plano tangente al que pertenece el vértice

Para calcular las normales en Meshlab deberemos ejecutar el menú **Filters > Point Set > Smooths normals on a point sets** entonces aparece la siguiente ventana Figura 5.4, en la cual se recomienda aceptable el valor por defecto [13]. Cuanto mayor sea este valor mayor es la aproximación a la superficie, pero aumenta el coste computacional.

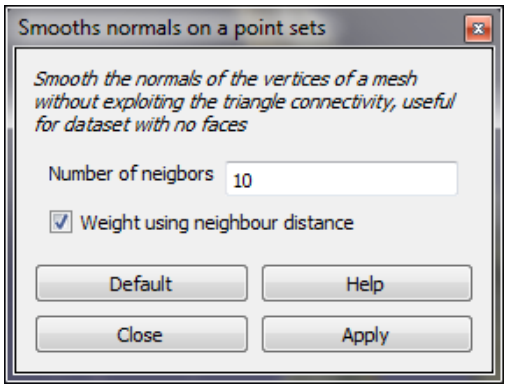

Figura 5.4: Ventana para calcular las normales de los puntos de nuestra malla

Como describe el mismo programa lo que el comando realiza es: "Calcular las normales de los vértices de una malla sin explotar la conectividad de triángulos, útil para un conjunto de datos sin superficies". Además tenemos la opción de utilizar el tamaño de las distancias entre puntos vecinos.

Deberemos pulsar el botón "Apply" para ejecutar el comando. Este no tiene variación gráfica en la nube de puntos y así podremos seguir con el siguiente paso, la creación de la superficie. Para comprobar que el resultado ha sido satisfactorio podemos visualizar todos los vectores que han sido calculados para cada punto en **Render > Show Vertex Normals** Figura 5.5.

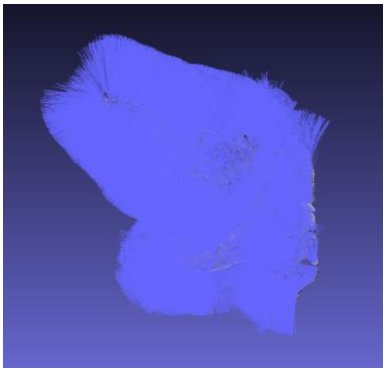

Figura 5.5: Vectores normales calculados en la cabeza de tiburón

5.4. Reconstrucción superficial

La reconstrucción superficial es el proceso de transformación de la nube de puntos en un modelo tridimensional. Al modelo creado inicialmente se le deberán eliminar aquellos elementos erróneos y redundantes, y optimizarlo si es necesario a las características del hardware como se comentó en el apartado 5.2.

MeshLab ofrece la posibilidad de una reconstrucción superficial en mallas poligonales, formadas por triángulos. A este tipo de reconstrucción se le llama triangulación. Destacan cuatro métodos de triangulación: Triangulación de Delaunay, Algoritmo de la bola pivotante, Algoritmo de los cubos móviles y Reconstrucción por *Poisson* [12].

En este proceso utilizaremos la triangulación por Reconstrucción de *Poisson.* A continuación se explicará brevemente en que consiste:

- Reconstrucción de *Poisson*

La reconstrucción de la superficie por *Poisson* se basa en determinar [14] una función implícita χ que asigne como 1 lo puntos que se encuentren dentro de la superficie, como 0 los de fuera y desechar estos últimos, esquema mostrado en la Figura 5.6.

Los datos de entrada son cada uno de los puntos x_i de la nube χ y los vectores normales $\vec{n}_i$ asociados, explicados en el apartado 5.3, los cuales forman el campo vectorial $\vec{N}$.

La idea principal es que existe una relación entre los puntos de la nube orientados y la función implícita del modelo. Concretamente, el gradiente de la función es un campo vectorial $\vec{V}$, que es 0 en casi todas partes excepto en los puntos cercanos a la superficie. De esta manera los puntos orientados pueden ser tratados como puntos del gradiente de la función implícita del modelo.

