


UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR DE INGENIEROS EN TOPOGRAFÍA, GEODESIA Y CARTOGRAFÍA

GRADO EN TECNOLOGÍAS DE LA INFORMACIÓN GEOESPACIAL

# TRABAJO FIN DE GRADO

## RECONSTRUCCIÓN DE OBJETOS EN 3D MEDIANTE EL USO DE FOTOGRAMETRÍA Y APLICACIÓN A LA GENERACIÓN DE ENTORNOS EN VIDEOJUEGOS

Autor/a: Jorge Gala García

Tutor/a:

Andrés Díez Galilea

Tutor/a:

José Antonio López Medina

Madrid, junio, 2025

## Agradecimientos

A mis padres y a mi hermano por apoyarme en todo lo que hago y ser parte de mi vida

A mis amigos por estar ahí en cualquier momento. En Especial a Adrián Bosco Blanquisco y a Ernesto Hontecillas por su implicacion directa en el proyecto prestandome material de fotografía, a Mario Martín por aconsejarme con su conocimiento en videojuegos y a Javier Rico por ceder su coche para capturar en las pruebas iniciales.

A mis tutores por aceptar tutelar este trabajo “a ciegas” y guiarme durante su desarrollo.

A INES Ingenieros consultores por cederme parte de las imágenes para la realización de este proyecto, en especial a Illán Paniagua por la captura de las mismas.

## Resumen

En las últimas décadas, la tecnología ha experimentado un desarrollo y unos avances exponenciales. Se puede decir que a día de hoy, todo es mas potente que ayer y mas lento que mañana. Esta constante evolución ha impulsado mejoras en el ámbito de la computación gráfica y, en consecuencia, en el campo de los videojuegos, lo que ha resultado en un aumento de la demanda de entornos visuales más complejos y realistas por parte de los usuarios, obligando a los desarrolladores a encontrar nuevas tecnicas y tecnologías que permitan alcanzar nuevos niveles de detalle.

Es bajo este contexto cuando la fotogrametría, acompañada de la visión artificial, se presenta como una solución, permitiendo la generación de objetos y texturas a partir de imágenes del mundo real. Un ejemplo de su uso fue el videojuego Star Wars: Battlefront, donde se utilizó la fotogrametria para digitalizar escenarios y objetos del videojuego.

El objetivo de este proyecto es explicar los fundamentos de la fotogrametría, tanto a nivel teórico como práctico, necesarios para la generación de entornos y objetos digitales, así como el tratamiento posterior que dichos objetos requieren para su correcta utilización en un ejemplo aplicado. Todo ello se desarrolla desde la perspectiva del software libre, con el fin de minimizar los costes del proyecto y fomentar la accesibilidad y la reutilización de las herramientas empleadas.

[PALABRAS CLAVE]: Fotogrametría, Visión artificial, videojuego, entorno virtual, 3D, imagenes, características, descriptores, SfM, Meshroom, Blender, Unreal Engine

## Abstract

In recent decades, technology has undergone exponential development and progress. It can be said that today, everything is more powerful than yesterday and slower than tomorrow. This constant evolution has driven improvements in the field of computer graphics and, consequently, in the field of video games, which has resulted in an increase in user demand for more complex and realistic visual environments, forcing developers to find new techniques and technologies to achieve new levels of detail.

It is in this context that photogrammetry, accompanied by computer vision, is presented as a solution, allowing the generation of objects and textures from real-world images. An example of its use was the video game Star Wars: Battlefront, where photogrammetry was used to digitise scenarios and objects in the video game.

The aim of this project is to explain the fundamentals of photogrammetry, both theoretically and practically, necessary for the generation of digital environments and objects, as well as the processing that these objects require for their correct use in an applied example. All of this is developed from the perspective of free software, in order to minimize project costs and promote the accessibility and reuse of the tools employed.

[KEYWORDS]: Photogrammetry, Computer vision, video game, virtual environment, 3D, images, features, descriptors, SfM, MVS, Meshroom, Blender, Unreal Engine

# Indice de contenido

El autor del presente Trabajo declara ser responsable de la elaboración y autoría del mismo, asegurando con ello que no existe plagio, ni parcial ni total, ni el uso de medios externos de generación de contenido automático en la redacción de la memoria del TFT y garantiza que el contenido del TFT es fruto de su trabajo y, por tanto, de aportación propia y original.

# 1. MARCO TEÓRICO

Para comprender mejor la metodología expuesta en este proyecto, es importante conocer algunas ideas y conceptos clave. En este apartado se desarrolla la teoría utilizada de manera clara mediante el uso de fuentes contrastadas y actualizadas. Se parte de la explicación de los conceptos mas generales hasta el contenido más especializado.

## 1.1 VISIÓN ARTIFICIAL Y FOTOGRAMETRÍA, CONCEPTOS GENERALES

Antes de empezar con el proyecto, es necesario entender qué es la visión artificial. La visión artificial[1], también conocida como visión por ordenador o computer vision en inglés, es la disciplina que se centra en en el desarrollo de técnicas y algoritmos que permitan a las máquinas comprender imágenes y videos del mundo real.

Cuando los seres humanos observamos una imagen, somos capaces de distinguir formas o patrones que el cerebro es capaz de interpretar. Por el contrario, una máquina "observa" una imagen como una o varias matrices donde cada elemento representa un píxel. Ese píxel consiste en un valor numérico que corresponde con un valor de intensidad lumínica en ese espacio.

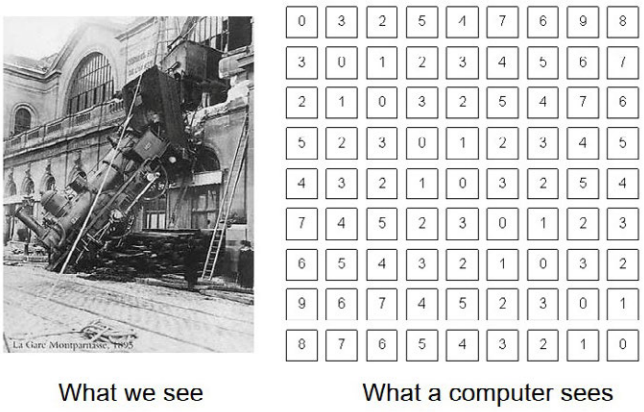


*Figura 1: Comparativa entre la visión humana y la visión artificial*

La visión artificial busca que las máquinas puedan percibir y comprender una imagen o secuencia de imágenes de manera similar a como lo hacen los humanos. Para ello, se apoya en otras disciplinas científicas:

- **Matemáticas**: La base de la visión artificial. Las imágenes digitales están compuestas por matrices de números que deben ser interpretadas y transformadas. Para ello se hace uso de disciplinas, como pueden ser el álgebra lineal, la probabilidad y la estadística y el cálculo diferencial.
- **Física**: Principalmente se usan los conceptos de óptica física y óptica geométrica para la formación de imágenes. Cuando se captura una imagen, se captura la luz que refleja un objeto. Esta luz, en forma de rayos, pasa por el objetivo (lente), haciendo que converjan en el plano focal dando lugar a una imagen invertida. Si el objeto esta correctamente enfocado, el plano focal coincidirá con los sensores fotosensibles encargados de transformar esa luz en información digital. La cantidad de luz que entra en la cámara, se regula mediante la apertura y la velocidad del obturador. Esto se explicará en detalle más adelante.

[1] *Overview of Computer Vision, Tanvir Siddike; Visión artificial, Wikipedia; CS231n, Universidad de Standford*

- **Biología**: Muchos de los algoritmos utilizados en visión artificial emulan o se inspiran en el funcionamiento del sistema visual, así como los procesos cognitivos que intervienen en el reconocimiento de imágenes.
- **Psicología**: Se apoya en el uso de teorías sobre la percepción y el reconocimiento de patrones para comprender cómo los humanos interpretan la información visual.
- **Ciencias de la computación**: Es la responsable de la creación de algoritmos, así como de la creación de arquitecturas de computación eficientes para el manejo de grandes volúmenes de datos.
- **Ingeniería**: Diseño de nuevo hardware y tecnología que permite el avance de la ciencia, así como su reaplicación en otras ramas que encuentran utilidad en este campo.

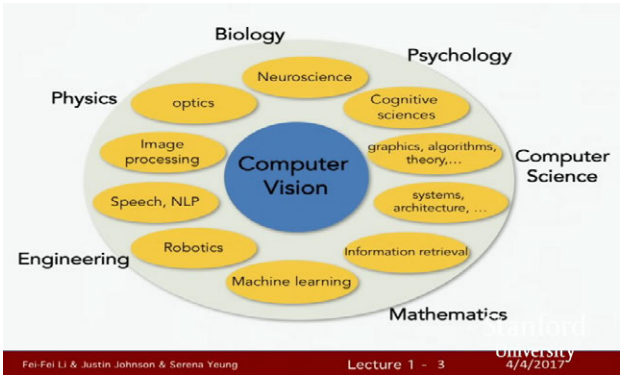


*Figura 2: Campos con competencias en la visión artificial*

La visión espacial da el contexto necesario para entender una de las bases principales de este trabajo, la fotogrametría. La fotogrametría es una ciencia, arte o técnica cuyo objetivo consiste en estudiar y definir con precisión la forma, dimensiones y posición en el espacio de un objeto mediante la medida e interpretación de imágenes, la cual se puede definir como una proyección de una escena 3D sobre un plano 2D.

Al momento de capturar una imagen, la información de profundidad asociada a esa escena se pierde. La fotogrametría se encarga de invertir ese proceso. Explicado de manera sencilla, el proceso del método fotogramétrico se basa en tomar imágenes (o vistas) del mismo objeto o entorno desde diferentes posiciones (o poses) para así poder estimar su profundidad a partir de la perspectiva.

*Figura 3: Vistas y poses de un objeto reconstruido*

Aunque a priori la visión artificial y la fotogrametría puedan parecer lo mismo, sus aplicaciones no lo son. La visión artificial se centra en la búsqueda de técnicas que

permitan identificar escenas de manera general, mientras que la fotogrametría tiene un propósito mas especializado, centrándose en definir de manera precisa las dimensiones de los elementos que conforman la escena. Se encuentran relacionadas en el sentido de que la fotogrametría utiliza técnicas de la visión artificial para lograr este objetivo.

Estas dos ciencias combinadas son las bases que permiten la resolución del problema de la reconstrucción de objetos en 3D. En el siguiente marco teórico se plantea un *pipeline* general que describe el problema en profundidad, citando a la vez alternativas.

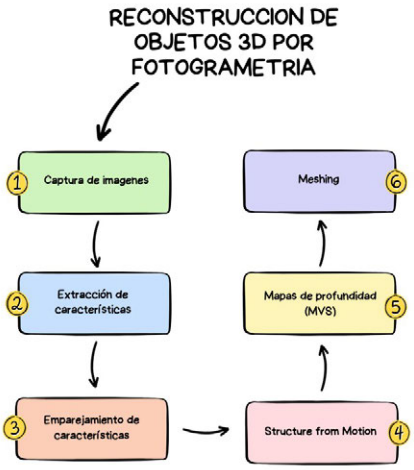


*Figura 4: Esquema general de reconstrucción de objetos 3D*

## 1.2 CAPTURA DE IMÁGENES

La captura de imágenes es el primer paso, y a la vez uno de los mas importantes de la reconstrucción por el método fotogramétrico. Consiste en tomar fotografías de una escena que se quiere capturar con la máxima cobertura y grado de detalle posible. Para ello, existen una serie de pautas a seguir:

- **Cámara:** Se recomienda el uso de cámaras digitales de alta resolución, con un mínimo de 12 MP. Por lo general, cuanto mayor sea el sensor, mayor es la cantidad de luz que puede capturar en un menor tiempo de obturación, por lo que mayor calidad de imagen. Para las lentes, se recomienda el uso de una focal fija, que se encuentre entre los 18 mm y los 50 mm. Estar por debajo de 35mm comienza a generar la conocida como distorsión de barril, mientras que estar por encima de los 70mm distorsión de cojín[2]. Estas distorsiones se magnifican en los bordes de la imagen:

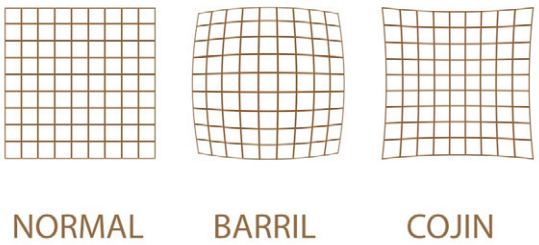


*Figura 5: Distorsiones asociadas a la lente*

[2]Recomendable leer *What is focal length in photography? A guide for beginners, Nikon, 2024* para entender las distintas distorsiones opticas.

La cámara utilizada también debe poder guardar metadatos en las imágenes. Estos metadatos, conocidos como parámetros EXIF, codifican información útil de la imagen, de la cámara y de otros aparatos utilizados en el momento de la captura. Existen mas de 700 etiquetas EXIF[3] (no todas están comprendidas dentro del estándar[4]). Para la reconstrucción de objetos 3D, se consideran estas como las mas necesarias:

| Etiqueta | Descripción | Comentarios |
|---|---|---|
| Make | marca | Utilizados para ver si el modelo se encuentra registrado en una base de datos y asi poder obtener sus parametros intrinsecos directamente |
| Model | modelo | |
| SerialNumber | Numero de serie de la cámara | |
| FocalLength | Distancia focal en mm | - |
| FocalLengthIn35mmFilm | Distancia focal en formato de 35 mm | Opcional. Si no existiera la camara en la base de datos, se usa para calcular el ancho y el alto del sensor |
| FocalPlaneXResolution | Ancho del sensor en pixeles | - |
| FocalPlaneYResolution | Ancho del sensor en pixeles | - |
| FocalPlaneResolutionUnit | Unidades de dimensiones del sensor | - |
| ImageWidth | Ancho de la imagen en pixeles | - |
| ImageHeight | Alto de la imagen en pixeles | - |
| GPSLatitude | Latitud GPS | Opcional, depende de si se guarda la ubicación o de si existe algun dispositivo auxiliar, como un RTK |
| GPSLongitude | Longitud GPS | |
| GPSAltitude | Altitud GPS | |

*Tabla 1: Parámetros EXIF necesarios*

Estas etiquetas dan información para la calibración de las cámaras. Se dice que una cámara esta calibrada cuando se tienen su parámetros intrínsecos. Como se verá mas adelante, contar con parámetros intrínsecos ayuda a simplificar la reconstrucción.

Por último, conviene evitar el uso simultáneo de cámaras de distintos modelos. Incrementar el número de dispositivos, como se verá más adelante, aumenta la complejidad de la reconstrucción. A parte, cada sensor suele capturar la luz de una forma distinta o contar con calibraciones de color específicas, lo que genera variaciones que afectan a la homogeneidad del resultado final.

- **Condiciones de iluminación:** Lo ideal es contar con una escena completamente iluminada y de manera homogénea. Estas condiciones se pueden conseguir fácilmente en interiores con la ayuda de fuentes de luz artificial, pero en exteriores resulta más complicado, ya que se depende mayoritariamente de la luz ambiental, condicionada por el clima y otros elementos del entorno que puedan distorsionar las condiciones lumínicas.

  En base a la experiencia obtenida durante la realización de este trabajo, en exteriores se recomienda evitar las horas de sol bajas, ya que se pueden producir reflejos de lente (*lens flare*) que pueden alterar las condiciones de iluminación de la imagen por incidencia directa del sol (subexposición por contraluz).

*Figura 6: Comparación de subexposición por contraluz (Figura Alcorcón 1)*

De la misma forma, las horas de luz altas permiten una buena iluminación, pero si el objeto reflexiona la luz, puede dar lugar a reflejos y efectos indeseados que

[3]Etiquetas registradas por *exiftool.org*
[4]*CIPA DC-008-2024*

provoquen inconsistencias entre imágenes. Esto, como se verá mas adelante, evita que se encuentren coincidencias entre imágenes. En la siguiente comparación de imágenes, se observa la reconstrucción de un coche en la que la carrocería y los cristales producen reflejos, haciendo que esta resulte fallida. Se observa también que las zonas sin reflejos y con bordes bien definidos se reconstruyen correctamente:

*Figura 7: Reconstrucción fallida de un coche por exceso de reflexión*

Lo ideal, sería tomar las imágenes con clima nublado, ya que los rayos de sol se difuminan. En su defecto, se podrían utilizar fuentes de iluminación artificial, como focos o elementos que bloqueen los reflejos, como filtros para la lente. En este trabajo se capturan escenas en distintas condiciones de luz para estudiar su viabilidad.

- **Captura de imágenes:** La captura de imágenes debe realizarse de toda la escena y parte de su entorno (si es posible, ya que facilita la reconstrucción). Las imágenes deben estar solapadas entre si, donde se recomienda contar de manera general con al menos un 75% de solapamiento longitudinal y un 60% de solapamiento transversal[5]. Se recomienda capturar las imágenes con el valor de ISO y la velocidad de obturación más bajos posible. Aumentar el ISO puede provocar la aparición de ruido en las imágenes, mientras que bajar la velocidad de obturación puede provocar que las imágenes salgan movidas[6].

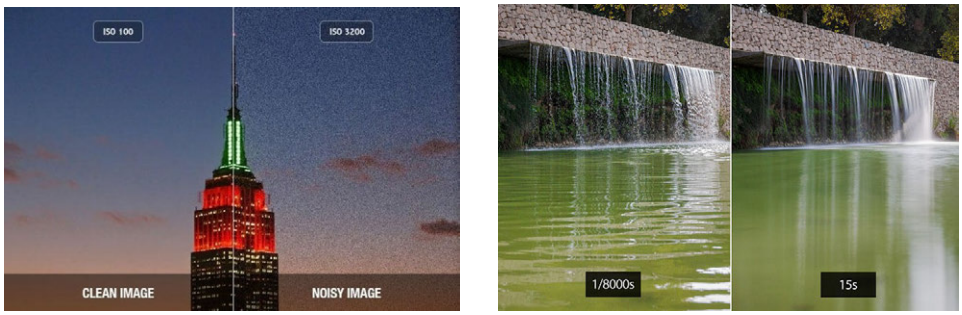


*Figura 8: ISO (skylum.com) y velocidad de obturación (dzoom.org)*

Por último, hay que tener en cuenta la distancia a la que se fotografía la escena. Cuanto mas cerca se capture las escena, mayor será el nivel de detalle, lo que obliga también a tomar mayor número de imágenes para cubrir la escena. Una imagen digital es un elemento finito compuesto por píxeles, por lo que, en función de la distancia de captura, estos representan un tamaño en la realidad:

[5]Recomendable leer *How to verify that there is enough overlap between the images, PIX4D*
[6]Recomendable leer *Efectos del ISO y la obturación sobre la exposición, ADOBE*.

$$GSD = \frac{\sqrt{d^2+h^2} * \frac{W_{sensor}}{W_{imagen}}}{f * \cos(\Theta)}$$

*Ecuación 1: Formula general de Ground Sampling Distance, Wikipedia*

Donde:

- *d* es la distancia horizontal al nadir
- *h* es la altura del sensor al suelo
- $W_{sensor}$ es el ancho del sensor en mm (se puede usar el largo también)
- $W_{imagen}$ es el ancho de la imagen en pixeles (se puede usar el largo también)
- *f* es la distancia focal en mm
- $\Theta$ es el angulo de inclinación desde el nadir

Con estas recomendaciones generales, se puede comenzar a capturar las imágenes de la escena. La fotogrametría plantea 3 técnicas de captura de imágenes en función de la posición de la cámara. La elección de estas depende de las características de la escena a capturar, así como del material disponible:

- **Fotogrametría aérea**: La cámara se encuentra en el aire, montada en una aeronave. Esta aeronave realiza pasadas a velocidad constante capturando imágenes de la superficie.

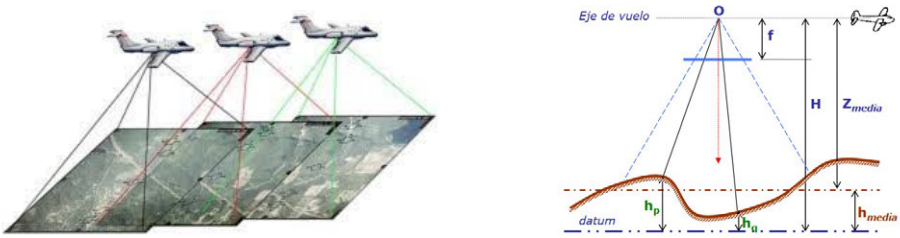


Figura 9: Representación de fotogrametría aerea8 9

En función del ángulo de inclinación que tenga el eje principal con la vertical, se puede distinguir entre:

1. **Fotogrametría aérea vertical:** El eje principal sigue la dirección de la vertical, siendo prácticamente perpendicular al suelo (Inclinaciones entre los 0º y 3º). Útil para el diseño de cartografía y ortofotos.

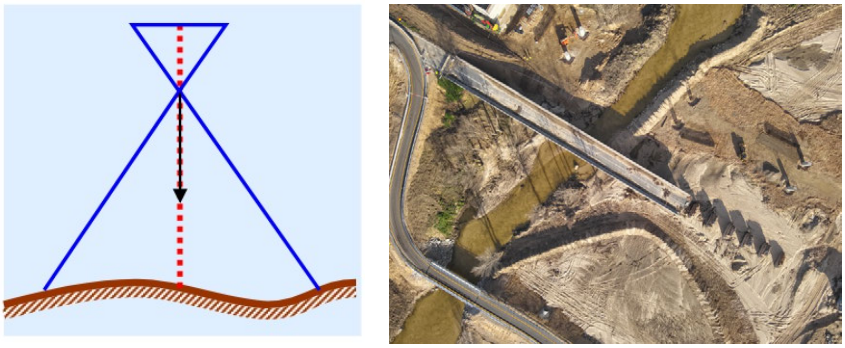

Figura 10: Representación fotogrametría aérea vertical (Pte. de la Pedrera, INES)

2. **Fotogrametría aérea inclinada u oblicua:** El eje principal se encuentra inclinado respecto a la vertical (Inclinación arbitraria). Se puede diferenciar entre oblicua baja y oblicua alta en función de si el horizonte aparece o no en las imágenes. Muy utilizado especialmente en la inspección de estructuras y reconstrucciones espaciales.

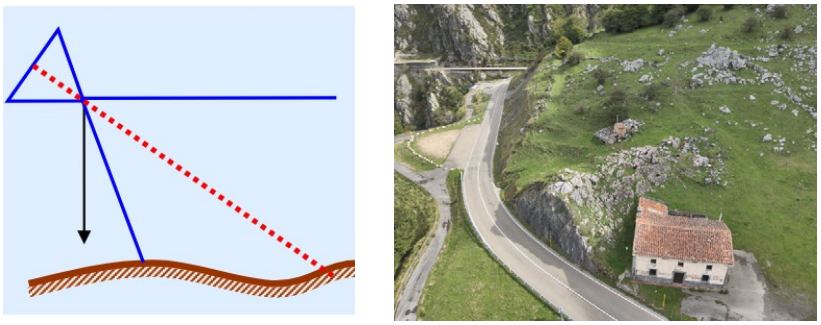

Figura 11: Representación fotogrametría aérea oblicua (Puerto San Isidro, INES)

- **Fotogrametría terrestre o de proximidad**: La cámara se encuentra al nivel del suelo, ya sea montada sobre un trípode, en un vehículo (mobile mapping) o sujetada por una persona.

Figura 12: Representación fotogrametría terrestre

En función de la posición de la cámara al momento de capturar la imagen, se puede clasificar en:

1. **Fotogrametría terrestre convergente**: Se toman imágenes desde múltiples posiciones a un mismo objeto. Se busca rodear este para tener una cobertura total.
2. **Fotogrametría terrestre panorámica**: Se toman imágenes desde un mismo punto de vista en múltiples direcciones. Útil en la captura de fachadas o escenas de gran amplitud que no interesa capturar enteras.
3. **Fotogrametría terrestre móvil**: La cámara se encuentra montada en un vehículo o plataforma que permite su movimiento. Se suele usar en la captura de elementos lineales largos, como túneles y carreteras. Se combina con otras tecnologías, como sensores LiDAR.

En este proyecto se hace uso de técnicas de fotogrametría terrestre convergente y panorámica, así como de fotogrametría aérea oblicua y vertical. Así mismo, se experimenta con distintos escenarios de iluminación y tipos de cámaras (incluyendo mezcla de cámaras) con el objetivo de analizar los resultados que pueden obtenerse.

## 1.3 EXTRACCIÓN DE CARACTERÍSTICAS

Como se ha comentado anteriormente, el método fotogramétrico se basa en el uso de vistas de un mismo objeto o entorno tomadas desde diferentes puntos de vista para poder calcular su profundidad. El primer problema que se presenta, como ya se ha explicado en el apartado de la visión artificial, es que una máquina no es capaz de identificar un objeto como lo haría una persona, por lo que es necesario establecer una aproximación computacionalmente eficiente. Una estrategia habitual consiste en la extracción de características locales.

Una característica es una zona que puede ser distinguible dentro de una imagen (una esquina, un borde, etc) permitiendo identificar y correlacionar entidades geométricas entre distintas imágenes. Está compuesta por su ubicación en la imagen (punto de interés o keypoint) y por un vector de información de sus alrededores que ayuda a caracterizar su información visual y poder compararla con otras características (descriptor).

Existen numerosos algoritmos de detección de características como pueden ser SIFT, KAZE, AKAZE, SURF, ORB o BRIEF. En este proyecto se centra su atención en la utilización de SIFT y sus distintas variables.

[7]SIFT, también conocido como *Scale Invariant Feature Transform* permite la obtención de características (detector y descriptor) a partir de la detección de *blobs*[8] sin verse afectado por posibles cambios de escala. Para ello sigue 4 fases:

- **Detección de extremos del espacio de escala:** Busca candidatos a puntos de interés en todas las posibles escalas utilizando la diferencia de Gaussianas, la cual se calcula como la diferencia de dos escalas consecutivas separadas por un factor *k.*

$$D(x,y,\sigma)=\big(G(x,y,k\sigma)\big)-\big(G(x,y,\sigma)*I(x,y)\big)=L(x,y,\sigma)-L(x,y,\sigma)$$

*Ecuación 2: Detección de candidatos puntos de interés basada en diferencia de Gaussianas*

Para detectar los extremos (máximos y los mínimos locales), los puntos de interés se comparan con sus 8 vecinos más cercanos en esa imagen y con los 9 vecinos más cercanos en la imagen de escala superior e inferior. Si el punto es mayor o menor que los 25 vecinos, es considerado como un extremo y pasa a ser un candidato a punto de interés.

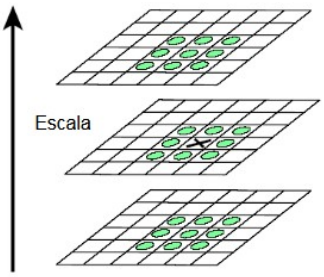


Figura 13: Candidatos a puntos de interés comparados con sus vecinos

[7]Este apartado se encuentra íntegramente basado en *Distinctive Image Features from Scale-Invariant Keypoints, Lowe* y en su adaptación al español *Capítulo 8: SIFT (Scale Invariant Feature Transform) Enrique Alegre, Laura Fernandez-Robles*

[8]*Capítulo 8: SIFT (Scale Invariant Feature Transform) Enrique Alegre, Laura Fernandez-Robles* define *blob* como una región de una imagen que se caracteriza porque algunas de sus propiedades se mantienen aproximadamente constantes. Se recomienda su lectura para profundizar en su funcionamiento.

- **Localización detallada del punto de interés:** Para cada candidato a punto de interés, se ajusta un modelo que defina de forma detallada su localización en la imagen y su escala (nivel piramidal):

  La localización en la imagen se determina utilizando el desarrollo de Taylor de segundo orden de la función del espacio de escala en el punto a evaluar:

$$D(x) = D + \frac{\partial D}{\partial x}^T x + \frac{1}{2} x^T \frac{\partial^2 D}{\partial x^2} x$$

*Ecuación 3: Calculo de posición del punto de interés, Taylor segundo orden*

  Al calcular las derivadas de dicha función respecto a x e igualando a 0, se obtiene una función que permite rechazar extremos con bajo contraste:

$$D(\hat{x}) = D + \frac{1}{2} \frac{\partial D}{\partial x}^T \hat{x}$$

*Ecuación 4: Función de extremos del desarrollo de Taylor*

  Según el modelo general descrito inicialmente por Lowe, se considera que para una imagen con valores comprendidos entre [0, 1] en escala de grises, se descartan todos los valores de $|D(\hat{x})| < 0.03$

  Ademas, es necesario tener en cuenta que debido a la naturaleza de la diferencia de gaussianas, existe la posibilidad de que los candidatos a puntos de interés a lo largo de los bordes presenten una respuesta (curvatura) muy fuerte, incluso si estos tienen un bajo contraste. Para filtrar estos puntos, se hace uso de una relación entre la traza y el determinante del Hessiano:

$$H = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{yx} & D_{yy} \end{bmatrix} \rightarrow \begin{array}{c} Tr(H) = D_{xx} + D_{yy} = \alpha + \beta \\ Det(H) = D_{xx} D_{yy} - (D_{xy})^2 = \alpha \beta \end{array}$$

*Ecuación 5: Matriz Hessiana, traza y determinante*

  Donde $\alpha$ y $\beta$ son los autovalores (valores propios) del Hessiano, siendo $\alpha$ el mayor y $\beta$ el de menor.

$$\begin{gathered} det(H - \lambda I) = 0 \\ (D_{xx} - \lambda)(D_{yy} - \lambda) - D_{xy}^2 = 0 \\ \lambda^2 - (D_{xx} + D_{yy})\lambda + (D_{xx} D_{yy} - D_{xy}^2) = 0 \\ \lambda_{mayor} = \alpha \\ \lambda_{menor} = \beta \end{gathered}$$

*Ecuación 6: Desarrollo del cálculo de autovalores*

  Estos representan los valores de curvatura principales en las dos direcciones perpendiculares del punto. Se considera que un punto de interés está bien definido cuando ambos valores son grandes, que esta mal definido si tiene un valor $\alpha$ grande y un valor $\beta$ pequeño (una curvatura muy fuerte y otra casi plana) y que tiene bajo contraste si ambos son pequeños. Calcular directamente los autovalores para todas las características y compararlos entre sí sería muy costoso a nivel computacional, por lo que se utiliza la siguiente relación entre ellos:

$$r=\frac{\alpha}{\beta}\rightarrow\frac{Tr(H)^2}{Det(H)}=\frac{(r\beta+\beta)^2}{r\beta^2}=\frac{(r+1)^2}{r}$$

*Ecuación 7: Relación entre la traza y el determinante*

$r$ actúa como un umbral arbitrario, de forma que si en el calculo de la relación, al despejar su valor si este resulta superior a 10 (valor establecido en el documento inicial de Lowe), el punto de interés se descarta.

Todo este procedimiento consigue el filtrado de aquellos candidatos próximos a lo bordes o con poco contraste (sensibles al ruido).

- **Asignación de la orientación:** Para cada punto de interés resultante, se asigna una o mas orientaciones basadas en las direcciones del gradiente local. Esto hace que sea invariante a cambios de orientación.

  Se comienza calculando la magnitud y la orientación del gradiente en cada pixel de la imagen $L(x,y)$, siendo esta la imagen del punto de interés en la escala de este, a la que se le ha aplicado un suavizado Gaussiano:

$$\|\nabla L(x,y)\|=\sqrt{(L(x+1,y)-L(x-1,y))^2+(L(x,y+1)-L(x,y-1))^2}$$
$$\theta(x,y)=\tan^{-1}\left(\frac{L(x,y+1)-L(x,y-1)}{L(x+1,y)-L(x-1,y)}\right)$$

*Ecuación 8: magnitud del gradiente y rotación[9], diferencia de píxeles*

  Posteriormente se construye un histograma de 36 intervalos (de 0º a 360º en intervalos de 10º), el cual es poblado con las rotaciones calculadas. Estas han sido previamente ponderadas mediante el uso de la función gaussiana bidimensional, de forma que tengan mas valor las que se encuentren próximas al punto característico:

$$G(x,y;x_0,y_0;\sigma)=A\exp\left(-\left(\frac{(x-x_0)^2+(y-y_0)^2}{2\sigma^2}\right)\right)$$

*Ecuación 9: función Gaussiana bidimensional[10]*

  Donde:

  - *A* es la amplitud de la curva. En este caso de uso, equivale a 1
  - $\sigma$ equivale 1.5 veces la escala del punto de interés
  - ($x_0$, $y_0$) es la ubicación en pixeles del punto de interés
  - *(x, y)* son los pixeles vecinos

  Los picos resultantes en el histograma coinciden con las orientaciones dominantes en el entorno del punto de interés. Se elige la orientación del pico más alto y aquellas que superen el 80% de este. La orientación final se interpola ajustando una parábola a los 3 valores del histograma mas próximos al mayor. Con esto, el punto de interés finalmente queda perfectamente definido e invariable. Ahora, toda calcular su descriptor asociado.

[9]Se ha cambiado la notación de la magnitud del gradiente para que no haya conflictos con la matriz de proyección de Structure from Motion

[10]Wikipedia, adaptada ya que σ_x y σ_y en este caso son iguales

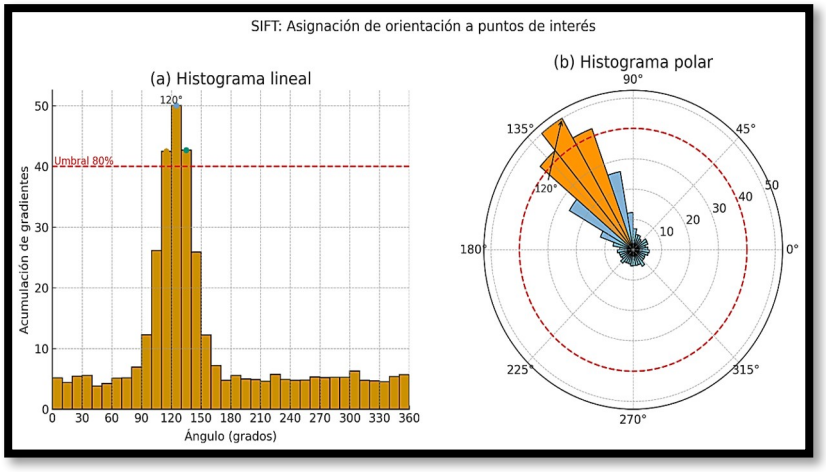


Figura 14: Asignación de la orientación a puntos de interés

- **Calculo del descriptor**: Como se ha comentado anteriormente, un descriptor es un vector que contiene información detallada sobre el entorno del punto de interés que permite diferenciarlo de otras características. El planteamiento de su cálculo es muy similar al de la asignación de la orientación:

  1. Se calcula la magnitud y la orientación del gradiente en una ventana 16x16 con centro en el punto característico. Cada una de estas 16 regiones esta a su vez dividida en 4x4 muestras (pixeles). Al igual que en el paso anterior, se realiza para la imagen en la escala del punto característico habiendo sido previamente suavizada [*Ecuación 8*].

  2. Para cada una de las 16 regiones de la ventana se plantea un histograma de 8 intervalos (de 0º a 360º en intervalos de 45º). En este caso, estos histogramas son poblados por la suma de las magnitudes de los gradientes que se encuentren en el correspondiente intervalo de orientación[11]. Asignar la totalidad de la magnitud a un intervalo puede provocar sesgos por discretización[12], por lo que los valores son redistribuidos entre sus intervalos vecinos mediante interpretación tri-lineal.

  3. Se obtiene un vector de 128 elementos resultado de concatenar los 16 histogramas de 8 valores (4 x 4 x 8 = 128) y se normaliza. El proceso de normalización consiste en realizar una primera normalización:

$$\hat{u} = \frac{u}{\|u\|}$$

*Ecuación 10: Normalización de un vector*

  Para evitar la predominancia de gradientes grandes, se limitan los valores a un umbral de 0,2. Posteriormente, se vuelve a normalizar.

  Con esto, se obtiene el descriptor asociado al punto de interés, quedando la característica completamente definida.

[11]Ejemplificación: Si un pixel tiene orientación 20º, su magnitud va al intervalo 0º – 44º. Se suman en función de *hacia donde apuntan.*

[12]Pueden existir magnitudes con orientación en 43,9º. Estas se encontrarían contenidas en el intervalo 0º - 44º

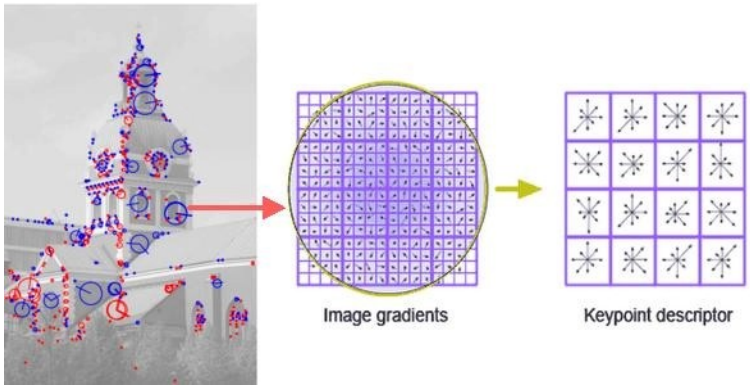


Figura 15: Representación de la obtención de un descriptor[13]

Este procedimiento lo convierte en un método de extracción robusto e invariante frente rotaciones de la imagen, transformaciones afines, cambios en la intensidad o cambios en la perspectiva. Su principal problema es el alto coste computacional que conlleva, derivado principalmente del uso de operaciones Gaussianas.

La extracción de características se hace para cada una de las imágenes que interviene en la creación del modelo. Posteriormente, se busca emparejar estas para obtener la posición relativa de las imágenes respecto de las otras.

## 1.4 EMPAREJAMIENTO DE IMÁGENES Y CARACTERÍSTICAS

Una vez que se han extraído las características de las imágenes, se procede a su emparejamiento. Este proceso consiste en encontrar las mismas características en un par de imágenes para posteriormente poder estimar la posición de estas en el espacio tridimensional, o dicho de otra forma, encontrar parejas de imágenes que contengan las mismas zonas u objetos de la escena 3D a partir de coincidencia de puntos. Como se ha visto en el anterior apartado, las características obtenidas mediante SIFT resultan invariantes ante cambios de escala no lineares, lo que hace posible el emparejamiento en perspectivas distintas[14].

Comparar todas las combinaciones posibles de características (fuerza bruta o *brute force*) no sería eficiente, entre otras cosas debido a la complejidad del descriptor y al numero de combinaciones resultantes[15]. Una solución a esto es el uso de arboles de vocabulario (Vocabulary Tree)[16]. Un árbol de vocabulario en una estructura en forma de árbol formada por un conjunto de descriptores cuantificados en "*palabras visuales*"[17] mediante el algoritmo de clustering no supervisado k-means. En este caso, k define el numero de hijos que tiene cada nodo, que corresponde con una palabra visual. Este se ramifica hasta un nivel L arbitrario:

[13] *Scale-Invariant Feature Transform (SIFT), Mohammad Yawar* vía *naukri.com*,
[14] Este apartado depende en gran medida del tipo de descriptor utilizado y del programa o pipeline a utilizar, por lo que solo se generaliza.
[15] Meshroom con sus parámetros por defecto, saca un target 20000 características por imagen. En una reconstrucción, dependiendo del objeto se puede llegar a mas de 500 imágenes.
[16] *Scalable Recognition with a Vocabulary Tree, David Nistér and Henrik Stewénius.* Es la implementación usada por Meshroom.
[17] Una palabra visual es un término textual, que no tiene por que tener significado. Es más fácil comparar palabras que vectores.

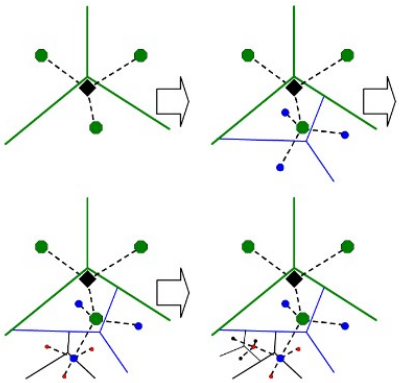

Figura 16: Representación de la creación de un árbol de vocabulario

A nivel practico, para cada imagen se elabora un histograma de las palabras visuales, en los que la frecuencia ha sido ponderada mediante TF-IDF[18]. Estos se comparan entre sí mediante en un sistema de puntuación (Norma L1. Cuanto menor es la puntuación, mayor es la similitud) que permite ordenar las imágenes por similitud.

Teniendo las imágenes ordenadas por similitud, se puede comenzar con la comparación de características. Para ello, se hace uso de la regla de Lowe (*Lowe's Ratio test*)[19]:

$$\frac{d_1}{d_2} < r$$

*Ecuación 11: Regla de Lowe*

En esta, se compara una característica de la imagen $I_1$ con los dos vecinos mas cercanos de su posible pareja $I_2$, donde:

- $d_1$ es la mínima distancia del descriptor de la imagen $I_1$ (definido como *q*) al de la imagen $I_2$ (definido como *x*).
- $d_2$ es la segunda mínima distancia del descriptor de la imagen $I_1$ (definido como *q*) al de la imagen $I_2$ (definido como *x*).

$$d_1 = min_{k=1} \left\| q - x_k \right\|$$
$$d_2 = min_{k=2} \left\| q - x_k \right\|$$

*Ecuación 12: Calculo de distancias mínimas entre descriptores*

En el documento oficial de Lowe, se establece que *r* no debe ser menor de 0,8. Este proceso no esta exento de posibles outliers. Una opción para eliminar estos es aplicar el mismo procedimiento de manera recíproca o el uso del algoritmo de eliminación de outliers RANSAC sobre las restricciones epipolares que existen entre los distintos pares de imágenes. Más adelante se explicará el concepto de geometría epipolar.

Con esto las imágenes quedarían correctamente emparejadas, dando paso a la base del proceso fotogramétrico, Structure from Motion.

[18] *Term frequency – Inverse document frequency*, es una medida numérica que expresa cuán relevante es una palabra para un documento en una colección. *Wikipedia*

[19] *Distinctive Image Features from Scale-Invariant Keypoints, Lowe.* Apartado 7.1

## 1.5 STRUCTURE FROM MOTION (SFM)

Structure From Motion (Estructura Desde el Movimiento o SfM) es donde la visión artificial y la fotogrametría se entrelazan. Como se ha comentado anteriormente, al capturar una imagen, se está proyectando una escena 3D sobre un plano 2D, perdiendo la información de profundidad. SfM es el "problema inverso". Consiste en reconstruir un objeto en 3D a partir de imágenes tomadas desde diferentes puntos de vista, a la vez que se estima la posición y la dirección de las cámaras (también conocidas como vistas o imágenes).

En un enfoque general, la resolución de este problema consiste en la optimización de una función de coste, conocida como **error de reproyección total** (*total reprojection error*), mediante el uso de técnicas de **ajuste por haces** (*bundle adjustment).* Para ello*, s*e parte del siguiente planteamiento matemático[20]:

$M_1, \dots, M_p$ son matrices (4x1) con las **coordenadas 3D de los puntos de la escena homogeneizadas referidas al mundo**, es decir, en un sistema de coordenadas común a todas las cámaras, donde *p* es el numero total de puntos. Estas coordenadas son desconocidas.

$$M_j = \begin{bmatrix} X_j \\ Y_j \\ Z_j \\ 1 \end{bmatrix} \in I\!R^4$$

*Ecuación 13: Matriz de coordenadas de los puntos de la escena homogeneizados*

$P_1, \dots, P_n$ corresponde con las distintas **matrices de cámara**, donde *n* es el número total de cámaras (o imágenes). Estas matrices también son desconocidas. Cuando las cámaras **no se encuentran calibradas**, estas se representan como una matriz de proyección de dimensiones 3 x 4 y de rango 3 definida hasta la escala:

$$P_i = \begin{bmatrix} p^{(i)}_{11} p^{(i)}_{12} p^{(i)}_{13} p^{(i)}_{14} \\ p^{(i)}_{21} p^{(i)}_{22} p^{(i)}_{23} p^{(i)}_{24} \\ p^{(i)}_{31} p^{(i)}_{32} p^{(i)}_{33} p^{(i)}_{34} \end{bmatrix} \in I\!R^{3x4}$$

*Ecuación 14: Matriz de proyección de cámara no calibrada.*

Cuando las cámaras **están calibradas**, esta matriz se encuentra factorizada en parámetros intrínsecos (matriz de calibración) y los extrínsecos (matriz de rotación y matriz de traslación, transforman las coordenadas mundo a coordenadas relativas a la cámara). Esta cuenta también dimensiones 3 x 4:

$$P_i = K_i \left[ R_i | t_i \right]$$

*Ecuación 15: Matriz factorizada de cámara calibrada*

Donde:

- $K_i$ es la **matriz de calibración** de la cámara que contiene los parámetros intrínsecos (conocida dependiendo del caso).

[20]Prácticamente la totalidad de este planteamiento está basado en *A Survey of Structure from Motion, Onur ¨Ozye¸sil, Vladislav Voroninski, Ronen Basri & Amit Singer* y en *A Taxonomy of Structure from Motion Methods, Federica Arrigoni.* Se recomienda leer para una información más detallada

$$K_i = \begin{bmatrix} f_x^{(i)} & \gamma^{(i)} & c_x^{(i)} \\ 0 & f_y^{(i)} & c_y^{(i)} \\ 0 & 0 & 1 \end{bmatrix}$$

*Ecuación 16: Matriz de calibración*

- $R_i$ es la **matriz de rotación** 3 x 3 de la cámara (desconocida)
- $t_i$ es la **matriz de traslación** 3 x 1 de la cámara (desconocida)

En el caso no calibrado, se habla de que la reconstrucción es **proyectiva**, mientras que en el caso calibrado la reconstrucción es **métrica**. En el planteamiento propuesto, se considera que se cuenta con parámetros EXIF en las imágenes, lo que puede dar lugar a dos situaciones:

1. La cámara es conocida[21], por lo que no es necesario estimar parámetros intrínsecos ya que se conocen.
2. No se conocen los parámetros intrínsecos finales pero se pueden estimar. Dado que este es el caso más común, es método que se desarrolla en este documento.