El problema de calcular la función implícita se reduce a invertir el operador del gradiente, o lo que es lo mismo, a encontrar la función escalar χ cuyo gradiente mejor se aproxime al campo vectorial $\vec{V}$ definido por los puntos. Esto quiere decir que $min_\chi = \left|\nabla - \vec{V}\right|$. Y al aplicar el operador de divergencia (operador laplaciano) el problema se convierte en una ecuación de *Poisson:*

$$\Delta\chi = \nabla \cdot \nabla_\chi = \nabla \cdot \vec{V} \qquad \textbf{(5.2)}$$

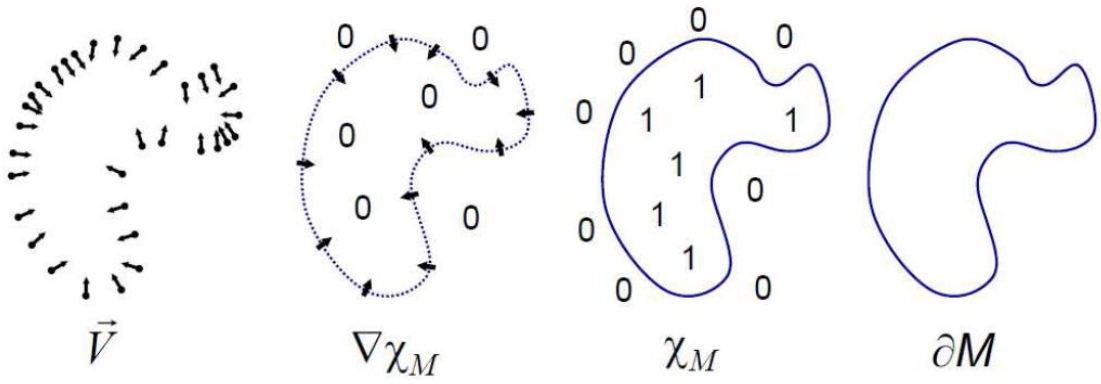

Figura 5.6: Esquema construcción de Poisson 2D

Para definir el campo vectorial es necesario primero dividir la superficie en celdas tridimensionales basadas en *octrees.* Un *octree* es una estructura de datos en árbol en la que cada nodo tiene 8 octantes o, también, denominados “hijos”, y

se representa en forma de celdas 3D. Cada nodo es una celda que se subdivide en 8 nuevas celdas. El nivel de profundidad D del *octree* hace referencia al número máximo de veces en que se divide una celda. Ejemplo Figura 5.7.

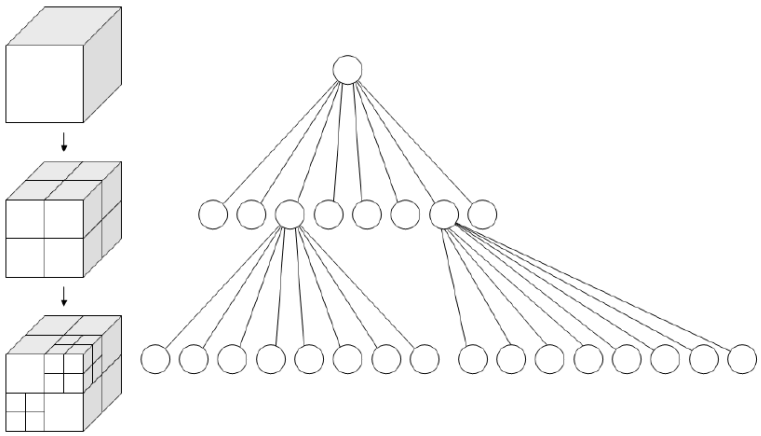

Figura 5.7: *Octree* de tres nodos y dos niveles de profundidad

En cada celda se representa un único punto de la nube, y cada nodo o del *octree* está asociado con una función F_o:

$$F_o(q) \equiv F\left(\frac{q - o.c}{o.w}\right)\frac{1}{o.w^3} \quad \textbf{(5.3)}$$

Donde $o.c$ y $o.w$ son respectivamente el centro y la anchura del nodo o.