El problema se plantea como una resolución métrica en la que se define una matriz de calibración inicial $K_{0i}$ a partir de los parámetros EXIF:

$$f_x \simeq f_y \simeq f_{pixel} \rightarrow f_{pixel} = \frac{f_{mm}}{W_{(sensor\ mm)}} W_{(imagen\ px)}$$

$$c_x = \frac{W_{(imagen\ px)}}{2}; c_y = \frac{H_{(imagen\ px)}}{2}$$

$$\gamma = 0$$

*Ecuación 17: Criterios de matriz de calibración inicial*

Esta matriz se irá ajustando junto con los paramentos $R_i$ , $t_i$ y $M_j$ durante el ajuste por haces hasta llegar a un error de reproyección global mínimo. Para ello, se comienza definiendo la **matriz de proyección** del punto *j* (3D) en la imagen *i.* Donde Esta matriz tiene dimensiones (3 x 1):

$$m_{ij} \simeq P_i M_j$$

*Ecuación 18: Matriz de proyección del punto 3D i sobre imágenes*

El término de la izquierda ($m_{ij}$) representa los puntos de la imagen de entrada, por lo que se trata de las coordenadas de un punto 2D homogeneizadas, que resultan ser las coordenadas que se obtienen tras el emparejamiento de características.

$$m = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

*Ecuación 19: Matriz de puntos de la imagen de entrada homogeneizada*

El término de la derecha ($P_i M_j$) es desconocido y representa los nuevos puntos imagen obtenidos de reproyectar los puntos 3D estimados a través de las cámaras encontradas.

[21]En el caso de Meshroom, este caso se da si la cámara se encuentra registrada en la base de datos del programa, por lo que se tienen sus parámetros intrínsecos finales.

Es aquí cuando se plantea uno de los principales problemas en este proceso. Structure from Motion se basa en un problema de optimización no lineal con múltiples soluciones en el que se busca encontrar una solución en la que el error de proyección global sea mínimo, lo que se conoce como **ajuste por haces** (*bundle adjustment)*. Este ajuste debe iniciarse muy próximo a la solución final, ya que en muchos casos el resultado de este problema no es convexo, haciendo que la solución acabe convergiendo en un mínimo local que no pueda ser aplicado de manera global. Una forma de evitar esto es utilizando el concepto de geometría epipolar. La geometría epipolar consiste en establecer restricciones geométricas en base a la proyección de un mismo punto 3D en un par de imágenes, lo que provoca que las correspondencias entre proyecciones no sean arbitrarias, si no que obedezcan una relación geométrica definida. En este caso, esa relación geométrica viene definida por los parámetros de la cámara (intrínsecos y extrínsecos). Esto proporciona un inicio próximo a la solución final.

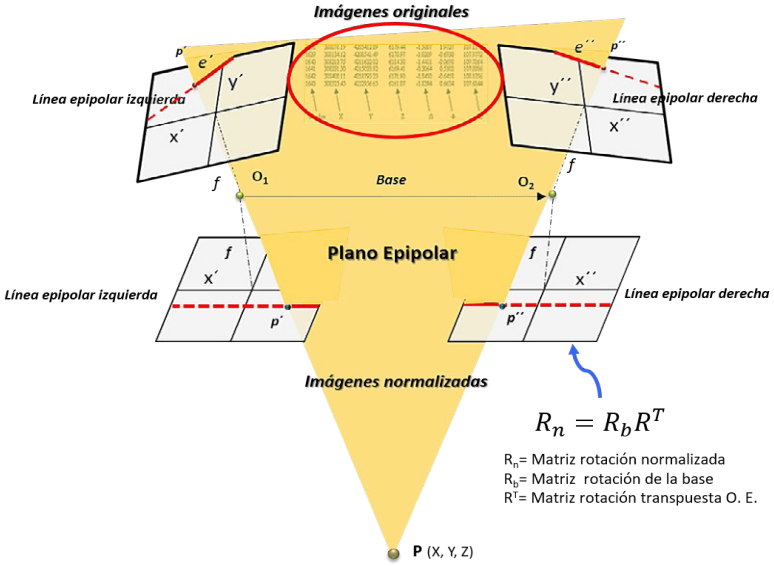


Figura 17: Representación de la geometría epipolar

Para ello, se comienza planteando restricciones epipolares a un par de imágenes $i_1 i_2$. Este par de imágenes será la base de toda la reconstrucción SfM, por lo que debe encontrarse altamente correlacionado. Se usa el que mayor cantidad de descriptores en común se haya calculado en el paso anterior. Dado que se cuenta con una matriz de calibración inicial, se pueden aplicar restricciones epipolares mediante la obtención de la matriz esencial, sin necesidad de calcular la matriz fundamental.

$$\widetilde{m}_{ij} \simeq {K_{0i}}^{-1} m_{ij}$$
$$\widetilde{m}_{i_1 j}^T E_{i_1 i_2} \widetilde{m}_{i_2 j} = 0$$
$$E_{i_1 i_2} = \left[ R_{i_1}^T \left( t_{i_2} - t_{i_1} \right) \right]_x R_{i_1}^T R_{i_2}$$

*Ecuación 20: Obtención de matriz esencial E de las cámaras $i_1$, $i_2$*

Donde:

- $\widetilde{m}_{ij}$ es la matriz de proyección **normalizada** respecto de $K_{0i}$
- $R_{i_1}^T R_{i_2} = R_{i_1 i_2}$ es una **matriz de rotación** relativa al par de cámaras $i_1 i_2$
- $\left[ R_{i_1}^T \left( t_{i_2} - t_{i_1} \right) \right] = t_{i_1 i_2}$ es **matriz de traslación** relativa al par de cámaras $i_1 i_2$

Esta rotación y esta traslación se utilizan para establecer el sistema de coordenadas sobre el que reconstruir el objeto 3D. Despejando estas ecuaciones se obtienen las matrices de cámara $P_{i_1}, P_{i_2}$. Con esta, se pueden obtener los puntos escena $M_j$ visibles desde de las imágenes $i_1 i_2$.

Este par de de imágenes queda completamente definido, por lo que es momento de aplicar **ajuste por haces**. Esta técnica consiste en minimizar el error de proyección que se da entre todos sus términos. En este caso, para las imágenes $i_1 i_2$. Para ello, se considera la "distancia" entre el término de la izquierda y el término de la derecha como el **error de reproyección**.

$$\underset{\left(K_i, R_i, t_i, M_j\right)}{min} \sum_{i=1}^{n} \sum_{j=1}^{p} \left\| m_{ij} - \pi\left(P_i M_j\right) \right\|^2$$

*Ecuación 21: Función de coste del error de reproyección total*

Donde $\pi$ es la proyección de las coordenadas homogéneas a la imagen.

Con el primer par de imágenes ya ajustado, se van añadiendo imágenes sucesivamente que se someten a un ajuste por haces local donde solo se ajustan los nuevos puntos y parámetros. Cuando el error acumulado comienza a ser elevado, se realiza un ajuste global en que se vuelven estiman todos los parámetros de nuevo. Este proceso se repite sucesivamente hasta que la reconstrucción queda completa.

Este enfoque de Structure from Motion es conocido como incremental o secuencial. Permite la reconstrucción en 3D de un objeto a partir de un par de imágenes fuertemente correlacionado sobre el que se añaden imágenes sucesivamente. Este método minimiza la probabilidad de converger en una solución local, ya que se tiene un control y un refinamiento constante de los resultados. Este es el enfoque utilizado por el programa Meshroom.

El resultado final de Structure from Motion es una nube de puntos conocida como dispersa[22] y los parámetros intrínsecos y extrínsecos de cada cámara involucrada en el ajuste. Esto da pie a la creación de los mapas de profundidad.

## 1.6 MAPAS DE PROFUNDIDAD, MULTI-VIEW STEREO (MVS)

En la amplia mayoría de los casos, SfM no es suficiente para reconstruir un objeto 3D. El resultado de SfM es una nube de puntos dispersa con la posición 3D de los distintos puntos característicos coincidentes entre las distintas imágenes que cumplen con la condición de tener un menor error de reproyección global. Esto quiere decir que en muchos casos, esta nube de puntos no tiene la densidad suficiente como para reconstruir el objeto con un nivel de detalle adecuado[23]. Para ello, se recurre a los mapas de profundidad (Depth maps).

Un mapa de profundidad es una reinterpretación de una imagen en la que cada valor pixel representa la distancia de la cámara que tomó la imagen original a la superficie visible en esa dirección[24].

[22]Se le llama dispersa porque no cubre la totalidad de la escena, solo los puntos característicos coincidentes tras el ajuste.

[23]Hay casos en los que combinando el uso de distintos algoritmos de detección de características se puede obtener una densidad de puntos decente para reconstruir un objeto en 3D, pero no es un caso ideal.

[24]La unidad en la que se representa el valor pixel del mapa de profundidad depende del programa. Meshroom usa metros.

Figura 18: Imagen original y mapa de profundidad, Puente de la Pedrera (INES, 2024)

Para la obtención de estos mapas es necesario recurrir a técnicas de Multi-View Stereo (MVS). Multi-View Stereo, al igual que Structure From Motion, buscan la reconstrucción de objetos 3D a partir de imágenes, con la diferencia de que en MVS las cámaras deben estar **calibradas.** Esto significa que deben conocerse sus parámetros **intrínsecos** y **extrínsecos**.

Al contrario que en Structure from Motion, en el campo de Multi-Stereo View existen gran cantidad de técnicas basadas en diferentes algoritmos y principios. Condensar todas estas en este apartado no sería posible, por lo que se describe la técnica utilizada por Meshroom.

Meshroom crea un mapa de profundidad para cada posición no descartada utilizando **Semi-Global Matching (SGM)**[25] . El principio detrás de esta técnica es similar al planteado por Structure from Motion. El objetivo consiste en calcular la disparidad a nivel de pixel entre pares de imágenes que se solapan, o dicho de otra forma, el desplazamiento horizontal entre un par de imágenes[26] que minimicen una función de energía global:

$$E(D)=\sum_{(x,y)}\left(C\left((x,y),D_{(x,y)}\right)+\sum_{(x\pm1,y\pm1)N(x,y)}PT\left[\left|D_{(x,y)}-D_{(x\pm1,y\pm1)}\right|>1\right]\right)$$

*Ecuación 22: función de energía a minimizar*

La disparidad resultante es convertida a profundidad relativa a la cámara:

$$Z_{c(x,y)}=\frac{f_i * B_{i_1 i_2}}{d_{i_1 i_2(x,y)}}$$

*Ecuación 23: Cálculo de la profundidad a partir de la disparidad*

Donde:

- $B_{i_1 i_2}$ es la distancia entre los centros proyección de las imágenes $i_1 i_2$ para calcular la disparidad (baseline).

$$C_i=-R_i^T t_i=\begin{bmatrix} X_0^{(i)} \\ Y_0^{(i)} \\ Z_0^{(i)} \end{bmatrix}$$

*Ecuación 24: Ecuación del centro de proyección en la cámara i*

[25]Se recomienda leer *Semi-Global Matching – Motivation, Developments and Applications, HEIKO HIRSCHMÜLLER, Oberpfaffenhofen*

[26]La disparidad se calcula mediante métodos como SAD, Census o NCC, Meshroom

- $d_{i_1 i_2(x,y)}$ es la disparidad calculada en el pixel (x, y) comparando las imágenes $i_1 i_2$

Esto devuelve mapas de profundidad relativos a cada cámara, que se utilizan para reconstruir el objeto final.

## 1.7 MESHING

Con los mapas de profundidad ya creados, que devuelven la profundidad del objeto a nivel píxel, es el turno de reconstruir el objeto en 3D. De manera simplificada, el proceso consiste en transformar los mapas de profundidad en una malla poligonal (o mesh). Un mesh es una colección de vértices, aristas y caras que dan lugar a una superficie tridimensional. En función de su naturaleza, las caras pueden estar formadas por 3 vértices (triángulos), 4 vértices (quads) o 5 o más (n-gones).

El proceso de meshing comienza convirtiendo los mapas de profundidad obtenidos anteriormente en nubes de puntos. Esto se hace proyectando cada pixel de la imagen original corregida de distorsiones, de tal forma que:

$$X_c = \frac{(x - c_x) * Z}{f_x}; Y_c = \frac{(y - c_y) * Z}{f_y}; Z_c = d(x, y)$$

*Ecuación 25: Transformación a coordenadas 3D*

Esto da lugar a tantas nubes de puntos como mapas de profundidad, lo que tiene un problema, y es que cada fragmento de la nube de puntos densa se esta construyendo en base a un sistema de coordenadas dado en la posición de la cámara, lo que se define de ahora en adelante como $M_c(x, y)$.

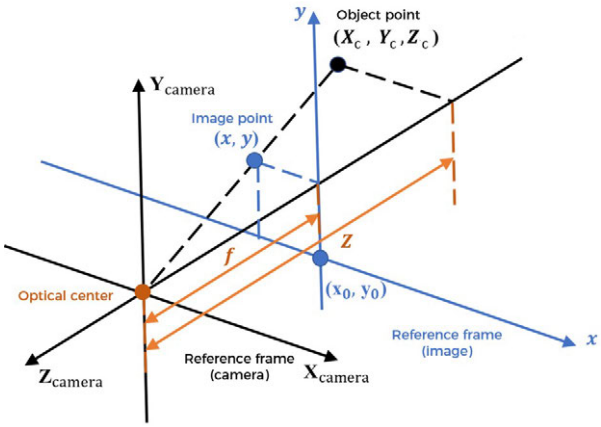


Figura 19: representación de proyección de un punto imagen a punto 3D[27]

Para unirlos en una única nube de puntos densa deben estar en un sistema de coordenadas común, definido como $M_w(x, y)$. La transformación se realiza mediante el uso de las rotaciones y traslaciones de cámara ($R_i, t_i$) calculadas en Structure from Motion:

$$M_w(x, y) = R_i^T (M_c(x, y) - t_i)$$

*Ecuación 26: Transformación de coordenadas cámara a coordenadas globales (World)*

[27]Modificada de: *Depth map to point cloud conversion sample (C++), mech-mind.com.* La imagen original solo contempla el uso de un único mapa de profundidad

Estando todos los fragmentos en un sistema de coordenadas común, ya se pueden combinar dando lugar a una nube de puntos densa[28] que puede ser convertida en un mesh. Para ello, existen multitud de planteamientos y estrategias. Como no se pueden recoger todas en este trabajo y la mayoría se encuentran bien documentadas, solo se profundiza en las distintas estrategias:

- **Métodos locales** que reconstruyen el objeto en fragmentos que luego unen, como *Ball Pivoting* o *Delaunay con filtrado*.
- **Métodos globales** que buscan optimizar una función que resuelva todo el conjunto de una sola vez, como la reconstrucción de Poisson.
- Métodos basados en **Machine Learning y Deep Learning.**

Todos estos métodos se encuentran correctamente documentados en sus respectivos artículos. Se decide no explicar ninguno ya que a día de hoy todos se encuentran en uso por distintos programas y todos ellos se basan en metodologías diferentes difíciles de generalizar.

A esta fase también suele estar asociado el texturizado del objeto, en que los colores RGB de la imagen se proyectan sobre el mesh desde los centros de proyección del objeto dando lugar a un mapa de UV.

[28]La mayoría de programas, como MetaShape y Meshroom, post-procesan la nube de puntos eliminando ruido, calibrando el sistema de coordenadas común y eliminando duplicados. Para el marco teórico, no se contempla.

# 2 PREPARACIÓN Y MATERIAL UTILIZADO

Con el marco teórico del proyecto ya definido se da comienzo a la preparación del proyecto. En esta fase se eligen los componentes de hardware y software a emplear y se planifica que zonas y objetos vas a ser escaneados.

## 2.1 HARDWARE UTILIZADO

El hardware consiste en todo el material físico utilizado durante el proyecto, ya sea para la captura de imágenes, el procesado de los datos o las pruebas de ejecución. La adecuada elección de este resulta fundamental, ya que afecta directamente a la calidad del trabajo y define una correcta optimización de los recursos. Esta selección debe llevarse a cabo cumpliendo los objetivos del proyecto y su filosofía.

**Cámaras**: Se utilizan para la captura de las imágenes que dan lugar a los objetos 3d. Se usan las siguientes cámaras:

| Canon EOS 60D + 50mm f/1.8 STM | |
|---|---|
| Tipo de cámara | Réflex digital (DSLR) |
| Sensor | CMOS APS-C (22,3 × 14,9 mm) |
| Resolución efectiva | 18 MP (5184 × 3456 px) |
| Procesador de imagen | DIGIC 4 |
| Formato de imagen | JPEG, RAW (CR2) |
| ISO | 100–6400 (extendible a 12800) |
| Óptica | Fija 50 mm (~80 mm / crop 1.6×), f/1.8 |
| Estabilización de imagen | No |
| Enfoque | 9 puntos tipo cruz, enfoque STM silencioso |
| Distorsión óptica | Muy baja (lente fija de 50 mm) |
| Visor | Óptico (pentaprisma) |
| Velocidad de obturación | 1/8000 s – 30 s |
| Disparo continuo | 5,3 fps |
| Flash incorporado | Sí |
| RAW | Si |


| Sony DSC-H400 | |
|---|---|
| Tipo de cámara | Bridge (lente fija) |
| Sensor | CCD 1/2.3" |
| Resolución efectiva | 20,1 MP (5152 × 3864 px) |
| Procesador de imagen | Desconocido |
| Formato de imagen | JPEG |
| ISO | 80–3200 |
| Óptica | Zoom 63× (24.5–1550 mm equiv.), f/3.4–6.5 |
| Estabilización de imagen | Sí, óptica (SteadyShot) |
| Enfoque | Detección de contraste, enfoque macro > 1 cm |
| Distorsión óptica | Alta en rangos antos de zoom |
| Visor | Digital |
| Velocidad de obturación | 1/2000 s – 30 s |
| Disparo continuo | No |
| Flash incorporado | Sí |
| RAW | No |


| Nothing Phone (2) | |
|---|---|
| Tipo de cámara | Smartphone |
| Sensor | Sony IMX890 (CMOS) + Samsung JN1 (CMOS) |
| Resolución efectiva | 50 MP (principal) + 50 MP (ultrawide) |
| Procesador de imagen | SnapDragon 8 gen 1+ |
| Formato de imagen | JPEG, RAW (DNG) |
| ISO | 50-3200 |
| Óptica | Lentes fijas |
| Estabilización de imagen | OIS y EIS (principal y ultrawide) |
| Enfoque | PDAF (principal y ultrawide) |
| Distorsión óptica | Baja (con corrección automática por IA) |
| Visor | Pantalla |
| Velocidad de obturación | 1/12000 – 32 s |
| Disparo continuo | No (por defecto) |
| Flash incorporado | Si |
| RAW | Si (modo experto) |

| Panasonic Lumix GF1 + 14 – 42mm f/3.5 – 5.6 | |
|---|---|
| Tipo de cámara | Réflex mirrorless |
| Sensor | CMOS (17,3 × 13 mm) |
| Resolución efectiva | 12,1 MP (4000 x 3000 px) |
| Procesador de imagen | Desconocido |
| Formato de imagen | JPEG, RAW (DNG) |
| ISO | 100-3200 |
| Óptica | Zoom 14 - 42mm, f/3.5 – 5.6 |
| Estabilización de imagen | Optica (lente) |
| Enfoque | Detección de contraste |
| Distorsión óptica | Alta en niveles de zoom altos |
| Visor | Sin visor |
| Velocidad de obturación | 1/4000 – 60 s |
| Disparo continuo | 3 fps |
| Flash incorporado | Si |
| RAW | Si |


| Pentax K20D | |
|---|---|
| Tipo de cámara | Réflex digital (DSLR) |
| Sensor | CMOS APS-C (23,4 × 15,6 mm) |
| Resolución efectiva | 14,6 MP (4672 x 3104 px) |
| Procesador de imagen | PRIME |
| Formato de imagen | JPEG, RAW (PEF, DNG) |
| ISO | 100-6400 |
| Óptica | Fija 50 mm, f/1.4 |
| Estabilización de imagen | Shake Reduction |
| Enfoque | SAFOX VIII, 9 puntos en cruz |
| Distorsión óptica | Muy baja (lente fija de 50 mm) |
| Visor | Optico pentaprisma |
| Velocidad de obturación | 1/4000 – 30 s |
| Disparo continuo | 3 fps |
| Flash incorporado | Si |
| RAW | Si |


| Dron DJI Mavik 3 Thermal (M3T) + RTK | |
|---|---|
| Tipo de cámara | Dron multimódulo |
| Sensor | CMOS (Telefoto y gran angular),<br>VOx (térmico) |
| Resolución efectiva | 12,1 MP (4000 x 3000 px) Telefoto,<br>48MP (8000 x 6000 px) Gran angular,<br>0,3MP (640 x 512 px) térmico |
| Procesador de imagen | Desconocido (DJI) |
| Formato de imagen | JPEG y JPEG/R-JPEG |
| ISO | 100 – 25600 |
| Óptica | Fija 162mm f/4.4 Telefoto,<br>Fija 24mm f/2.8 Gran angular,<br>Fija 40mm f/1.0 Térmico |
| Estabilización de imagen | Gimbal 3 ejes |
| Enfoque | Deteccion de contraste |
| Distorsión óptica | Muy baja |
| Visor | Remoto, Live View O3 |
| Velocidad de obturación | 1-8000 – 8 s |
| Disparo continuo | 1,4 fps |
| Flash incorporado | No |
| RAW | No |

*Tabla 2: Listado de cámaras utilizadas*

**Estación de trabajo y procesamiento de datos:** Corresponde con la estación encargada de llevar a cabo todas las operaciones de reconstrucción de objetos 3D, así como de su posterior edición e implementación en el entorno de videojuego. Para ello, se necesita que la máquina sea computacionalmente potente a nivel de procesamiento por CPU y GPU, capaz de almacenar todos los archivos resultado del proyecto y de poder mover el entorno digital en su fase de edición. Se usa la siguiente estación de trabajo:

| Lenovo Thinkstation P720 | |
|---|---|
| CPU | x2 Intel Xeon Glod 6161 2.2GHz (44 nucleos 88 hilos combinados) |
| RAM | 192 gb (12x16gb) DDR4 ECC |
| GPU | Nvidia Quadro P2000 5gb GDDR5<br>Nvidia RTX 5060 8gb |
| Almacenamiento | x2 NVME 3.0 1tb<br>x2 HDD 2tb |
| Sistema operativo | Fedora Workstation 42 |
| Notas | |
| El equipo cuenta con potencia suficiente para hacer las reconstrucciones de objetos 3D, aunque las tarjetas gráficas pueden ser un factor limitante en este caso, sobre todo a nivel de memoria. El apartado de generación del entorno virtual es muy demandante de GPU. | |

*Tabla 3: Estación de trabajo utilizada*

Con el hardware ya bien definido, es turno de pasar al **software**.