La función implícita del modelo representa la suma de las funciones F_o de cada nodo. Por este motivo es necesario representar la ecuación anterior como la suma lineal de F_o:

$$F(q) = F\left(\frac{q}{2^D}\right) \quad \textbf{(5.4)}$$

Finalmente, es posible definir el campo vectorial $\vec{V}$ como:

$$\vec{V}(q) \equiv \sum_{x \in \chi} \sum_{o \in Ngbr_D(x)} \alpha_{o,s} F_o(q) s.\vec{N} \quad \textbf{(5.5)}$$

$Ngbr_D(x)$ corresponde a los ocho posibles nodos de una celda, y $\{\alpha_{o,x}\}$ a los valores de la interpolación trilineal en el caso de que los puntos vecinos no formen parte del *octree*.

Una vez definida la función implícita a partir del campo vectorial, la triangulación se realiza creando los triángulos teniendo en cuenta que aristas de la celda interseccionan con cada vértice de la nube.

En definitiva, la reconstrucción por *Poisson* genera nuevos vértices ayudando a un mayor densificado de nuestra nube, y éste se adapta muy bien a distribuciones aleatorias de puntos. Gracias en parte a la estructura de *octree*

Para realizar la reconstrucción por *Poisson* en MeshLab, después de haber calculado las normales en el anterior apartado, se debe ejecutar el menú **Filters > Point Set > Surface recostruction: Poisson.**

Este comando según su descripción "Usa los puntos y normales para construir una superficie usando la aproximación de reconstrucción de superficies de Poisson"

Para todas las reconstrucciones que vamos a llevar a cabo se utilizarán los valores en los parámetros que se indican en la Figura 5.8.

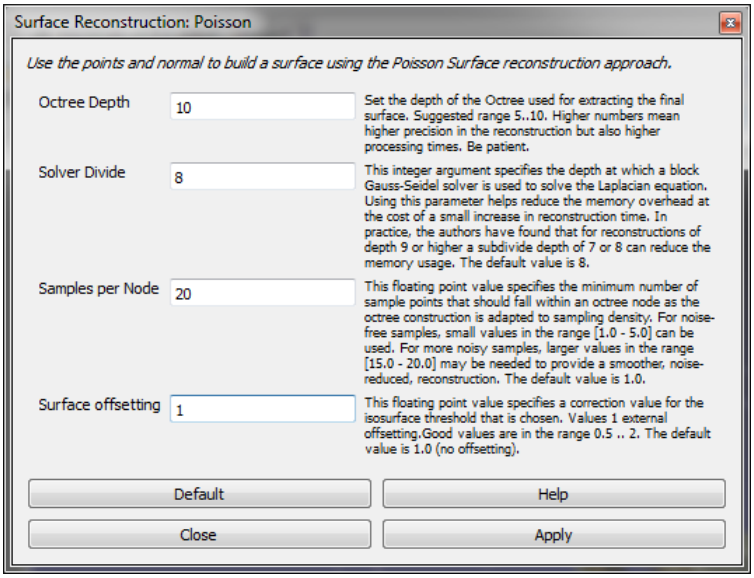

Figura 5.8: Parámetros de reconstrucción por *Poisson*

Estos valores son los más recomendables a la hora de crear una superficie a partir de una nube de puntos calculada con VisualSfM, debido a su alta aleatoriedad, gran densidad y superficies complejas.

El resultado obtenido después de la reconstrucción y la eliminación de algunas superficies se muestra en la Figura 5.9.