## 2.2 SOFTWARE UTILIZADO

El software es el encargado del procesamiento e interpretación de la información tomada en campo y la creación y edición de los objetos y el producto final. La elección del software viene condicionada por las necesidades del proyecto, donde se necesita poder generar objetos 3D a partir de fotogrametría, editarlos y construir un entorno coherente con ellos; así como por la filosofía establecida, donde se busca fomentar el uso del software libre.

**Meshroom:** Se trata de un programa que permite la reconstrucción de objetos 3D utilizando el framework AliceVision. Su funcionamiento se basa en el desarrollo de un pipeline mediante nodos, en los que cada nodo de estos realiza una tarea por bloques. Por defecto, Meshroom utiliza el siguiente pipeline para la reconstrucción de objetos 3D.

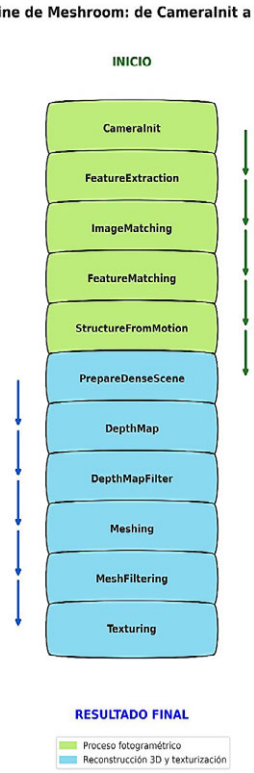


Figura 20: Pipeline por defecto de Meshroom

Este será la base de la reconstrucción 3D de todos los objetos. En la documentación oficial de Meshroom se describe detalladamente el funcionamiento de cada nodo y sus distintos parámetros de configuración. En este proyecto se comienza utilizando la versión 2023.3.0, que posteriormente es actualizada a la 2025.1.0.

**Scripts de Python**: Como se dicho anteriormente, Meshroom trabaja encadenando nodos que procesan la información por bloques. Esto permite que, si fuera necesario, el pipeline puede detenerse sin tener que repetir todo el proceso, ya que cada nodo guarda un resultado de manera independiente. Esto resulta muy útil, ya que existen casos en los que es necesario modificar el resultado de alguno de los nodos durante la ejecución del pipeline.

Aunque Meshroom no está estrictamente diseñado para trabajar fuera de su flujo de trabajo, muchos de los resultados generados en los distintos nodos se encuentran en formatos estandarizados, como JSON. Esto permite poder detener el pipeline, realizar modificaciones de los resultados obtenidos y reanudar el proceso.

Bajo esta premisa, se plantean varios scripts:

- extractXYZFromSFM.py: El nodo *StructureFromMotion* genera un archivo *sfm.abc* que contiene la nube de puntos dispersa generada mediante Bundle Adjustement y las cámaras desde donde es visible cada punto de la nube. Este archivo no es texto plano, pero utilizando el nodo *ConvertSFMFormat,* es posible transformarlo a JSON, lo que permite modificarlo libremente.

  Con este contexto, se plantea un script que extraiga del archivo JSON la nube de puntos dispersa y la almacene en formato .xyz ([x y z r g b], en texto plano). Esto permite poder limpiar la nube de puntos dispersa de posibles falsas coincidencias o ruido en programas como CloudCompare.

- BuildSFMFromXYZ.py: De forma análoga al script anterior, una vez que se ha filtrado la nube de puntos dispersa hay que reconstruir el JSON. Este debe estar correctamente estructurado, por lo que es necesario respetar el formato anterior. Es por eso que se desarrolla un script en el que se comparan los puntos del archivo JSON original y los de la nube de puntos filtrada. Los puntos que estén en ambos archivos se mantienen, eliminando del JSON los que no tengan coincidencia (junto con sus cámaras).

- ConvertEXRtoPNG: Meshroom por defecto exporta sus archivos en formato obj, asociando múltiples texturas a este como imágenes en formato exr mediante un archivo mlt. Unreal Engine no soporta texturas en formato exr, por lo que es necesario convertirlas a un formato reconocible, como PNG. A pesar de que el nodo Texturing permite exportar texturas en PNG de manera nativa, se opta por el uso de este script debido a que inicialmente no se contempló la falta de soporte de de archivos EXR en Unreal, por lo que se exportaron todos los objetos con las texturas en este formato, ya que ofrece una mejor calidad y rango dinámico. Este script transforma las texturas en PNG y actualiza el archivo MTL sin tener que volver a exportar los objetos, ahorrando tiempo.

**Blender:** Es un programa de gráficos en 3D multiplataforma que permite la creación, edición, renderizado, animación y texturización de objetos 3D. Su uso esta muy extendido, siendo muy utilizado de manera profesional en la industria del cine y los videojuegos para la creación de entornos y VFX. En este proyecto, se utiliza para la edición de geometrías mediante el modo edición y las herramientas de escultura.

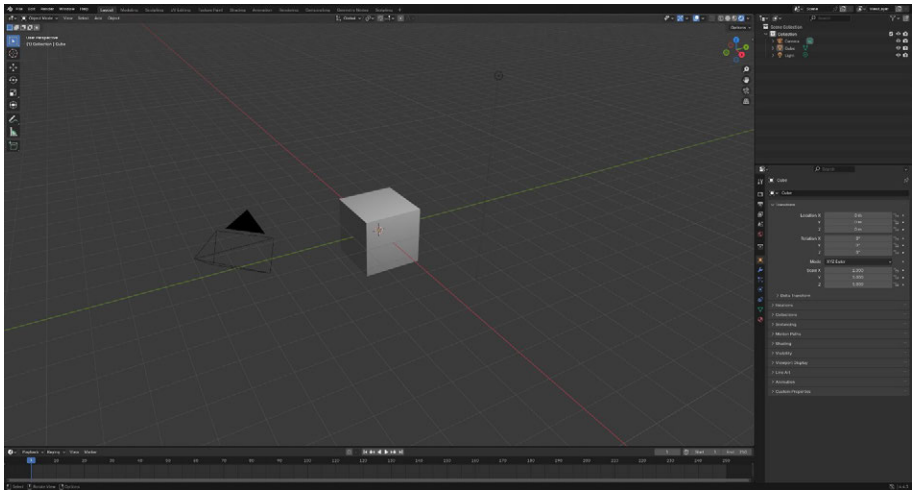

Figura 21: Interfaz de Blender

**CloudCompare**: Es un programa multiplataforma y de software libre que permite la edición de nubes de puntos y, de manera más limitada, de archivos mesh. En este proyecto se usa para el tratamiento de las nubes de puntos dispersas, utilizando las herramientas de filtrado de ruido automático y la herramienta de recorte.

## 2.3 PREPARACIÓN

Con todo el hardware y el software seleccionado, se comienza a buscar zonas para digitalizar en 3D que puedan dar forma a un espacio coherente en conjunto. Se plantean un total de 17 ubicaciones repartidas por España, concentradas en Galicia, Asturias, Zaragoza y Madrid. Cada una cuenta con unas características distintas para testear los límites del proyecto:

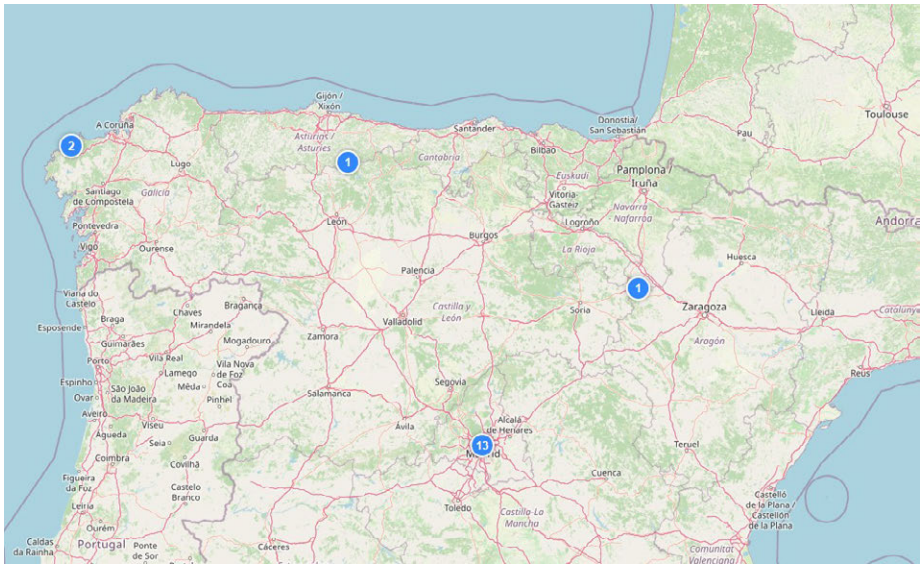

Figura 22: Distribución de localizaciones , generado con Folium

En Galicia, se toman 2 objetos:

1. Banco Ponteceso: Es un banco de piedra situado en la localidad de Ponteceso (Puenteceso), con vistas a la ría. Se propone con la idea de ser usado como *prop*[29].
2. Playa de los cristales. Ubicada en Laxe en *A Costa da Morte*. Es una pequeña cala rodeada de roca, con la particularidad de que la arena está formada por cristales verdes a consecuencia de vertidos de botellas al mar. Con el paso de los años, estos se han ido fragmentando y limando dando lugar a pequeñas piedras que le dan ese color característico. LA idea es utilizar este espacio como escenario completo

[29]Se define como prop


Figura 23: Ubicación banco Ponteceso y Playa de los cristales

En Asturias se usa el puerto de San Isidro. Este puerto une Asturias con la provincia de León atravesando la cordillera Cantábrica. Su captura fue realizada mediante un vuelo de dron por la empresa INES Ingenieros Consultores en el marco de un proyecto de instalación de viseras para la prevención de aludes. Se propone con la idea de ser usado como elemento de relleno en los limites del mapa.

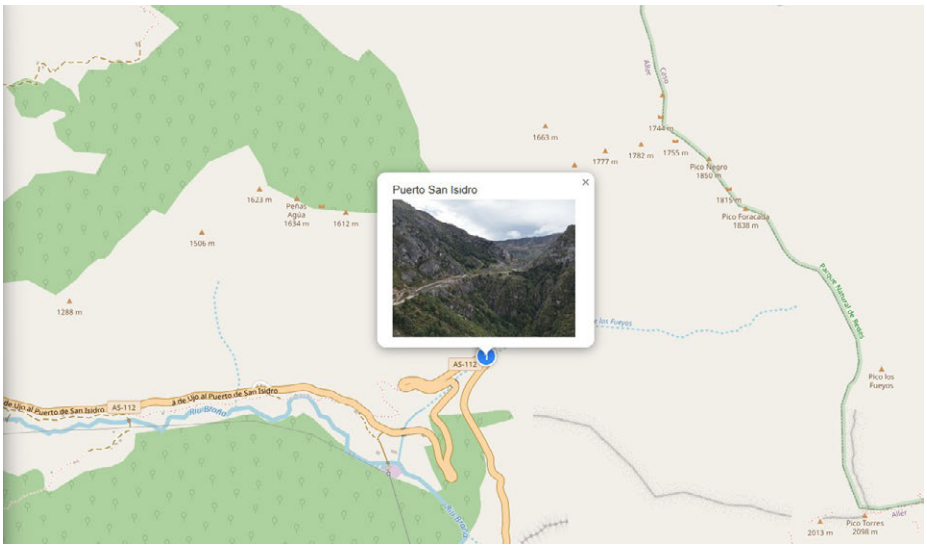


Figura 24: Ubicación del puerto de San Isidro

En la provincia de Zaragoza se utilizan las casas colgadas de Tarazona. Estas casas se encuentran localizadas en el barrio de la judería y resultan un atractivo turístico. Su captura fue realizada por un vuelo de dron, también por INES Ingenieros Consultores en el marco de las obras de emergencia para la estabilización del terreno a causa de la DANA de julio de 2025.

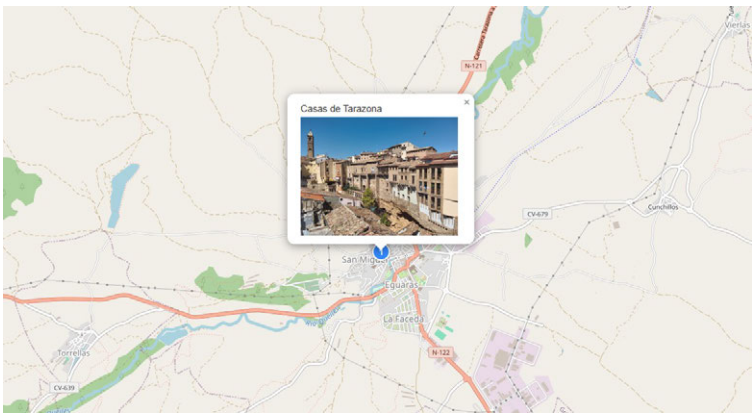


Figura 25: Ubicación casas de Tarazona

El resto de ubicaciones se encuentran localizadas en Madrid. Se capturan 2 escenas en Alcorcón, una escena en Madrid Rio, 2 elementos en Parla, 1 en Aldea del Fresno y 7 elementos en el jardín del capricho:

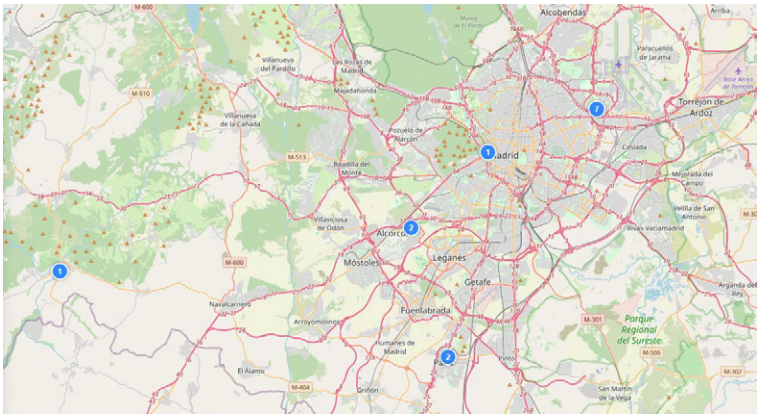


Figura 26: Distribución de ubicaciones en Madrid

En Alcorcón, se capturan dos esculturas decorativas de piedra localizadas en el interior del Parque de los castillos. Dado que no tienen un nombre oficial, se usa “Figura Alcorcón 1” y “Figura Alcorcón 2” para referirse a ellas.

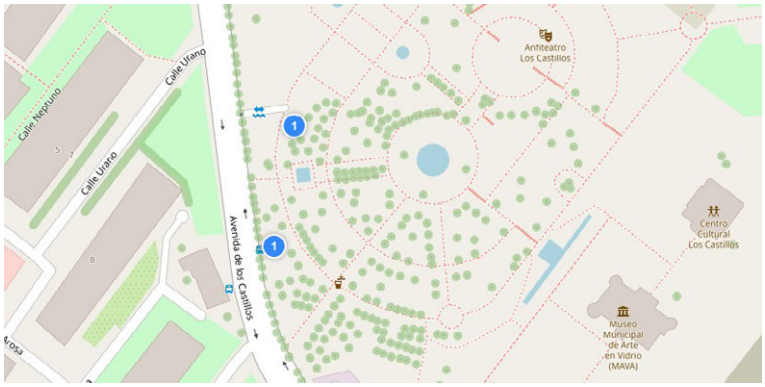


Figura 27: Ubicación de figuras de Alcorcón

En Madrid Rio, se capturan unos bancos de piedra situados entre el río Manzanares y la Puerta de San Vicente.

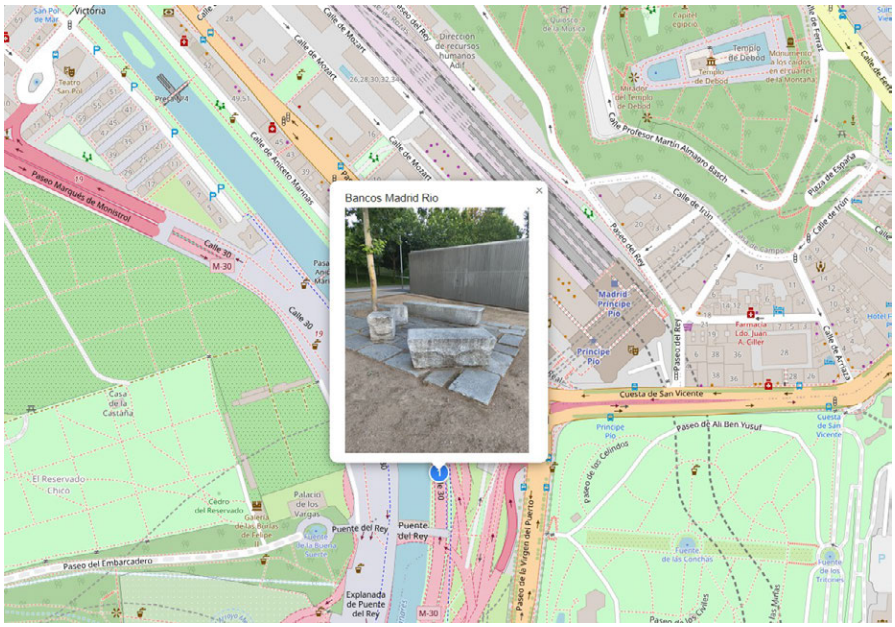


Figura 28: Ubicación bancos Madrid Rio

En Parla, se captura una fuente en un parque al lado de la calle Real, conocida como “Un pueblo en lucha, Homenaje a los Parleños y a Nicasio Ursinio”. El objetivo es utilizar este como prop dentro de un mapa y a su vez, observar los efectos de la reconstrucción en presencia de agua en movimiento. El segundo elemento es una de las 3 esculturas de un León que encuentran en la calle Valladolid. Esta en concreto, es la más cercana a la calle Cuenca. Se plantea utilizar esta como prop.

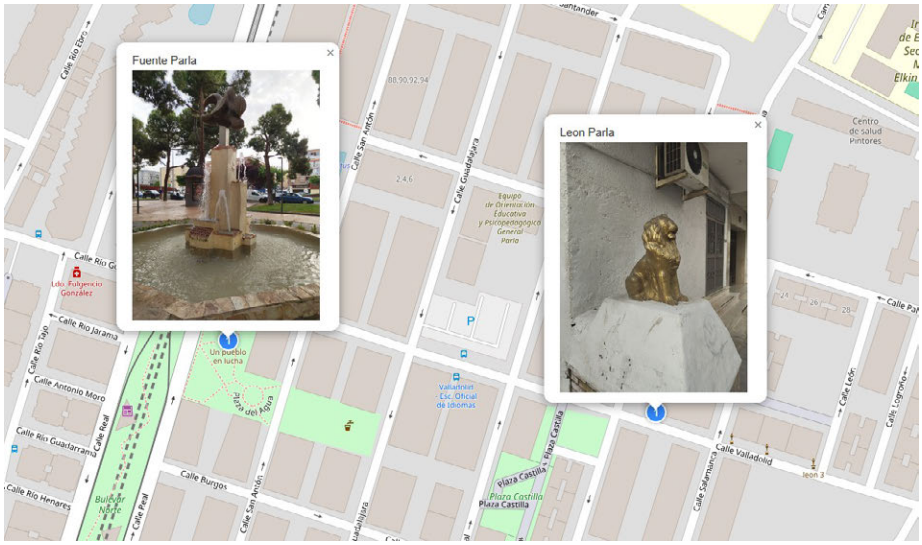


Figura 29: Ubicación León Parla y Fuente Parla

En Aldea del fresno, se captura el Puente de la Pedrera. Este puente histórico fue construido en el siglo XVIII y ha sufrido varias transformaciones a lo largo de los años. En septiembre de 2023 el puente quedó parcialmente destruido por una DANA que provocó una crecida del rio Alberche. Su captura fue también realizada por INES, en el proyecto de musealización del mismo.

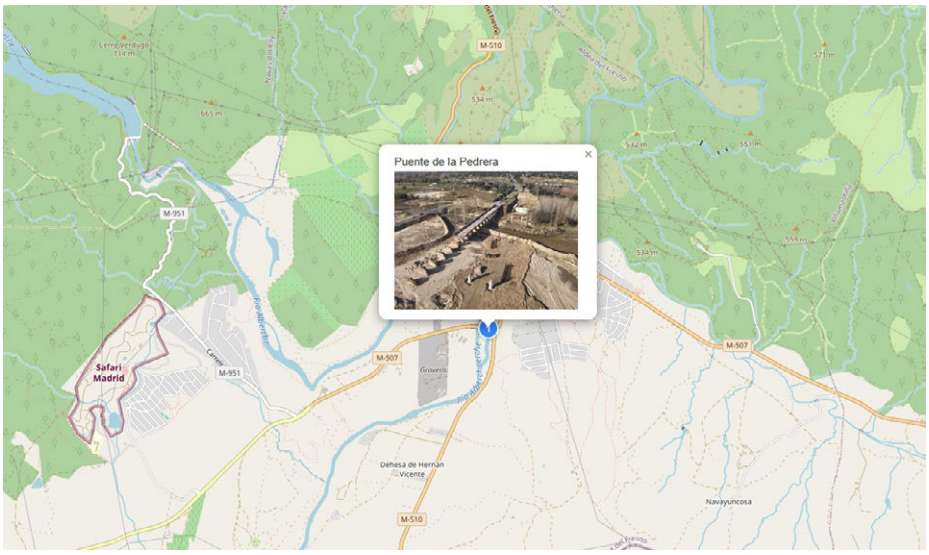


Figura 30: Ubicación Puente de la Pedrera

Por último, en el Jardín del capricho. Estas escenas tienen la particularidad de que se capturaron en diferentes días con diferentes cámaras. En el apartado de ejecución del proyecto se :

1. Columna entrada: Es una de las conocidas como Columnas de los Enfrentados (la más alejada de la entrada). Estas fueron realizadas en 1840 y se encuentran separadas por 40 pasos, simbolizando un duelo. Como se verá más adelante, su captura se ve condicionada por el entorno. Es utilizado como prop.
2. Columna lago: Columna baja con un busto en la parte superior situada al lado del estanque de los patos. Se encuentra rodeado árboles, lo que condiciona la iluminación del entorno. Es utilizado como prop.
3. Tocón arbusto: Se trata del tocón de un árbol cubierto de vegetación cercano al lago de El Capricho. La idea es utilizar este como prop y observar como se comporta la reconstrucción ante la presencia de elementos que puedan estar en movimiento, como las hojas del arbusto.
4. Templo: Se trata de un templete de finales del siglo XVIII dedicado al dios Baco, de quien hay una escultura en su interior (esta escultura es posterior al templo, de principios del siglo XIX). Este se encuentra en lo alto de una pequeña colina. Se plantea utilizar este como prop y estudiar los efectos de la captura por fotogrametría terrestre en escenas grandes.
5. Templo esfinges: Conocido formalmente como Exedra. Data de mediados del siglo XVIII y fue completamente restaurado a finales del siglo XX. Se caracteriza por tener múltiples esfinges colocadas en forma semicircular salvaguardando un busto homenaje a Dª María Josefa, duquesa de Osuna. Se plantea su uso como prop y para el estudio de los efectos de la captura de superficies reflectantes (las esfinges producen reflejos y no pueden ser digitalizadas por detrás).
6. Busto 1. Se trata de uno de los 12 bustos situados en la Plaza de los Emperadores. Cada uno de los bustos, representa a uno de los emperadores del imperio Romano. Se utiliza como prop.
7. Busto 2. Se trata de otro de los 12 bustos situados en la Plaza de los Emperadores. Este en concreto, se encuentra situado a lado del la Exedra. Se utiliza como prop.