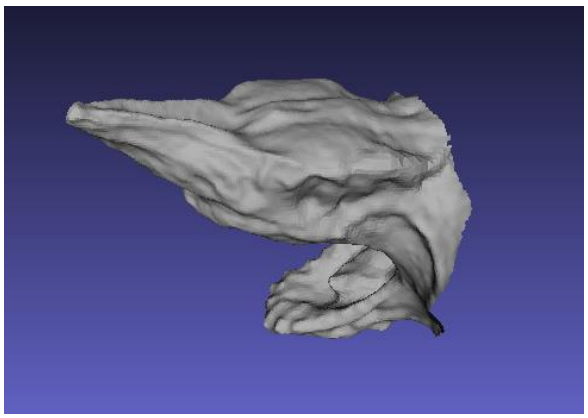

Figura 5.9: Superficie de la cabeza de tiburón.

Una vez obtenida la superficie, MeshLab nos permite guardarla en formatos: *.ply *.stl *.obj *.off *.wrl *.dxf *.dae *.ctm *.xyz, etc. Siendo los tres primeros los más utilizados en programas de software abierto para el modelado de superficies.

Una vez finalizado este proceso, ya podríamos imprimir el objeto en cuestión, utilizando el programa de nuestra impresora para el fileteado de la pieza. Estos programas crean superficies de apoyo y facilitan la impresión de las figuras creadas.

En el apartado de resultados comentaremos los parámetros seleccionados a la hora de imprimir las piezas y los factores que más se han tenido en cuenta.

6. FLUJO DE TRABAJO

Asentando todos los procesos comentados anteriormente, este apartado presenta el flujo de trabajo (Figura 6.1) que se deberá seguir para crear el modelo 3D impreso a partir de nuestro modelo real. Tendremos dos decisiones que tomar: aceptar la nube de puntos poco densa, comentada en el apartado 4.4 c), y analizar si la reconstrucción superficial es satisfactoria, comentada en el apartado 5.4.

Cuando la nube de puntos discreta, calculada por Visual SfM, es muy pobre, no podremos ver con claridad los rasgos más característicos de nuestra pieza. Por lo tanto, éste modelo no será aceptable. Puede deberse a que:

- Los factores fundamentales de la cámara son inadecuados, comentados en el apartado 3.
- El número de fotos tomadas ha sido escaso.
- Las superficies son inaceptables para este proceso. Ya sea porque son muy reflectantes o son de un color muy homogéneo.

Cuando la reconstrucción superficial no es satisfactoria, se debe a que hay zonas de nuestra nube que tienen muy poca densidad. Una baja densidad de puntos, en la reconstrucción de Poisson, se traduce en ondulaciones. Si el número de ondulaciones es pequeño, y éstas se encuentran en lugares no muy problemáticos, podremos suavizarlas por medio de programas de modelado superficial (Ej: Blender). Como lugares problemáticos nos referimos a las zonas que definan la forma característica de nuestro modelo.

También, aunque el modelo superficial sea satisfactorio, éste no tiene porqué ser el resultado final. Perfectamente puede verse modificado dependiendo de las decisiones del usuario.

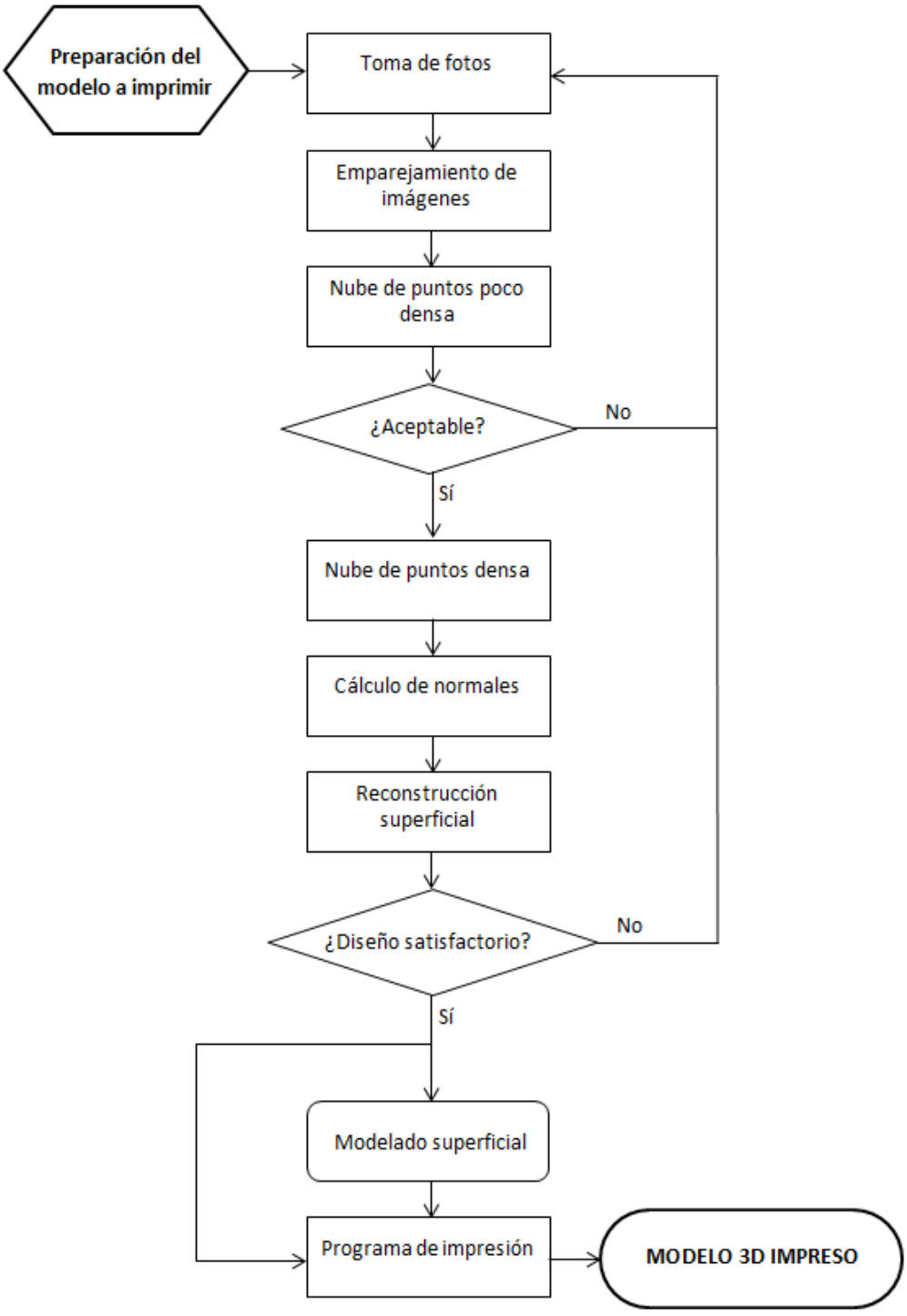

Figura 6.1: Flujo de trabajo del sistema de modelado 3D

7. RESULTADOS

Como resultado de las técnicas explicadas anteriormente se han impreso 3 modelos diferentes (Figura 7.1). En este apartado, comentaremos los parámetros de impresión, los datos obtenidos y dificultades generadas en el proceso.