Figura 31: Ubicaciones jardín de El Capricho

# 3 PROCESO DE RECONSTRUCCIÓN

Una vez que se cuenta con el material y que se conocen las zonas, se comienza con la reconstrucción de los objetos. En este apartado, se documenta el proceso que se ha seguido para cada objeto desde la captura de imágenes hasta su construcción final. Para todas las escenas se toma como referencia un mismo flujo de trabajo con el fin de maximizar la calidad del resultado.

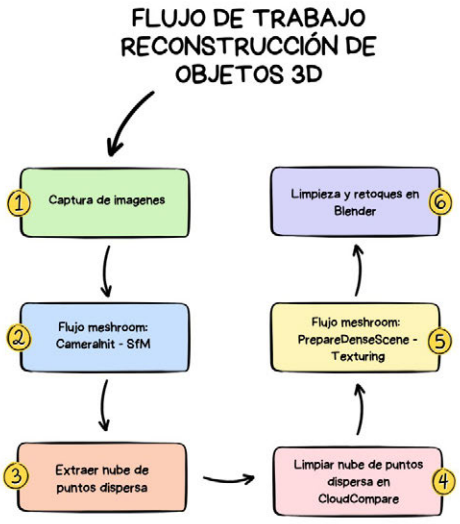


Figura 32: Flujo de trabajo general

En Meshroom, se suele partir de la configuración por defecto que ofrecen los nodos para una reconstrucción fotogramétrica. Para cada escena, solo se mencionan los cambios realizados respecto a este.

## 3.1 BANCO PONTECESO

La captura de imágenes se realizó entre las 21:43 y las 21:45 del día 15 de julio de 2025 utilizando la cámara del Nothing Phone 2 en modo experto[30]. Se capturaron un total de 176 imágenes en formato JPG. En un primer filtrado visual, no se descarta ninguna de las imágenes.

Figura 33: Captura de imágenes Banco Ponteceso

Con las imágenes tomadas y con un primer filtrado previo, se comienza con la primera parte del flujo en Meshroom. En el nodo FeatureExtraction se elige el descriptor DSPSift, se establece la densidad en Ultra y la calidad en High. La ejecución de esta primera parte da una nube de puntos dispersa de 1.562.103 puntos y las distintas posiciones de captura. Durante la fase de FeatureMatching no se descarta ninguna de las imágenes.

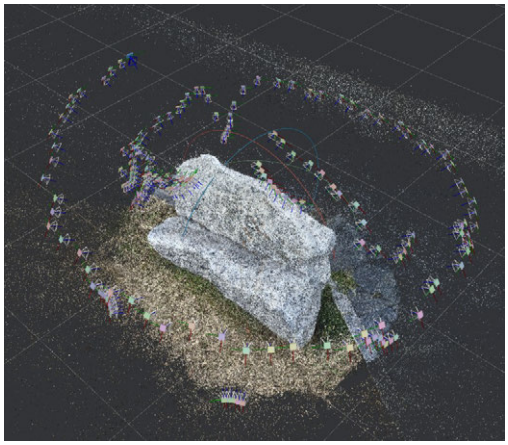

Figura 34: Nube de puntos dispersa y sus poses, Banco Ponteceso

Una vez finalizada la primera parte del flujo, se extrae la nube de puntos dispersa y se limpia en CloudCompare:

[30]En el Nothing Phone 2, activar el modo experto minimiza las correcciones por IA.

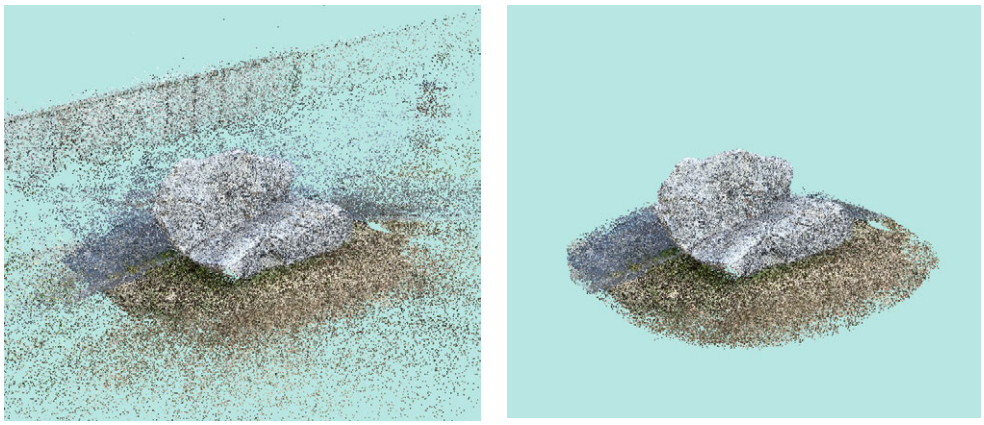

Figura 35: Limpieza nube de puntos dispersa Banco Ponteceso, CloudCompare

Con la nube de puntos dispersa filtrada, se procede con la segunda parte del proceso. En los nodos de DepthMap y Texturing se desactiva el Downscale y se añade el nodo MeshDecimate para decimar el modelo. El resultado de esta parte del flujo es el objeto ya texturizado.

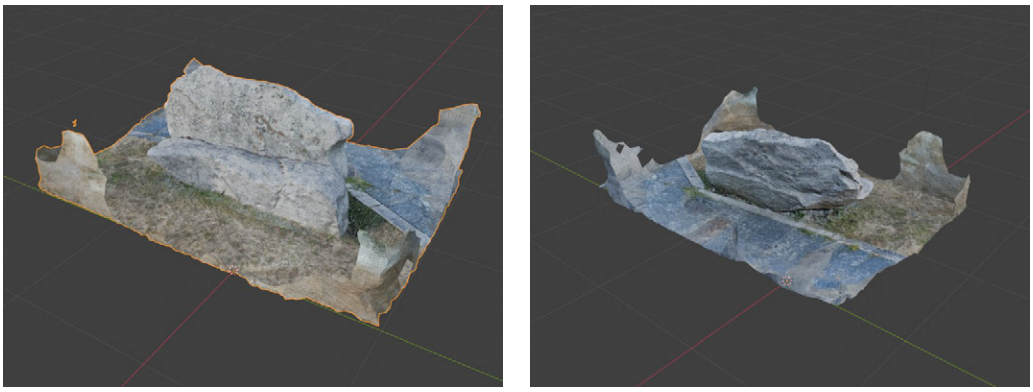

Figura 36: Mesh texturizado Banco Ponteceso

Con el objeto ya texturizado, se importa en Blender para recortar la zona de interés y aplicarle un suavizado, dando lugar al objeto final.

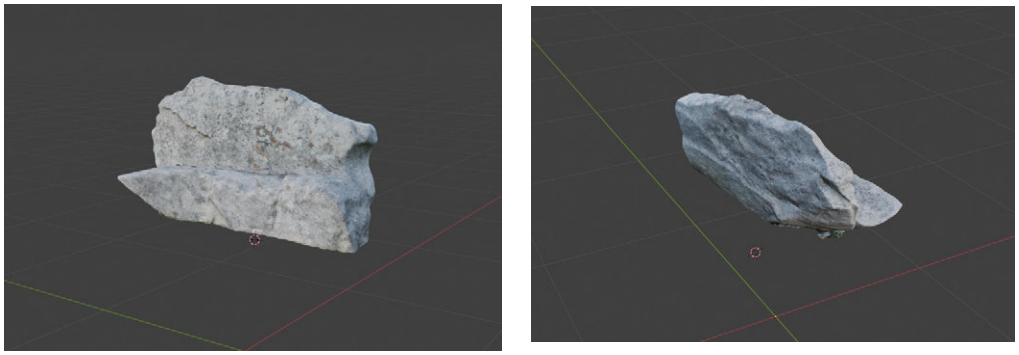

Figura 37: Mesh texturizado Banco Ponteceso

## 3.2 PLAYA DE LOS CRISTALES

La captura de imágenes se realizó entre las 20:10 y las 20:43 del día 17 de julio de 2025 utilizando la cámara del Nothing Phone 2 en modo experto. Se toman un total de 901 imágenes en formato JPG. Durante la captura aparecen varias personas. A priori, no se

descarta ninguna imagen ya que las personas se encuentran en constante movimiento, por lo que no debería haber coincidencias entre imágenes.

Figura 38: Captura de imágenes Playa de los Cristales

Con las imágenes tomadas, se comienza con el flujo en Meshroom. A modo de prueba se ejecuta el flujo completo de Meshroom con los siguientes cambios:

- En el nodo FeatureExtraction se utiliza el descriptor sift_float. Se establece la densidad y la calidad en Normal.
- En el nodo DepthMap y Texturing se desactiva el downscale.

El resultado es una de puntos dispersa con 1.177.030 puntos junto con las posiciones de las cámaras y el objeto ya texturizado. Durante este proceso, se descartan 63 imágenes, de las cuales casi todas eran planos muy cercanos de objetos poco visibles.

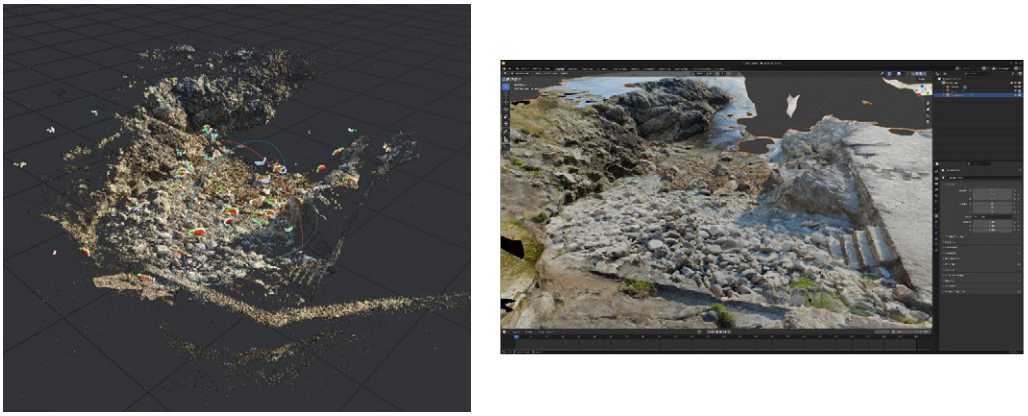

Figura 39: Nube de puntos dispersa y mesh texturizado Playa de los Cristales

El resultado se considera aceptable, por lo que no se extrae la nube de puntos dispersa para su limpieza. Se procede directamente al decimado y texturizado. Al decimar el objeto, se observa que las texturas quedan desplazadas, por lo que se procede a volver a ejecutar el nodo de texturing de Blender con el nuevo mesh.

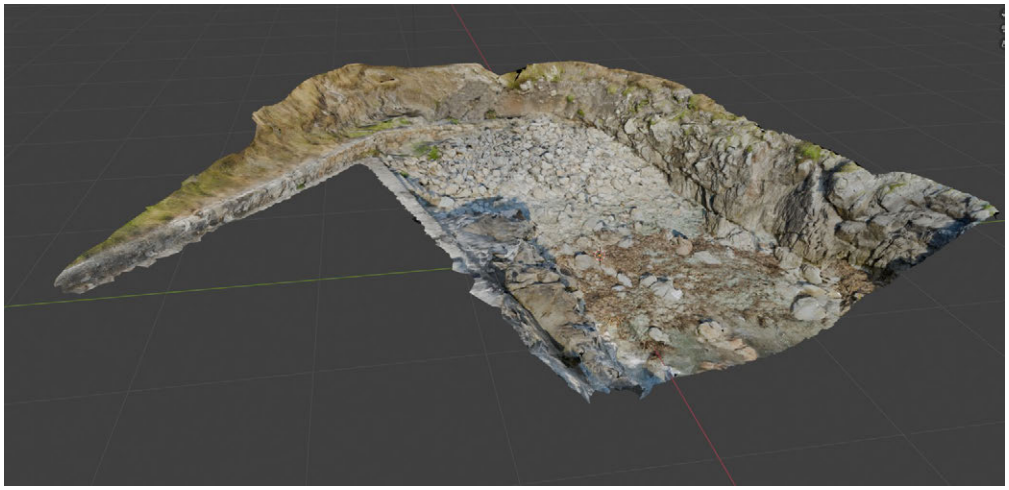

Figura 40: Mesh filtrado Playa de los Cristales

## 3.3 PUERTO DE SAN ISIDRO

La captura de imágenes se realizó en dos tramos, siendo el primero entre las 12:56 y las 13:37 y el segundo entre las 14:13 y las 14:41 del día 4 de octubre de 2024 utilizando el dron DJI a una gran altura del suelo. Se capturan un total de 369 imágenes en formato JPG. Durante la captura aparecen varias personas. A priori, no se descarta ninguna imagen ya que las personas se encuentran en constante movimiento, por lo que no debería haber coincidencias entre imágenes.

Figura 41: Captura de imágenes Puerto de San Isidro

Con las imágenes tomadas, se comienza con la primera parte del flujo en Meshroom. En este caso, debido al tamaño y la naturaleza de la escena, se decide realizar el flujo de Meshroom completo, sin exportar la nube de puntos dispersa para limpiar. Esto se debe a que la distancia de captura de las imágenes se hace lo suficientemente lejos como para que el filtrado a nivel de detalle no sea necesario. El resultado da una nube de puntos

dispersa con 1.021.389 puntos y sus poses, junto con el mesh texturizado. Durante este proceso, se descartan 24 imágenes por falta de coincidencias.

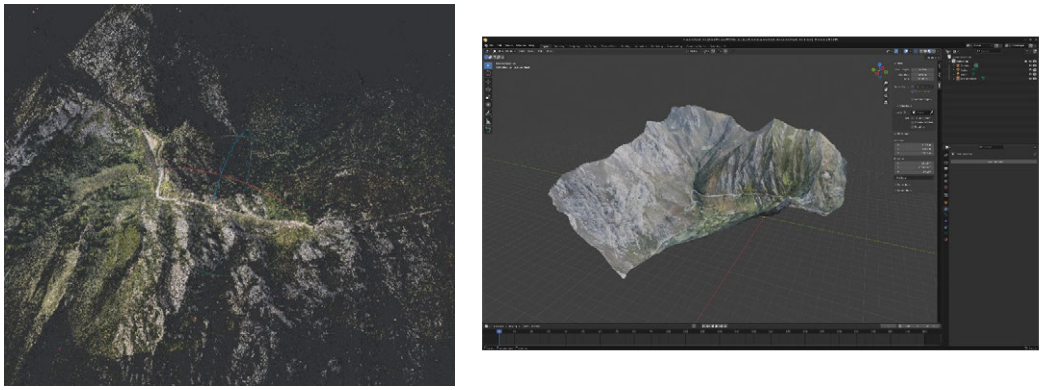

Figura 42: Nube de puntos dispersa y mesh texturizado Puerto de San Isidro

Con el flujo de Meshroom terminado, se exporta el objeto a Blender, donde se hace un decimado del modelo y un suavizado de la geometría. Debido a que este modelo se capturó usando también un módulo RTK, el modelo aparece escalado.

Figura 43: Nube de puntos dispersa y mesh texturizado Puerto de San Isidro

### 3.4 CASAS DE TARAZONA

La captura de imágenes se realizó entre las 12:56 y las 17:33 del día 4 de agosto de 2025 utilizando el dron DJI. Se capturan un total de 342 imágenes en formato JPG. En un primer filtrado visual no se descarta ninguna imagen.

Figura 44: Captura de imágenes Casas de Tarazona

Con las imágenes tomadas, se comienza con el flujo en Meshroom. En una primera aproximación, se ejecuta el flujo completo de Meshroom:

- En el nodo FeatureExtraction se utilizan los descriptores sift, sift_float, sift_upright y DSPSift. Se establece la densidad en Ultra y la calidad en Normal.
- En el nodo DepthMap y en el de Texturing se desactiva el downscale.

El resultado es una nube de puntos dispersa con 4.390.265 puntos con las posiciones de captura y el archivo mesh ya texturizado. Al haber usado un dron equipado con un RTK, el modelo se escala automáticamente a su tamaño real.

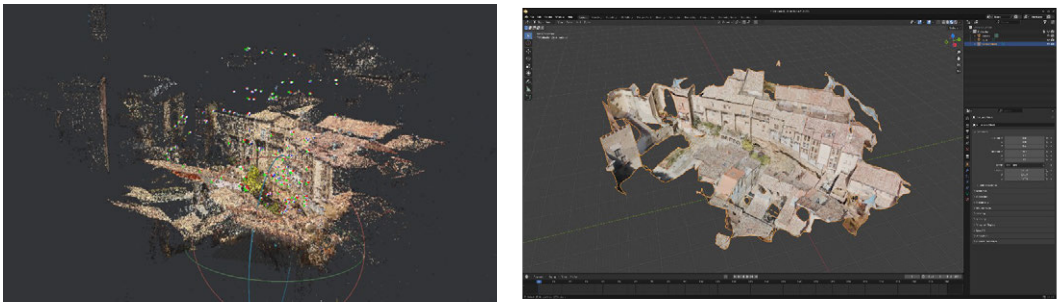

Figura 45: Nube de puntos dispersa y mesh texturizado Casas de Tarazona

El resultado es bastante aceptable, por lo que únicamente se recorta, se decima y se suaviza para su uso.

## 3.5 FIGURA ALCORCÓN 1

La captura de imágenes se realizó entre las 20:18 y las 20:26 del día 14 de junio de 2025 utilizando la Canon EOS. Se toman un total de 235 imágenes en formato RAW (CR2). No se descarta ninguna imagen.

Figura 46: Captura de imágenes Figura Alcorcón 1

Con las imágenes tomadas, se comienza con la primera parte del flujo en Meshroom. En el nodo FeatureExtraction se elige el descriptor sift_float, se establece la densidad en Ultra y la calidad en High. La ejecución de esta primera parte da una nube de puntos dispersa de 391.897 puntos y las distintas posiciones de cámara. Durante esta fase, no se descarta ninguna imagen.

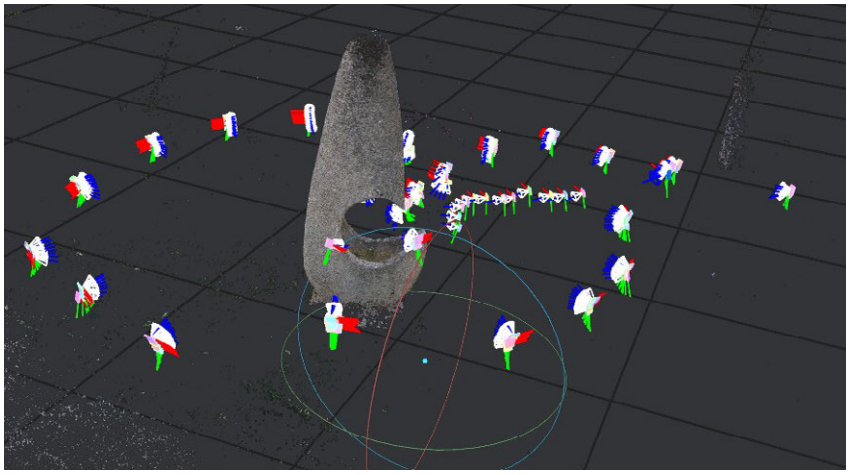

Figura 47: Nube de puntos dispersa y posiciones Figura Alcorcón 1

Una vez finalizada la primera parte del flujo, se extrae la nube de puntos dispersa y se limpia en CloudCompare:

Figura 48: Limpieza nube de puntos dispersa Figura Alcorcón 1, CloudCompare

Con la nube de puntos dispersa filtrada, se procede con la segunda parte del proceso. En los nodos de DepthMap y Texturing se desactiva el Downscale. El resultado es el objeto ya texturizado.

Figura 49: Objeto texturizado Figura Alcorcón 1, CloudCompare

Con el objeto ya texturizado, se importa en Blender para recortar el ruido de la parte superior y aplicarle un suavizado y un decimado, dando lugar al objeto final.

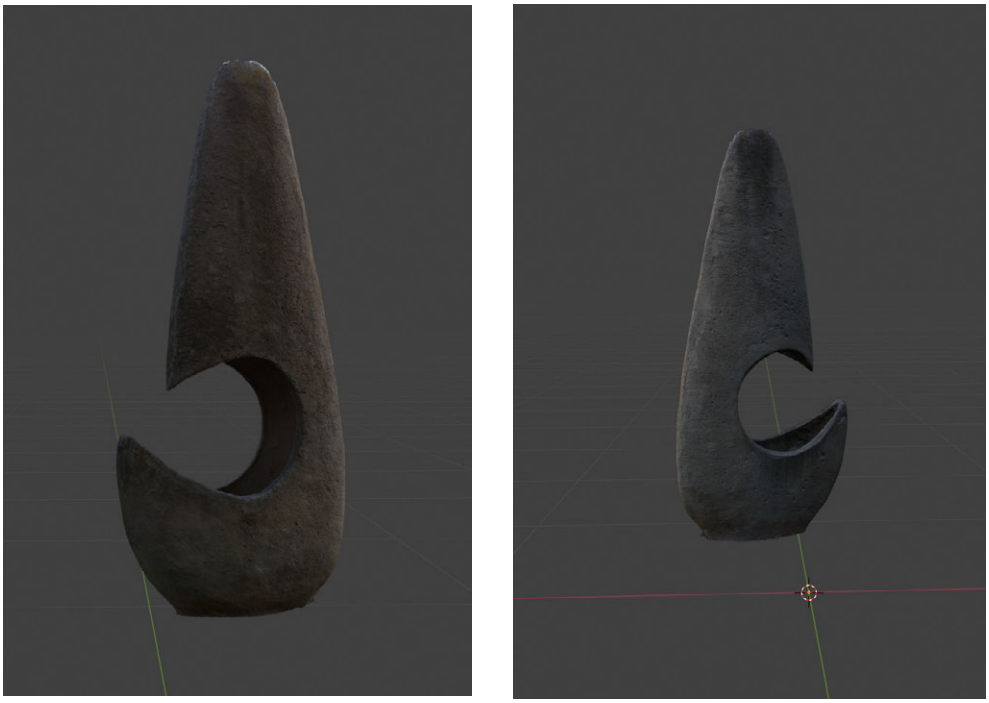

Figura 50: Objeto filtrado Figura Alcorcón 1, CloudCompare

## 3.6 FIGURA ALCORCÓN 2

La captura de imágenes se realizó entre las 20:28 y las 20:33 del día 14 de junio de 2025 utilizando la Canon EOS. Se capturan un total de 201 imágenes en formato RAW (CR2). No se descarta ninguna imagen.