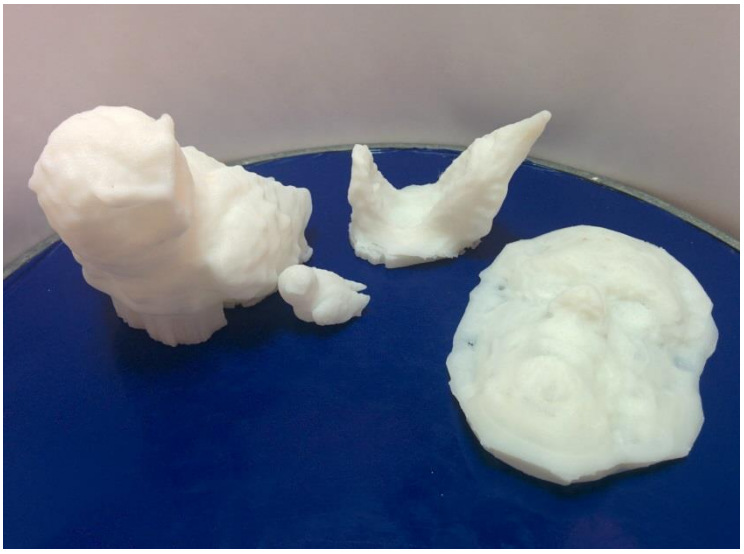

Figura 7.1: Conjunto de modelos 3D impresos

7.1. Parámetros de impresión

A la hora de imprimir se han seleccionado los siguientes parámetros:

Para todos los modelos:

- Altura de capa 0.2 mm
- Primera capa 0.3 mm
- Ventilador activado para capas tiempos>60 s, para la primera desactivado
- Temperatura extrusor 240 ºC
- Temperatura cama 80 ºC
- Dos capas en la base

- Relleno: Panal de abeja

Para el tiburón y el búho:

- Velocidad avance 30 mm/s
- Dos perímetros
- Dos capas en la parte superior

Para la cara:

- Velocidad avance 40 mm/s
- Tres perímetros
- Tres capas en la parte superior

El plástico utilizado ha sido PLA blanco, éste crea objetos con mayor rigidez y resistencia que otros plásticos, como por ejemplo el ABS. Pero como inconveniente, las tensiones creadas al enfriarse pueden crear deformaciones y estropear nuestra impresión. Además las superficies que hemos creado son complejas, aumentando la generación de tensiones. Por ello, es importante utilizar el ventilador para capas muy extensas, o si nuestra impresora no tiene ventilador, podemos reducir la velocidad de impresión para que el propio proceso convectivo enfríe la superficie.

7.2. Modelos

Los tres modelos impresos fueron:

- Búho

  En la Figura 7.2 vemos la figura real y el modelo 3D digital, donde podemos apreciar que a la hora de imprimirlo (Figura 7.3) recortamos la zona inferior para tener una base de apoyo más adecuada. El propio material de aporte crea la estructura que sustenta el objeto final.

Figura 7.2: Modelo Búho Real (Izquierda) y Modelo 3D digital (Derecha)

Figura 7.3: Modelo Búho 3D Impreso

- Cabeza de Tiburón

En la Figura 7.4, se aprecia que el modelado de la dentadura es imposible de captar debido al gran número de dientes y su disposición. Siendo una limitación del proceso fotogramétrico. Pero la resolución que nos ofrece la aproximación por Poisson es bastante acertada en todo el exterior.

La superficie superior de la cabeza ofrece un resultado excepcional, si la comparamos a la superficie real. Una buena iluminación y una superficie con gran número de zonas características son el punto fuerte de este proceso.

Figura 7.4: Modelo Cabeza de Tiburón Real (Izquierda) y Modelo 3D digital (Derecha)