Figura 51: Captura de imágenes Figura Alcorcón 2

Con las imágenes tomadas, se comienza con la primera parte del flujo en Meshroom. En el nodo FeatureExtraction se elige el descriptor sift_float, se establece la densidad en High y la calidad en Ultra. La ejecución de esta primera parte da una nube de puntos dispersa de 148.452 puntos y las distintas posiciones de cámara. Durante esta fase, no se descarta ninguna imagen.

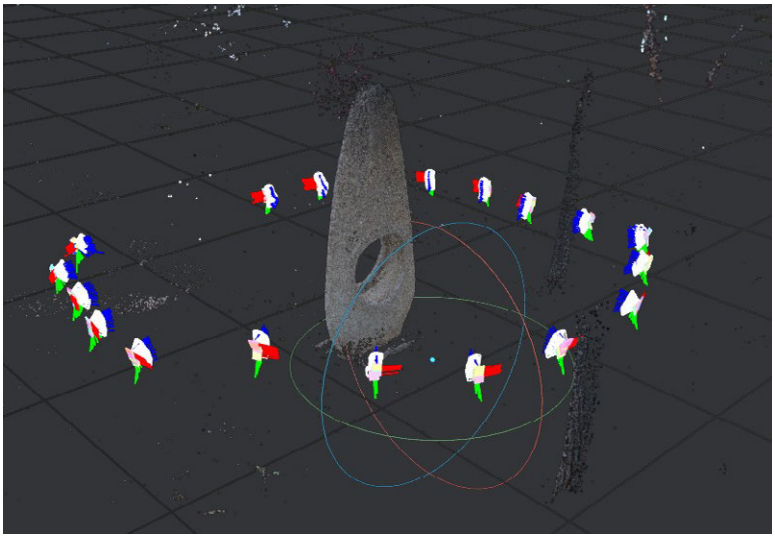

Figura 52: Nube de puntos dispersa y posiciones Figura Alcorcón 2

Una vez finalizada la primera parte del flujo, se extrae la nube de puntos dispersa y se limpia en CloudCompare:

Figura 53: Limpieza nube de puntos dispersa Figura Alcorcón 2, CloudCompare

Con la nube de puntos dispersa filtrada, se procede con la segunda parte del proceso. En los nodos de DepthMap y Texturing se desactiva el Downscale. El resultado es el objeto ya texturizado.

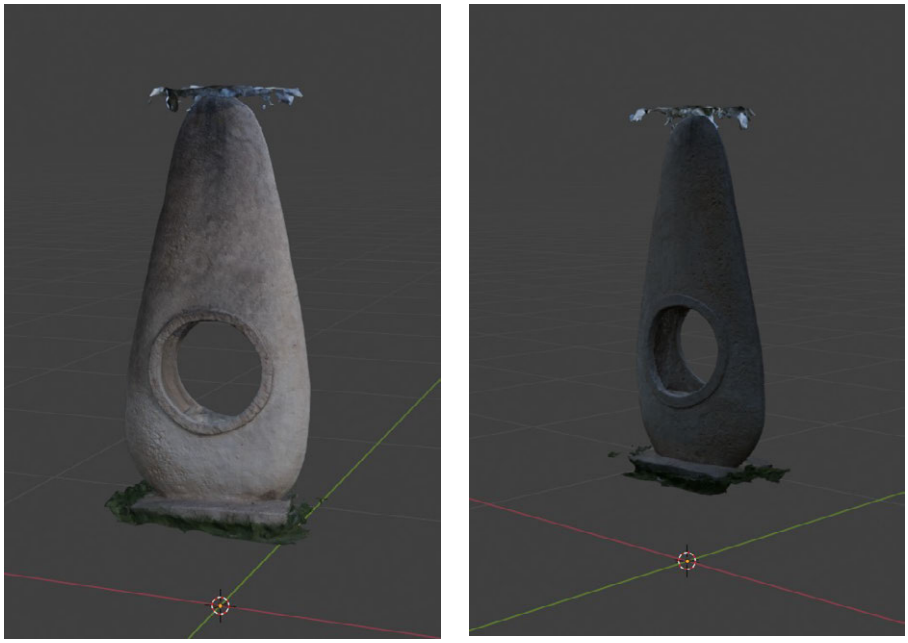

Figura 54: Objeto texturizado Figura Alcorcón 2, CloudCompare

Con el objeto ya texturizado, se importa en Blender para recortar el ruido de la parte superior y aplicarle un suavizado y un decimado, dando lugar al objeto final.

Figura 55: Objeto filtrado Figura Alcorcón 2, CloudCompare

## 3.7 BANCOS MADRID RIO

La captura de imágenes se realizó entre las 21:26 y las 21:34 del 5 de junio de 2025 utilizando la cámara del Nothing Phone 2 en modo experto. Se toman un total de 327 imágenes en formato JPG. Haciendo un primer filtrado visual, no se descarta ninguna imagen.

Figura 56: Captura de imágenes Bancos Madrid Rio

Con las imágenes tomadas, se comienza con el flujo en Meshroom. Se considera que en base a las condiciones de captura el modelo puede realizarse en un flujo único:

- En el nodo FeatureExtraction se utiliza el descriptor DSPSift. Se establece la densidad y la calidad en Normal.
- En el nodo DepthMap se desactiva el downscale.
- Se añade el nodo MeshDecimate para decimar sin tener que pasar por Blender.

El resultado de este ofrece una nube de puntos dispersa de 455.836 puntos junto con sus poses y el mesh ya texturizado. Durante el proceso no se descarta ninguna imagen.

Figura 57: Nube de puntos dispersa y mesh texturizado Bancos Madrid Rio

Con el mesh ya texturizado, se procede a recortar la zona de interés y a suavizarlo en Blender. Dado que se ha realizado el decimado en el flujo de Meshroom, no es necesario volver a aplicarlo en Blender. Tampoco se ve necesario retexturizar después del paso por Blender:

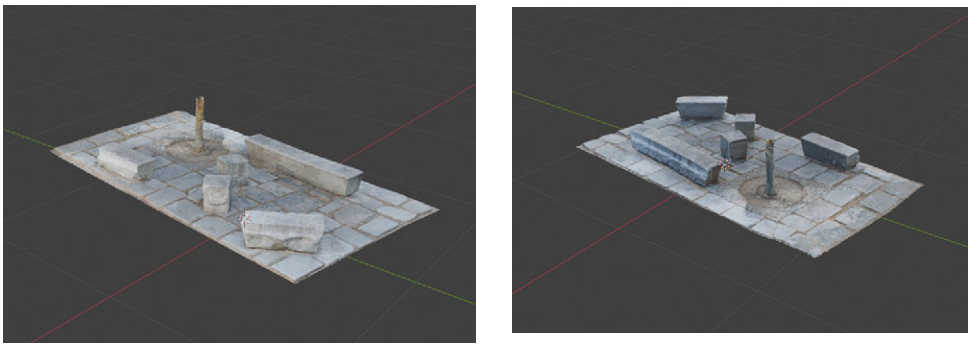

Figura 58: mesh filtrado Bancos Madrid Rio

### 3.8 FUENTE PARLA

La captura de imágenes se realizó entre las 20:39 y las 20:45 del 30 de mayo de 2025 utilizando la cámara del Nothing Phone 2 en modo experto. Se capturaron un total de 254 imágenes en formato JPG. Haciendo un primer filtrado visual, se observa la presencia de múltiples personas en las distintas imágenes, estando algunas en posiciones estáticas. También se puede ver que la fuente esta en funcionamiento, lo que puede ocasionar que el agua provoque inconsistencia entre las imágenes. Estas condiciones hacen que el modelo sea bastante deficiente, como se verá mas adelante. A priori, no se descarta ninguna imagen.

Figura 59: Captura de imágenes Fuente Parla

Con las imágenes tomadas, se comienza con el flujo en Meshroom. Debido a las estimaciones realizadas en la captura de imágenes, se hace un primer tanteo de la reconstrucción aplicando el flujo completo de Meshroom.

- En el nodo FeatureExtraction se utiliza el descriptor DSPSift. Se establece la densidad en Ultra y la calidad en High.
- En el nodo DepthMap y en Texturing se desactiva el downscale.

El resultado es una nube de puntos dispersa con 923.070 puntos y distintas posiciones de captura, junto con el archivo mesh ya texturizado. Analizando la nube de puntos, se puede observar la presencia de mucho ruido alrededor de la fuente, y que el fondo de esta a penas ha tenido coincidencias. Si se observa el mesh, se puede confirmar esto. Se considera que el trabajo de limpieza y retopología implicado en la reconstrucción de este objeto sería bastante laboriosa, por lo que decide descartarse su uso.

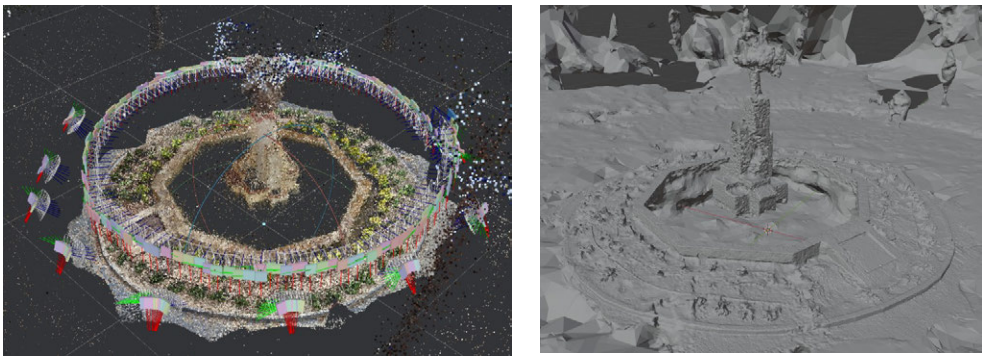

Figura 60: Nube de puntos dispersa y mesh (solid) Fuente Parla

## 3.9 LEÓN PARLA

La captura de imágenes se realizó entre las 20:24 y las 20:30 del 30 de mayo de 2025 utilizando la cámara del Nothing Phone 2 en modo experto. Se capturan un total de 110 imágenes en formato JPG. En un primer filtrado visual, se observa la presencia de múltiples personas en las distintas imágenes, en posiciones constantes entre imágenes. Aun así, no se descarta ninguna imagen ya que estas personas no se encuentran en primer plano.

Figura 61: Captura de imágenes León Parla

Con las imágenes tomadas, se comienza con el flujo en Meshroom. En esta escena, debido a que es pequeña y se espera un resultado bueno, se plantea el uso del flujo completo:

- En el nodo FeatureExtraction se utiliza el descriptor DSPSift. Se establece la densidad y la calidad en Normal.
- En el nodo DepthMap se desactiva el downscale.

Como resultado del flujo se obtiene una nube de puntos dispersa de 182.010 puntos, adecuada para el tamaño de la escena, las poses de la cámara y el mesh ya texturizado. No se descarta ninguna imagen durante el proceso:

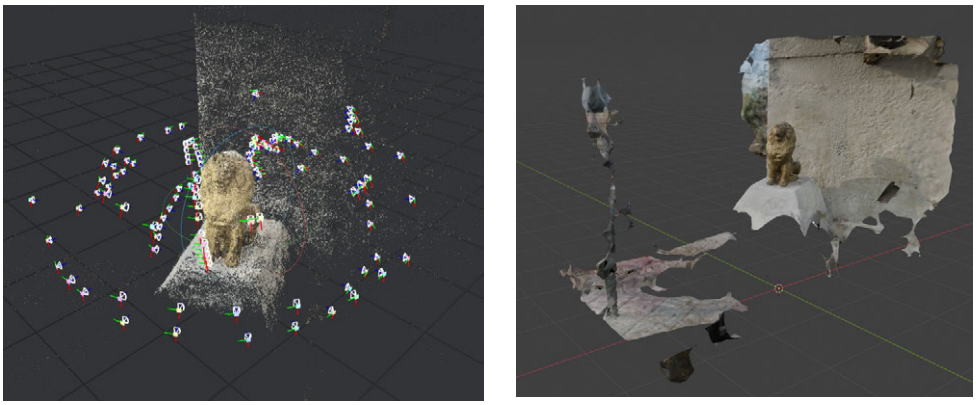

Figura 62: Nube de puntos dispersa y mesh texturizado León Parla

Con el objeto mesh ya generado, se procede a recortarlo y a la aplicación de un decimado y un suavizado en Blender. También, se vuelve a aplicar el texturizado para corregir desplazamientos de las texturas:

Figura 63: Mesh filtrado León Parla

## 3.10 PUENTE DE LA PEDRERA

La captura de imágenes se realizó el día 13 de febrero de 2024 en tres tramos, siendo el primero entre las 10:15 y las 10:38, el segundo entre las 10:41 y las 11:01 y el tercero entre las 11:03 y las 11:17. Para todos estos se utiliza el dron DJI. Se capturan un total de 654 imágenes entre todos los tramos en formato JPG. Observando las imágenes, se puede ver que las imágenes se concentran especialmente en el tramo de puente afectado por la DANA, por lo que se estima que las zonas mas próximas a los estribos estarán peor definidas. A esto hay que añadir que, debido a la posición de la cámara en

el dron y a factores situacionales, no hay imágenes de debajo de los arcos y la presencia de múltiples operarios trabajando en el puente, así como de maquinaria. Se decide no descartar ninguna imagen ya que en este caso, se pierde más información de la que se puede limpiar.

Figura 64: Captura de imágenes Puente de la Pedrera

Con las imágenes tomadas, se da comienzo con el flujo en Meshroom. Para esta escena, dado que se sabe de antemano el resultado que se puede obtener debido a la falta de información en los extremos, se plantea el uso del flujo completo:

- En el nodo FeatureExtraction se utilizan descriptores sift_float, sift_upright y DSPSift. Se establece la densidad en Ultra y la calidad en High.
- En el nodo DepthMap y en Texturing se desactiva el downscale.

La ejecución de este proceso da una nube de puntos dispersa de 8.441.504 puntos, bastante densa debido al uso de múltiples descriptores simultáneos. En el nodo FeatureMatching se descartan 2 imágenes.

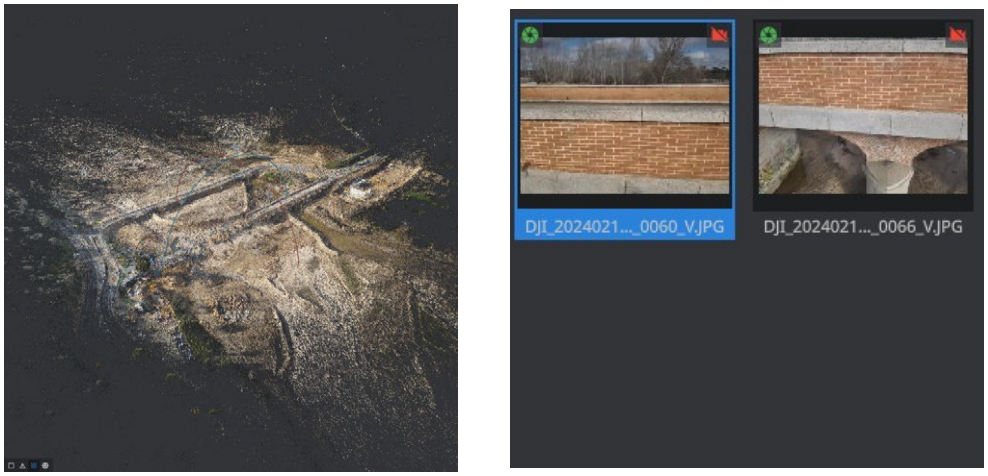


Figura 65: Nube de puntos dispersa e imágenes descartadas Puente de la Pedrera

Al haber ejecutado el flujo completo, también se obtiene un mesh ya texturizado. Como ya se comento en la captura de imágenes, estas no cuentan con tanta cobertura en los extremos y en los arcos. También se observa que el modelo se encuentra escalado debido al uso de un modulo RTK en la captura:

Figura 66: Objeto mesh Puente de la Pedrera

Hacer que este modelo pueda ser utilizable conllevaría realizar un trabajo de retopología intensivo y la captura de más imágenes, por lo que se descarta su uso como escena completa. En cambio, la parte central del modelo y el suelo se encuentra bastante detallada, por lo que se podría aprovechar su uso. Se procede a recortar y limpiar estas zonas para su uso, haciéndoles un decimado y un suavizado. También se reaplica el texturizado en Meshroom.

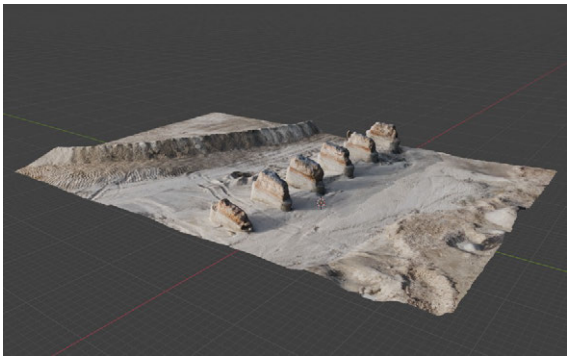

Figura 67: Objeto filtrado Puente de la Pedrera

## 3.11 COLUMNA ENTRADA

Con esta escena, se comienza con la primera de las del jardín de El Capricho. Como se ha comentado anteriormente, la captura de gran parte de estas se realizó con múltiples cámaras y en múltiples días. En concreto, esta fue capturada por primera vez el día 7 de junio de 2025:

1. Sony DSC-H400: Se capturan 139 imágenes en formato JPG, entre las 17:49 y las 18:20 (1).
2. Pentax K20D: Se capturan 41 imágenes en formato JPG, en las mismas horas que para la Sony (2).

Se vuelve a capturar una segunda vez el día 14 de junio de 2025:

1. Sony DSC-H400: Se capturan 12 imágenes entre las 16:52 y las 17:03 en formato JPG.

2. Canon EOS: Se capturan 58 imágenes entre las 16:48 y las 17:10 en formato RAW (CR2) (3).
3. LUMIX GF1: Se capturan 73 imágenes entre las 16:44 y las 16:56 en formato JPG y RAW (RW2) (4). La mayoría de estas imágenes se tomaron rotadas 90º en sentido antihorario.

Se toman un total de 323 imágenes con 4 cámaras distintas en dos días distintos. Las condiciones de iluminación entre ambos días son relativamente similares, lo que mejora la consistencia entre imágenes. En un primer análisis visual, se observa sobre todo una marcada diferencia entre las imágenes de por parte de la Sony y la LUMIX con las tomadas por la Pentax y la Canon.

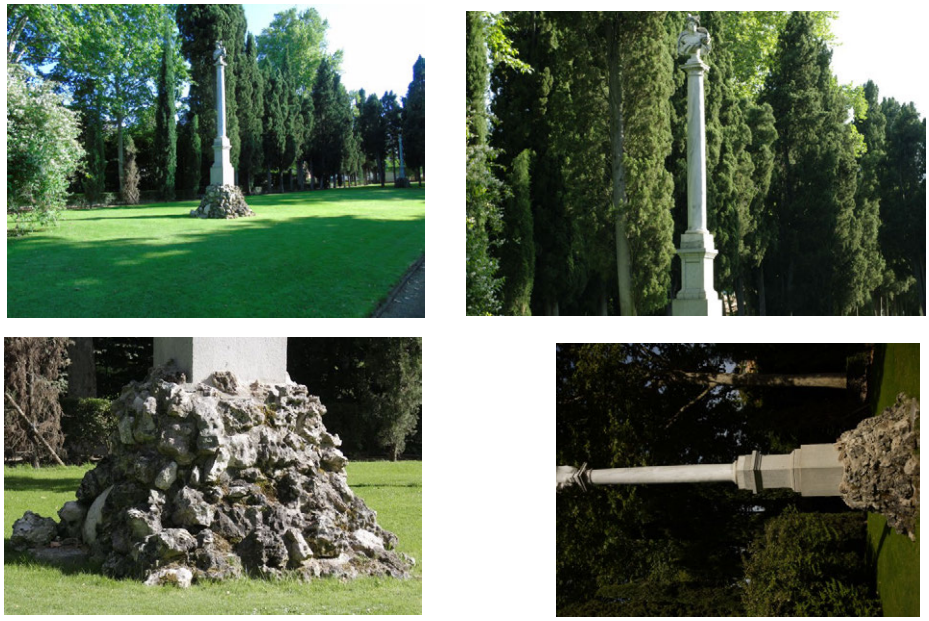

Figura 68: Captura de imágenes Columna entrada (ordenadas 1-4, left-right, up-down)

A pesar de esto, se decide mantener todas las imágenes ya que, a que la situación de la escena hace difícil su captura. La columna encuentra en el centro de una zona de césped rectangular a la que no se puede acceder, dando opción unicamente a capturar desde los caminos habilitados. Dos de ellos se encuentran relativamente lejos de la escultura y un tercero esta cubierto por arboles casi en su totalidad:

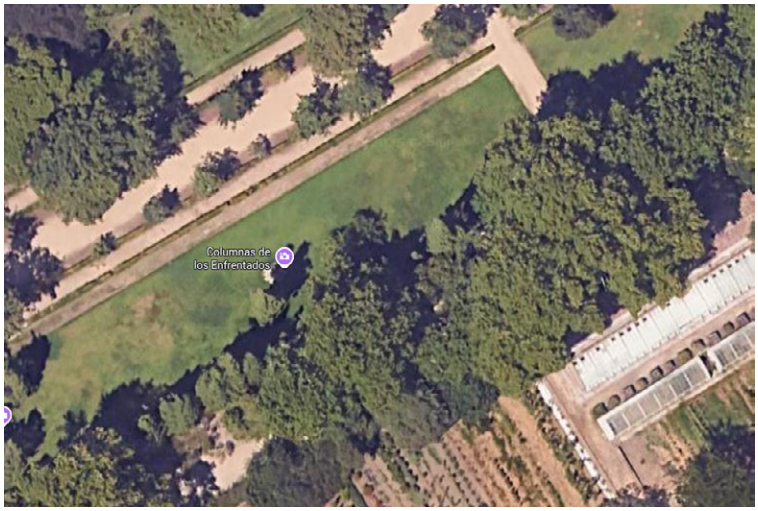


Figura 69: Vista aérea del monumento, Google Maps

Con las imágenes tomadas, se comienza con la primera parte del flujo en Meshroom. En el nodo FeatureExtraction se elige el descriptor sift_float, se establece la densidad en Ultra y la calidad en High. La ejecución de esta primera parte da una nube de puntos dispersa de 195.800 puntos, la cual es bastante reducida. Durante la fase de FeatureMatching se descartan 6 de las imágenes, de las cuales todas ellas corresponden al objeto principal. Aquí es necesario comentar que, debido a la naturaleza de la lente de la cámara Sony, en cada nivel de zoom la focal se reajusta, lo que a nivel de programa produce que se tenga que estimar nuevos parámetros de cámara. En este modelo se ha necesitado capturar imágenes a distintos niveles de zoom, lo que ha resultado en que el programa deba estimar 24 parámetros de cámara distintos.

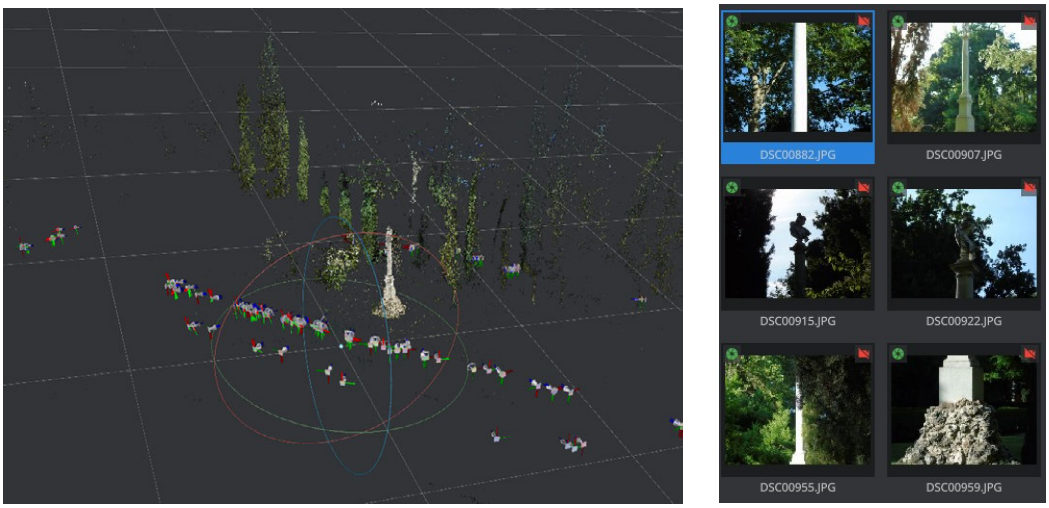


Figura 70: Nube de puntos dispersa e imágenes descartadas Columna entrada

Una vez finalizada la primera parte del flujo, se extrae la nube de puntos dispersa y se limpia en CloudCompare:

Figura 71: Limpieza de nube de puntos dispersa CloudCompare Columna entrada

Con la nube de puntos dispersa filtrada, se procede con la segunda parte del proceso. En los nodos de DepthMap y Texturing se desactiva el Downscale. El resultado de esta parte del flujo es el objeto ya texturizado.

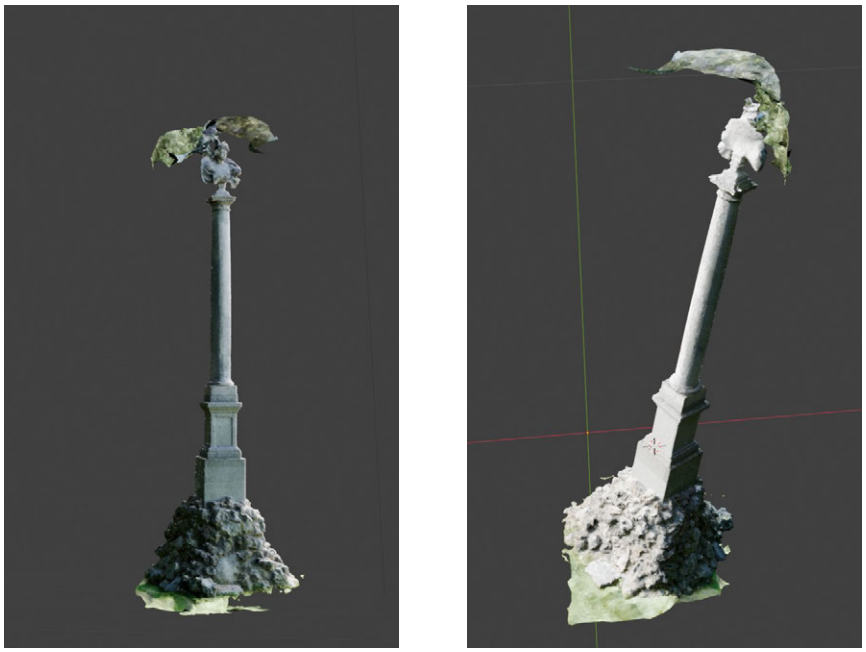

Figura 72: Mesh texturizado Columna entrada.

A pesar de que a priori el resultado no parecía ser bueno, sobre todo por la situación de captura, se acaba resolviendo un mesh texturizado bastante coherente. Se observa en la parte superior artefactos generados al intentar cerrar la geometría. En muchos casos, estos se deben a la falta de contexto del resto del entorno, por lo que se prueba a ejecutar el flujo completo de Meshroom sin filtrar.

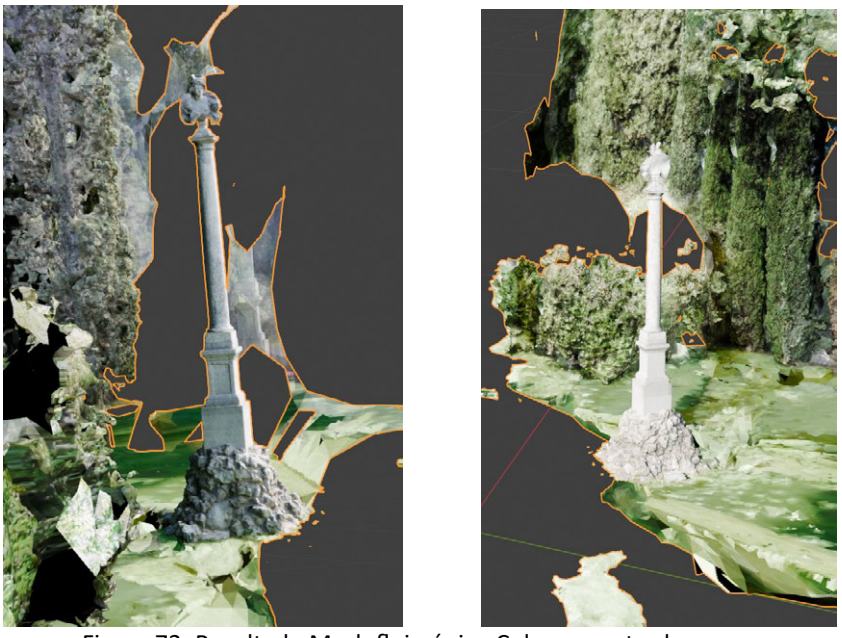

Figura 73: Resultado Mesh flujo único Columna entrada.

Se observa que a nivel geométrico, el programa resuelve mejor la geometría del objeto, por lo que se toma este para la fase de filtrado. El mesh se recorta y se le aplica un decimado y un suavizado en Blender. Al decimar y suavizar los mesh, los mapas UV no

quedan correctamente colocados. Para solucionarlo, se vuelve a texturizar el mesh de nuevo utilizando el nodo de Meshroom.

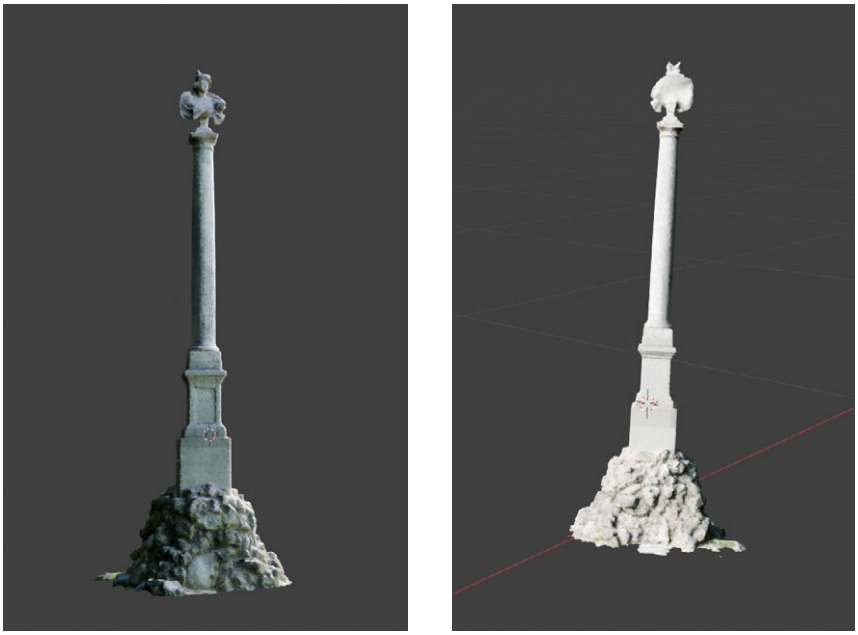

Figura 74: Mesh filtrado Columna entrada.

### 3.12 COLUMNA LAGO

La captura de imágenes se realizó entre las 18:59 y las 29:17 del 7 de junio de 2025 utilizando la Sony DSC-H400 en modo automático, con un ISO fijo a 200. Se capturan un total de 77 imágenes, de las que se descartan 2 de ellas debido a que salen borrosas. En una primera inspección visual, se puede observar que varias de ellas se encuentran subexpuestas. Estas no se descartan ya que el modelo cuenta con pocas imágenes.

Figura 75: Captura de imágenes Columna lago

Con las imágenes tomadas, se comienza con la primera parte del flujo en Meshroom. En el nodo FeatureExtraction se elige el descriptor sift_float, se establece la densidad en Ultra y la calidad en High. La ejecución de esta primera parte da una nube de puntos dispersa de 196.588 puntos, la cual es bastante reducida. Durante la fase de FeatureMatching se descarta una de las imágenes (relativamente lejos de la zona de interés). Como ha pasado en modelos anteriores, la cámara Sony en cada nivel de zoom reajusta la focal, lo que a nivel de programa produce que se tenga que estimar nuevos parámetros de cámara. En este modelo se han tenido que estimar 10 parámetros de cámara distintos.

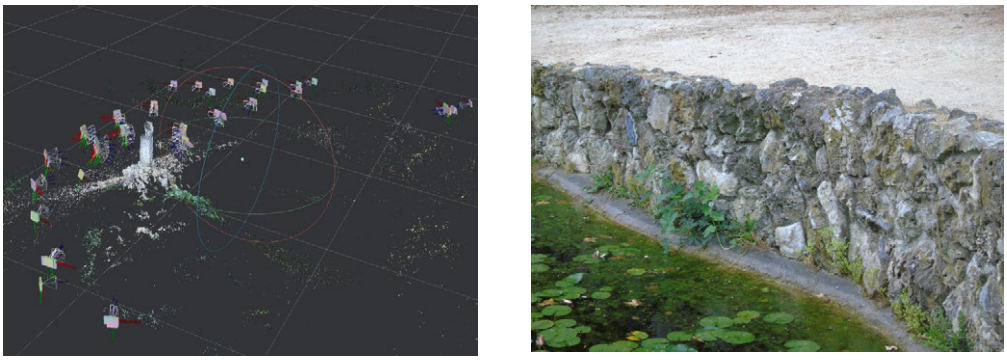

Figura 76: Nube de puntos dispersa e imagen descartada Columna lago

Una vez que finaliza esta primera parte del flujo, se extrae la nube de puntos dispersa y se limpia en CloudCompare:

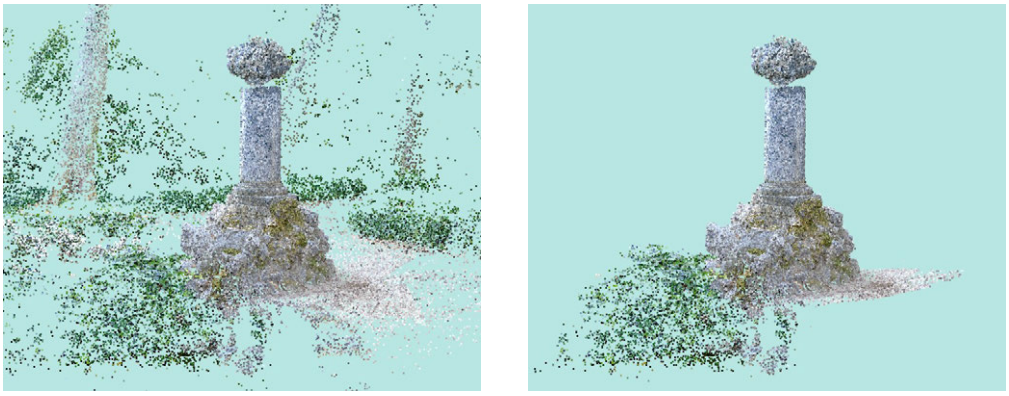

Figura 77: Limpieza de nube de puntos dispersa CloudCompare Columna lago

Se puede apreciar que esta se encontraba relativamente limpia. Solo fue necesario eliminar puntos aislados. Con la nube de puntos dispersa filtrada, se procede con la segunda parte del proceso. En los nodos de DepthMap y Texturing se desactiva el Downscale. El resultado de esta parte del flujo es el objeto ya texturizado.

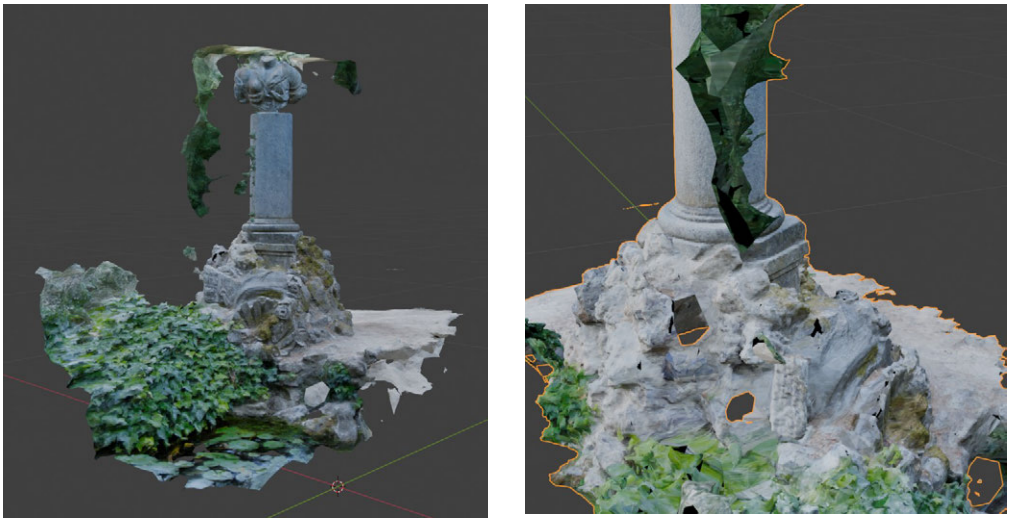

Figura 78: Mesh texturizado Columna lago

De la misma forma que para Columna Lago, se observa la presencia de muchos artefactos y agujeros. Se vuelve a probar a hacer la reconstrucción utilizando el flujo completo:

Figura 79: Mesh texturizado por flujo completo Columna lago

Se observa el mismo resultado. El contexto adicional del entorno hace que la geometría del objeto de interés cierre correctamente. Se procede a recortar, decimar y suavizar en Blender. Después, se vuelve a retexturizar usando el nuevo mesh para evitar el desplazamiento de texturas.

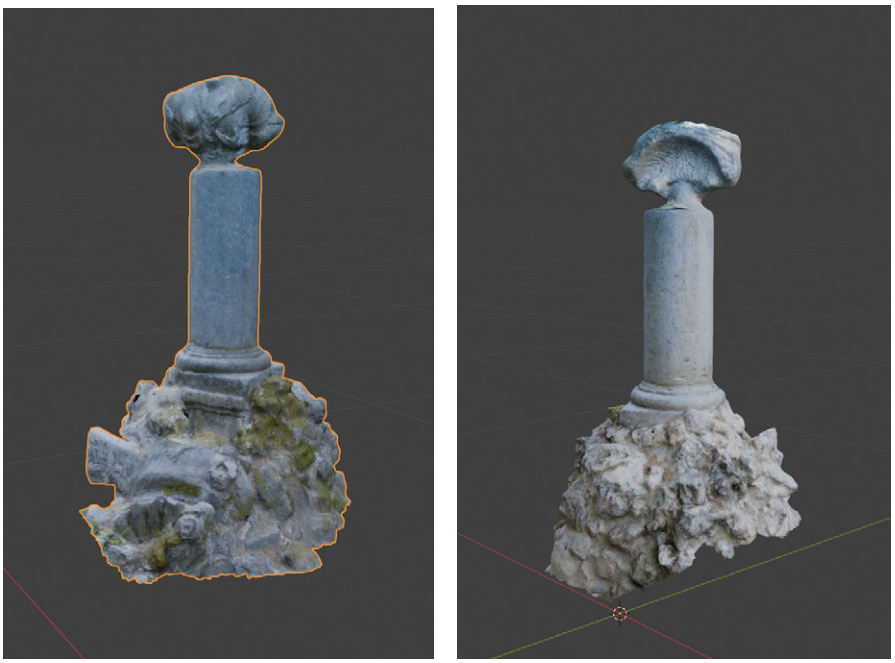

Figura 80: Objeto filtrado Columna lago

### 3.13 TOCÓN ARBUSTO

La captura de imágenes se realizó el día 14 de junio de 2025 utilizando dos cámaras distintas:

1. LUMIX GF1: Se capturan 52 archivos entre las 17:39 y las 17:43 en formato JPG y RAW (RW2) con un ISO fijo a 100.

2. Canon EOS: Se capturan 57 imágenes entre las 17:46 y las 17:51 en formato RAW (CR2) con un ISO fijo a 800 (relativamente alto)

Se capturan un total de 109 imágenes. En una primera inspección visual, se puede observa que la lumix ofrece una paleta de colores más saturada que la Canon. Se descarta una de las capturadas por la Canon debido a que sale movida.

Figura 81: Captura de imágenes Tocón arbusto (Lumix left, EOS right)

Con las imágenes tomadas, se comienza con el flujo en Meshroom. Dado que se trata de un modelo bastante reducido y que la vegetación es un elemento difícil de limpiar en nubes de puntos, se decide realizar el flujo completo.

- En el nodo FeatureExtraction se utiliza el descriptor sift_float, se establece la densidad en Ultra y la calidad en High.
- En el nodo DepthMap y en Texturing se desactiva el downscale.

La ejecución del flujo da como resultado una nube de puntos dispersa de 232.954 puntos construida con la totalidad de las imágenes y un mesh ya texturizado:

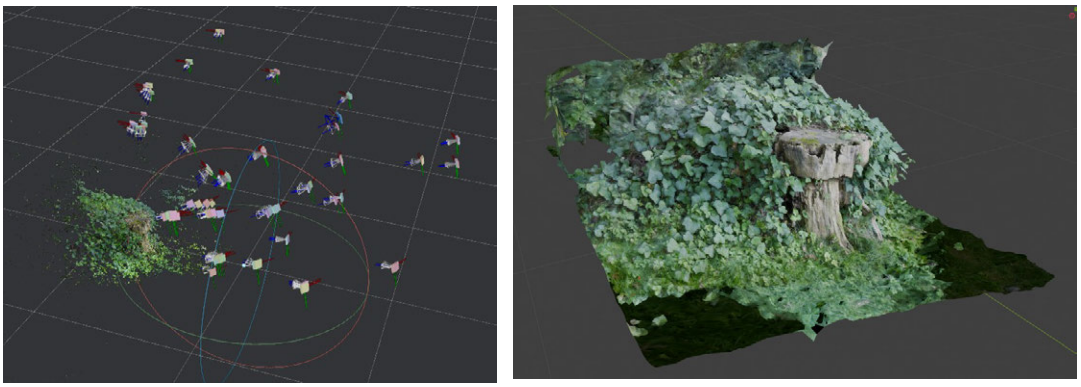

Figura 82: Nube de puntos dispersa y mesh texturizado Tocón arbusto

El objeto se importa en Blender, donde se modifican desperfectos, se aplica un decimado y se suaviza la geometría.

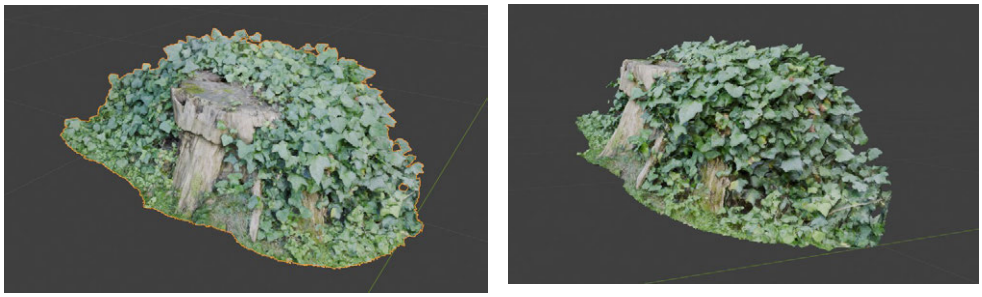

Figura 83: Mesh filtrado texturizado Tocón arbusto

### 3.14 TEMPLO

La captura de imágenes se realizó el día 14 de junio de 2025 utilizando 4 cámaras distintas:

1. Canon EOS: Se capturan 316 imágenes entre las 17:28 y las 18:15 en formato RAW (CR2) con un ISO fijo a 400. Estas se centran sobre todo en la parte exterior del templo.
2. LUMIX GF1: Se capturan 90 imágenes entre las 17:19 y las 17:31 en formato JPG y RAW (RW2) con un ISO fijo a 100. Están imágenes corresponden casi en su totalidad al exterior del templo. Parte de estas imágenes se encuentran rotadas 90º en sentido antihorario.
3. Sony DSC-H400: Se toman 64 imágenes entre las 17:18 y las 17:35 en formato JPG en modo automático con un ISO fijo a 100.
4. Nothing Phone 2: Se toman 476 imágenes entre las 18:31 y las 19:14 en formato JPG con el modo experto. La mayoría de las imágenes se centran en el interior del templo y en los espacios entre columnas.

Se toman un total de 946 imágenes. Debido a su carácter turístico, la captura de imágenes tuvo que interrumpirse y reanudarse en múltiples ocasiones debido a la afluencia de personas. En una primera inspección visual, se observa inconsistencia de color entre las distintas cámaras. No se tiene demasiada información de la parte superior del templo, ya que las imágenes se han sacado con técnicas de fotogrametría terrestre y el templo es demasiado grande para ser cubierto entero. No se descarta ninguna imagen.