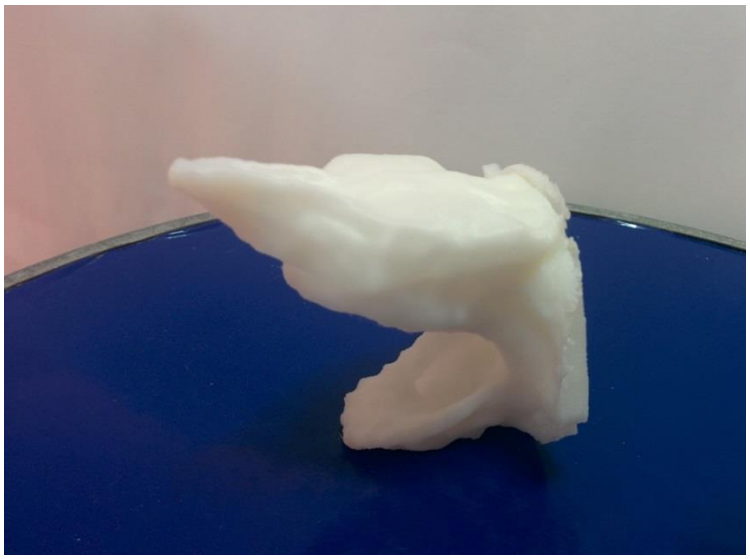

Figura 7.5: Modelo Cabeza de Tiburón 3D Impreso

- Cara

La obtención de un modelo facial no es fácil. Los brillos que refleja la piel y conseguir un buen enfoque son el mayor problema. Al estar iluminando con dos flexos es difícil tomar las fotos desde los ángulos adecuados para conseguir todos los puntos característicos. También un mínimo movimiento del modelo se traduce en puntos mal localizados. Aun así, la reducción de escala y los errores de impresión reducen las ondulaciones y generan un modelo (Figura 7.7) parecido al real (Figura 7.6).

Figura 7.6: Modelo Cara Real (Izquierda) y Modelo 3D digital (Derecha)

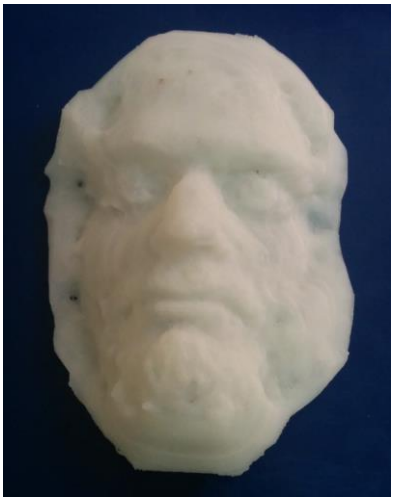

Figura 7.7: Modelo Cara 3D Impreso

7.3. Tabla de factores, características y tiempos

En este apartado, presentamos una tabla con todos los datos más característicos de todo el proceso que se ha seguido para cada modelo:

	Búho	Tiburón	Cara
Cámara	Nexus	Nexus	Canon EOS550D
Nº Fotos	42	64	42
Tamaño Fotos	3264x2448	3264x2448	5184x3456
Carga Fotos	9s	16s	24s
Emparejamientos	861	2016	861
	130s	114s	124s
Puntos clave	33562	30098	13930
	21s	29s	14s
Nube densa	667194	545734	923640
	2GB RAM	3.5GB RAM	5GB RAM
	328s	403s	1008s
Puntos útiles	75244	50040	64038
Nº Caras	150454	100078	123166
Tiempo Impresión	9h	5h	3h
Escala	1:1	1:2	1:2
Tiempo Total	10h	6h	4h30min

El tiempo total es una aproximación del tiempo empleado para todo el proceso de prototipado, teniendo en cuenta preparación, iluminación, transferencia de datos, cálculos e impresión. Como vemos en la tabla, la mayor parte del tiempo se emplea solo para la impresión del modelo. La digitalización de la superficie 3D suele durar entre 1h y 2h.

7.4. Posible modelado de grandes objetos

Aunque este resultado se aleja un poco del objetivo del trabajo, como curiosidad se realizaron las fotos de la turbina situada a la entrada de la Escuela de Ingenierías Industriales de Valladolid. Mediante 100 fotos y después de dos horas de cálculo Visual SfM consiguió crear una nube de puntos densa (Figura 7.8).

Figura 7.8: Nube puntos densa Turbina EII

Después de eliminar puntos no deseados el número de vértices de la nube que aparece en la figura es de 1615276. El problema de este modelo son las zonas que no se pueden fotografiar fácilmente, como la parte superior e inferior. Esto produce grandes huecos que a la hora de reconstruir la superficie crean grandes deformaciones inviables para su modelado e impresión.