Figura 84: Captura de imágenes templo

Con las imágenes tomadas, se comienza con la primera parte del flujo en Meshroom, del que se obtiene una nube de puntos dispersa. Esta es exportada a CloudCompare para su filtrado:

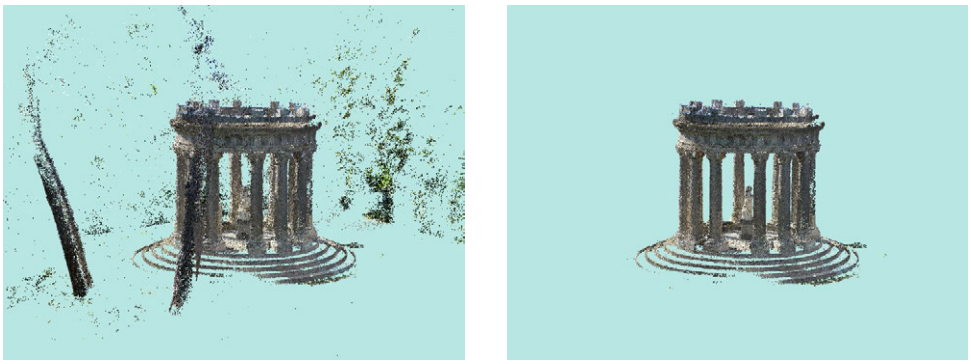

Figura 85: Limpieza nube de puntos dispersa templo, CloudCompare

Con la nube ya limpia, se procede a realizar la segunda fase del flujo, dando lugar a un objeto ya texturizado:

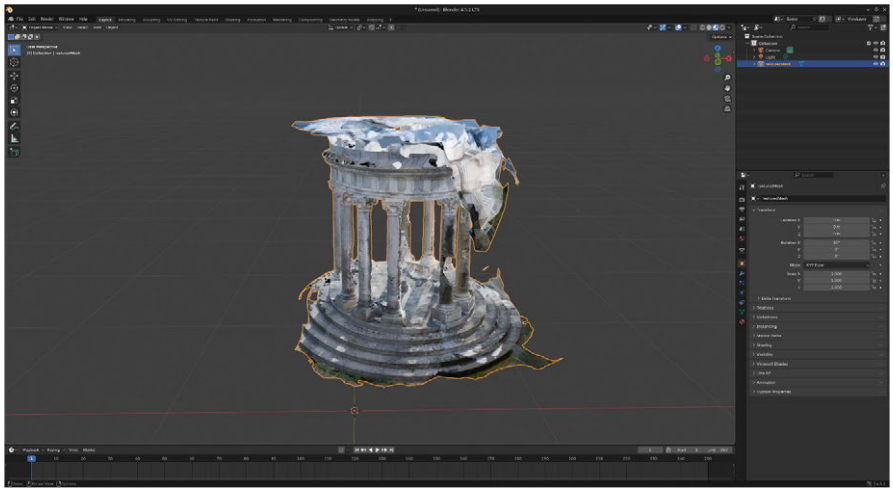

Figura 86: Mesh texturizado templo

Como se puede observar, la parte superior y algunas columnas no han sido reconstruidas correctamente. A parte, se puede observar diversas manchas en el texturizado a causa del empleo de múltiples cámaras. Reconstruir el objeto en unas condiciones aceptables requeriría de mucho tiempo, a parte del uso de imágenes de fotogrametría aérea. Este objeto se descarta.

## 3.15 TEMPLO ESFINGES

La captura de imágenes se realiza en múltiples días y con múltiples cámaras. La primera captura se realiza el día 7 de junio de 2025:

1. Nothing Phone 2: Se toman 283 imágenes entre las 20:26 y las 20:45 en formato JPG con el modo experto. Están se centran de manera general en toda la escena.
2. Sony DSC-H400: Se capturan 69 imágenes entre las 19:31 y las 19:39 en formato JPG en modo automático con un ISO fijo de 200. Estas se centran especialmente en las escultura del centro y de los laterales, debido a su zoom óptico.

La segunda captura se realiza el 14 de junio de 2025:

1. Sony DSC-H400: Se toman 42 imágenes entre las 17:07 y las 17:12 en formato JPG en modo automático con un ISO fijo de 100. Estas se centran especialmente

en las escultura del centro y de los laterales para reforzar las tomadas el primer día.

2. Canon EOS: Se sacan 123 imágenes en formato RAW (CR2) entre las 17:14 y las 18:15 con valores de ISO entre los 400 y 800 (relativamente altos para la escena, pero que garantizaban una velocidad de obturación rápida). Estas se centran en la escena de manera general.

Al igual que pasa con las otras escenas, el uso de múltiples cámaras provoca discrepancias en la gama de colores de las imágenes. Se observa las esfinges son brillantes, lo que puede condicionar

Este modelo quedó corrupto al ser movido en un USB. No se pudo recuperar ningún tipo de información sobre este, así que no se contempla en el trabajo. Se deja reflejada su captura y unas imágenes.

Figura 87: Captura de imágenes templo esfinges

### 3.16 BUSTO 1

La captura de imágenes se realiza en múltiples días y con múltiples cámaras. La primera captura se realiza el día 7 de junio de 2025:

1. Nothing Phone 2: Se toman 163 imágenes entre las 19:25 y las 19:30 en formato JPG con el modo experto.

La segunda captura se realiza a fecha de 14 de junio de 2025:

1. Nothing Phone 2: Se toman 116 imágenes entre las 17:07 y las 17:12 en formato JPG en modo experto.

2. Canon EOS: Se sacan 45 imágenes en formato RAW (CR2) entre las 17:21 y las 17:23 con valores de ISO 800 (relativamente altos para la escena, pero que garantizaban una velocidad de obturación rápida).

Se sacan un total de 342 imágenes. En una primera inspección visual, se observa que el desfase horario entre días provoca sombras en el segundo día que no se encuentran presente en el primero. Se observa también que el perfil de color de la Canon disocia mucho con los de las imágenes del Nothing Phone 2. Dado que son pocas imágenes y con un valor de ISO susceptibles e generar ruido, estas son descartadas.

Figura 88: Captura de imágenes Busto 1

Con las imágenes tomadas, se comienza con la primera parte del flujo en Meshroom. En el nodo FeatureExtraction se elige el descriptor DSPSift, se establece la densidad en Ultra y la calidad en High. La ejecución de esta primera parte da una nube de puntos dispersa de 1.301.463 puntos. Durante el proceso, no se descarta ninguna imagen.

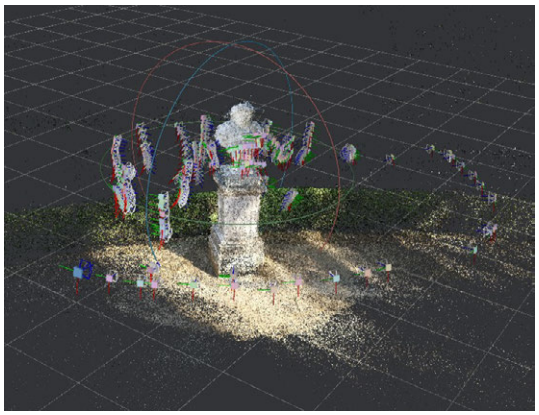

Figura 89: Nube de puntos dispersa y poses estimadas Busto 1

Una vez que finaliza esta primera parte del flujo, se extrae la nube de puntos dispersa y se limpia en CloudCompare:

Figura 90: Limpieza en CloudCompare Busto 1

Con la nube de puntos dispersa filtrada, se procede con la segunda parte del proceso. En los nodos de DepthMap y Texturing se desactiva el Downscale. El resultado de esta parte del flujo es el objeto ya texturizado.

Figura 91: Objeto texturizado Busto 1

El objeto se importa en Blender, donde se modifican desperfectos, se aplica un decimado y se suaviza la geometría, quedando listo para su implementación en Unreal Engine.

Figura 92: Objeto filtrado Busto 1

### 3.17 BUSTO 2

Por último llegamos al busto 2. La captura de imágenes se realiza en múltiples días y con múltiples cámaras. La primera captura se realiza el día 7 de junio de 2025:

1. Nothing Phone 2: Se toman 155 imágenes entre las 20:41 y las 20:44 en formato JPG con el modo experto.
2. Pentax K20D: Se toman 91 imágenes solapadas con la captura realizada por el Nothing Phone 2 con valores de ISO de 280.

La segunda captura se realiza el 14 de junio de 2025:

1. Nothing Phone 2: Se toman 115 imágenes entre las 19:00 y las 19:05 en formato JPG en modo experto.
2. Canon EOS: Se sacan 36 imágenes en formato RAW (CR2) entre las 18:07 y las 18:09 con valores de ISO 800 (relativamente altos para la escena, pero que garantizaban una velocidad de obturación rápida).

Se sacan un total de 397 imágenes. En una primera inspección visual, el desfase horario entre días, al igual que en el caso anterior provoca diferencias en la iluminación. Se observa también que el perfil de color de la Canon disocia mucho con los de las imágenes del Nothing Phone 2. Dado que son pocas imágenes y con un valor de ISO susceptible de generar ruido, estas son descartadas.

Figura 93: Captura imágenes Busto 2

Con las imágenes tomadas, se comienza con el flujo en Meshroom. Se decide aplicar el flujo completo ya que se espera que, debido a las diferencias visuales entre imágenes, se espera que el modelo resulte fallido. Este proceso da como resultado una nube de puntos dispersa junto con las posiciones de captura, así como un archivo mesh texturizado. Se observa que, como se podía observar en la captura, las texturas han quedado muy distorsionadas debido al uso de cámaras con perfiles de color muy distintos. A parte, también se puede observar la presencia de agujeros en el objeto principal, el cual tiene el máximo de solapamiento. Esto es fruto de ruido en la nube de puntos dispersa, haciendo que la geometría también se haya visto comprometida

Figura 94: Mesh texturizado Busto 2

# 4 INTEGRACIÓN EN UNREAL ENGINE 5

Una vez que se tienen todos los objetos ya digitalizados y limpios, es turno de probar su integración en el motor de Unreal Engine 5. En una primera iteración de esta memoria, el proceso de importación de objetos en el motor de Unreal resultó fallido debido a una incompatibilidad entre los drivers de NVIDIA y la versión 5.6.0 de Unreal en Linux. Al momento de importar los objetos, el programa se detenía automáticamente, independientemente del formato de estos. Ha sido necesario mover esta fase a un entorno Windows donde esta incompatibilidad no se diera. Esta segunda iteración de la memoria documenta el proceso de integración desde Windows.

Antes de importar objetos en Unreal, es que estos se encuentren en un formato compatible. Meshroom por defecto exporta los archivos mesh texturizados en formato obj, acompañados de un archivo mtl donde se almacena la ruta relativa a las texturas. Las texturas son exportadas por defecto como imágenes en formato exr. Unreal Engine no acepta como textura el uso de archivos exr, por lo que es necesario asegurarse de que estas se encuentran en otro formato más estandarizado, como png. Para ello, se hace uso del script ConvertEXRtoPNG para convertirlas sin tener que volver a ejecutar el nodo de Texturing.

Con las texturas en un formato reconocible, el objeto se “empaqueta” en un contenedor que se pueda importar fácilmente en Unreal. Como se ha comentado anteriormente, El objeto texturizado esta formado por la geometría en obj, las texturas en formato imagen (png tras la conversión) y un archivo que empareja el objeto con las texturas. Tener todos estos por separados puede llevar a errores de importación, por lo que se opta por convertir estos objetos a formato glb. Este formato, variante de glTF permite empaquetar en un solo archivo geometría, texturas, materiales, animaciones y escenas completas. La conversión de formato se lleva a cabo en Blender.

Con los objetos convertidos a glb, ya se puede proceder con la integración en Unreal. Para ello, se crea un proyecto en Unreal utilizando la platilla de jugador en tercera persona. Esto define un espacio y un personaje jugable por defecto sobre el que empezar a modificar.

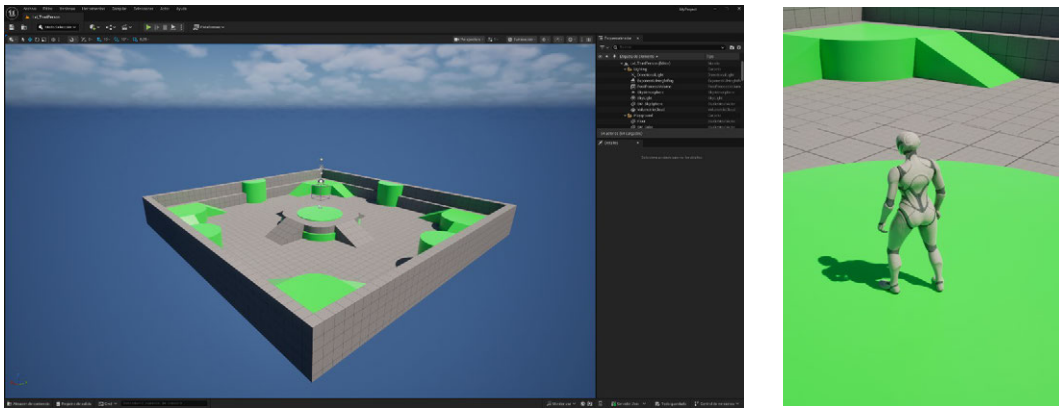

Figura 95: Plantilla por defecto en tercera persona, Unreal Engine

De todos los objetos realizados, se importan los siguientes:

1. Banco Ponteceso
2. Playa de los Cristales
3. Busto 1
4. Columna entrada
5. Columna lago
6. Bancos de Madrid Río
7. Figura Alcorcón 1
8. Figura Alcorcón 2
9. León Parla
10. Puente de la Pedrera
11. Puerto de San Isidro
12. Casas de Tarazona
13. Tocón Arbusto

El proceso de importación consiste en arrastrar los archivos glb al almacén de contenido del proyecto. Los glb se desempaquetan automáticamente en carpetas.

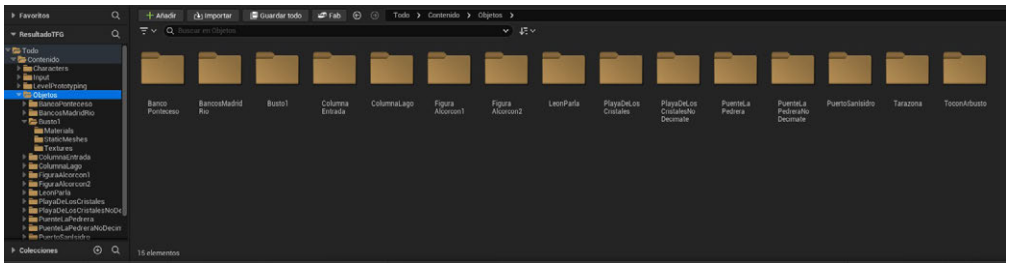

Figura 96: Objetos importados en Unreal Engine

Para incorporar los archivos, basta con arrastrar la malla al espacio de trabajo. Las texturas y los materiales se vinculan automáticamente al objeto. Salvo los objetos tomados por el dron, estos no se encuentran escalados a tamaño real. El carácter jugable que viene por defecto en

esta plantilla mide 1,80m[31], por lo que se puede extrapolar la dimensión de cada objeto a partir de este.

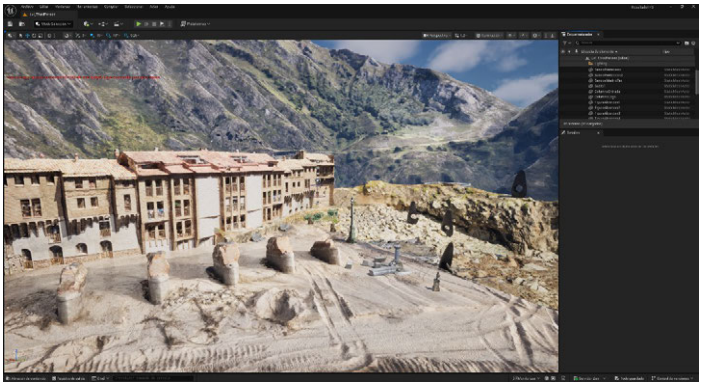

Figura 97: Objetos colocados en Unreal

A priori, el personaje jugable no va a poder interactuar correctamente con los objetos porque, por defecto, en estos no se establece ninguna configuración del motor de colisión. El motor de colisión es el sistema que se encarga de controlar las interacciones físicas entre objetos, definiendo reglas y restricciones para cuando estos entran en contacto, colisionan o se superponen. En Unreal, con establecer la siguiente configuración para cada objeto es suficiente:

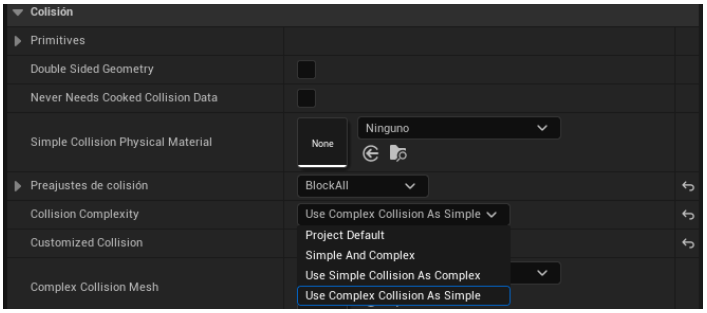


Figura 98: Configuración del motor de colisión

Con esta configuración, el entorno queda correctamente configurado. Para dar continuidad a la escena, se añade agua, haciendo que la escena central quede aislada a modo de isla y efectos de iluminación como niebla ligera y oclusión ambiental.

Figura 99: Escena final

[31]Se recomienda leer *UE5: Complete Guide to Player Scale, Environment Dimensions and Creating to Proportions, World of Level Design*

# 5 RESULTADOS Y COMENTARIOS

Con los resultados obtenidos, se puede llegar a la conclusión de que es viable la aplicación de fotogrametría a la generación de entornos en videojuegos. Dando lugar a las siguientes conclusiones.

El proceso de captura de imágenes define en gran medida la calidad de la reconstrucción. Como se ha observado en los modelos resultantes, las zonas que no quedan bien cubiertas por las imágenes sufren un deterioro notable tanto a nivel de geometría como de textura.

Figura 100: Playa de los cristales en Unreal Engine. Deformación del modelo en los extremos

Los modelos capturados a partir de varias cámaras también resultan penalizados. Aunque Structure From Motion es capaz de llegar a una solución coherente, en las fases de limpieza de la nube de puntos dispersa se observa mayor presencia de ruido a causa de falsos positivos en las coincidencias.

Figura 101: Nubes de puntos dispersas de varias cámaras, CloudCompare

También se observan discrepancias claras de color en la generación de texturas, debido a los diferentes ajustes a la hora de sacar las imágenes y a la calibración de color que aplica la

cámara. Como se recomendó en el marco teórico, es recomendable el uso de una única cámara.

A nivel geométrico, los objetos que han sido correctamente cubiertos cuentan con mallas con un alto nivel de detalle (siempre dependiendo de la distancia de captura), estando compuestos por una gran cantidad de caras triangulares (entre las 2.000.000 y 6.000.000). En muchos casos esto es excesivo, ya que las geometrías complejas penalizan mucho el rendimiento en motores de videojuego. Decimar estas geometrías soluciona el problema, aunque Unreal Engine parece comportarse de manera estable en presencia de mallas muy densas.

A nivel de texturas si el objeto ha sido correctamente cubierto, también se obtiene gran nivel de detalle. Se puede observar que la calidad de la textura depende de la cercanía de la cámara al objeto y del ángulo de captura.

Figura 101: Puerto de San Isidro en Unreal Engine, plano cercano

Este último en especial produce ligeras deformaciones en zonas que a priori han sido correctamente calculadas, pero que el ángulo de captura no es estrictamente paralelo a la normal del objeto. También remarcar que los objetos capturados en zonas con una iluminación que no es homogénea producen sombras en las texturas. La textura puede ser corregida y modificada en Blender, pero tanto las texturas como los mapas UV generados durante la reconstrucción son muy complejos.

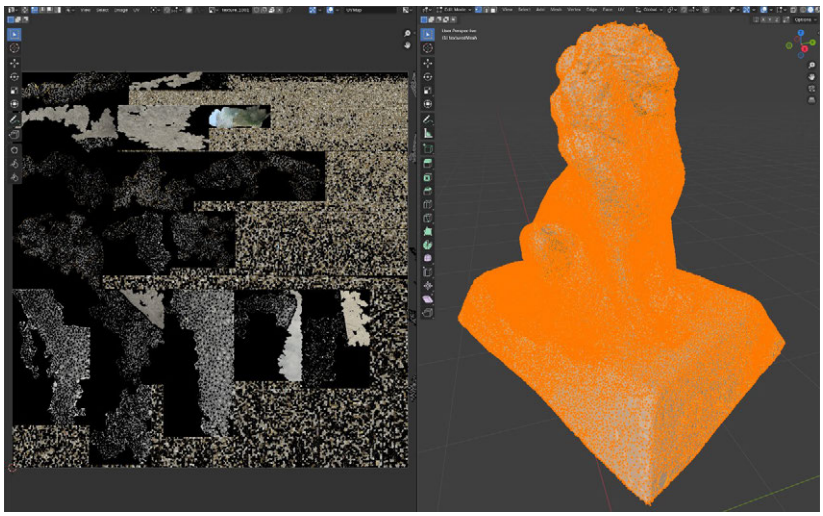

Figura 102: León Parla en el editor de UV, Blender

Sería necesario rebakear la textura y remapear los UV para que sea viable trabajar sobre estos.

Como se ha comentado anteriormente, la importación de modelos a Unreal Engine falló inicialmente debido a una incompatibilidad con los drivers de NVIDIA de Linux. El trabajo pudo continuar en un entorno Windows en esta segunda iteración de esta memoria.

Como conclusión final se puede decir que se ha llegado a un resultado satisfactorio del trabajo y que se han cumplido mayoritariamente los objetivos descritos. Aunque no se hubiera logrado la importación en Unreal, los modelos han podido ser desarrollados correctamente y podrían haber sido importados en otros motores como Unity. Se ha conseguido desarrollar un flujo de trabajo que permita llegar a un resultado satisfactorio, que puede ser replicado en entornos de pago y utilizando otras tecnologías y programas. También es importante mencionar que este documento solo aporta la base para la obtención de un resultado general. Este debería ser posteriomente tratado con técnicas de retopología y de texturización para la obtención de un resultado profesional, pero estas en sí son lo suficientemente complejas para ser tratadas en su propio proyecto.

Este proyecto ha dejado un balance de 88.877 archivos y 1,6TiB de datos en reconstrucción.

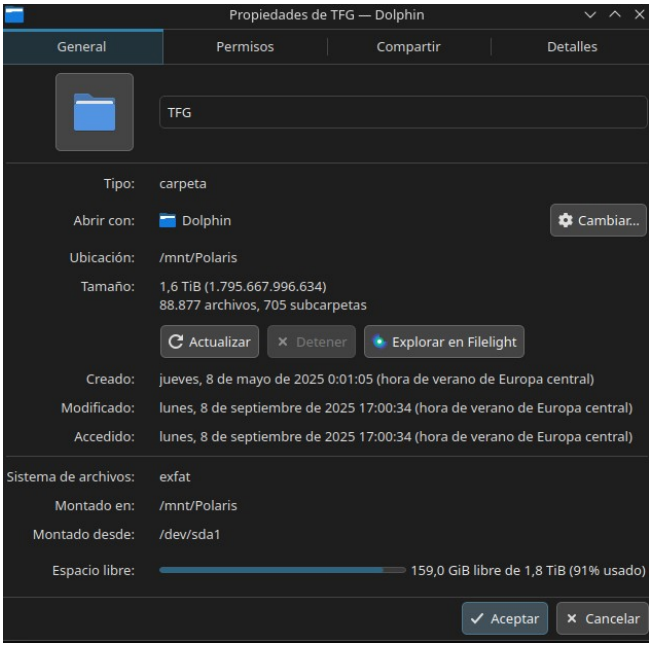

Figura 103: Balance de archivos

Enlace a los recursos utilizados > [Recursos TFG]