8. CONCLUSIONES

A partir de los objetivos marcados se han podido obtener las siguientes conclusiones:

- Es posible crear un proceso de prototipado 3D a partir de software libre.
- Una buena captura de fotos es fundamental para conseguir un modelo final adecuado.
- El programa Visual SfM es una gran herramienta para convertir una serie de imágenes de un modelo real, en una nube de puntos digital tridimensional.
- El programa MeshLab crea superficies adecuadas para su posterior impresión 3D de una forma fácil y rápida.
- El flujo de trabajo que desarrollamos para el proceso de prototipado es un elemento indispensable para llegar a un resultado final adecuado.
- Los grandes inconvenientes se pueden enumerar en:
    I. Es necesario una iluminación difusa por toda la superficie.
    II. La superficie a modelar no puede ser reflectante, ni puede tener un color homogéneo.
    III. Para crear una nube lo suficientemente densa se necesitan un gran número de fotos.
    IV. La toma de fotos es difícilmente automatizable.
- La impresión del modelo 3D una vez calculada su superficie, y gracias a las impresoras de software libre, es relativamente sencillo. Dependiendo más de la calidad de la propia impresora que la forma del modelo en cuestión.
- Como conclusión final, cualquier persona que disponga de una cámara, un ordenador y una impresora 3D siguiendo los pasos de este proyecto puede desarrollar su propio proceso de prototipado 3D.

9. BIBLIOGRAFÍA

[1] P. U. J. M., «Modelado 3D patrimonio cultural por técnicas de s*tructure from motion» ph invesitgación,* vol. 1, pp. 77-87, (2013).

[2] C. Wu, «VisualSfM: A Visual Structure from Motion System» (2010).

Obtenido de: http://ccwu.me/vsfm/doc.html.

[3] K. Dom, «cuda-waste» (2013).

Obtenido de: https://code.google.com/p/cuda-waste/.

[4] D. G. Lowe, «Distinctive image freatures from scale-invariant keypoints» *Intl. Journal of Computer Vision,* pp. 60 (2):91-100, (2004).

[5] J. P. Yasutaka Furukawa, «Accurate, Dense, and Robust Multi-View Stereopsis,» *Fellow, IEEE, (*2008).

[6] S. M. Seitz, «"A comparison and evaluation of multi-view stereo reconstruction"» *CVPR, (*2006).

[7] R. K. a. O. D. F. J. P. Pons, «"Multi-view stereo reconstruction and scene flow estimation with a globarl image-based matching score"» *IJCV, vol. 72, no. 2,* p. 179–193, (2007).

[8] C. H. E. y. F. Schmitt, «"Silhouette and stereo fusion for 3D object modeling"» *CVIU, vol. 96, no. 3, (*2004).

[9] R. F. a. L. V. G. C. Strecha, «"Combined depth and outlier estimation in multi-view stereo"» *CVPR,* p. 2394–2401, (2006).

[10] M. L. a. L. Quan, «A quasi-dense approach to surface reconstruction from uncalibrated images» *PAMI, vol. 27, no. 3,* p. pp. 418–433, (2005).

[11] C. S. d. o. U. o. Pisa, «Meshlab» (2005).

Obtenido de: http://meshlab.sourceforge.net/.

[12] I. S. Alameda, Metodología para la gestión y explotación de datos espaciales obtenidos con sistemas de captura masica de puntos, Barcelona. Proyecto final de carrera, Ingeniería Técnica Topográfica, (2013).

[13] P. I. d. E. Landa, Documentación y representación de paramentos verticales en riesgo de colapso. Aplicación en el Monasterio de San Prudencio, (2011).

[14] M. Kazhdan, M. bolitho y H. Hoppe, «Poisson Surface Reconstruction» USA, (2006).