

# Generación de nubes de puntos a partir de *stacking* sin información de pose de la cámara

José Julián Camacho Hernández

*jcamacho341@estudiantec.cr*

*Escuela de Ingeniería en Computadores*

*Instituto Tecnológico de Costa Rica*

**Resumen**—Este informe presenta el desarrollo de un algoritmo de *stacking* de nubes de puntos sin información de pose de la cámara, destinado a generar mapas tridimensionales precisos y coherentes. El propósito del estudio es abordar el problema de la combinación de múltiples nubes de puntos generadas a partir de vídeos o imágenes, que, sin una adecuada alineación, resultan en representaciones tridimensionales ineficaces. El diseño del estudio se centró en la implementación de un algoritmo basado en técnicas de visión por computadora y fotogrametría, utilizando principalmente el algoritmo *Iterative Closest Point* (ICP) y la biblioteca *Open3D*. Los principales hallazgos indican que, aunque el algoritmo es capaz de fusionar adecuadamente las nubes de puntos en la mayoría de los casos, se identificaron desafíos significativos en la alineación inicial y en la precisión de los resultados, especialmente con conjuntos de datos complejos. Las pruebas funcionales y unitarias verificaron la robustez del producto, aunque se resaltaron áreas para futuras mejoras, como la optimización del algoritmo y la exploración de técnicas de aprendizaje automático. Como conclusión, la solución desarrollada demuestra ser una herramienta prometedora para la generación de mapas tridimensionales, con aplicaciones potenciales en diversas áreas de la ingeniería y la cartografía, aunque requiere de ajustes y mejoras adicionales para alcanzar su máximo potencial.

**Palabras clave**—Fotogrametría, ICP, Mapas tridimensionales, Nubes de puntos, Pose, *Stacking*

## I. INTRODUCCIÓN

El presente informe aborda la problemática de la generación de mapas tridimensionales a partir de nubes de puntos individuales, sin disponer de información sobre la pose de la cámara en cada instancia de captura. La falta de datos precisos sobre la orientación y posición de la cámara presenta desafíos significativos en el procesamiento, fusión y alineación correcta de las nubes de puntos, ya que estas provienen de diversas imágenes capturadas desde diferentes ángulos y alturas. Esta dificultad se agrava en escenarios donde los drones deben operar bajo condiciones variables de iluminación, presencia de obstáculos y cambios en la topografía del terreno, complicando aún más la generación de mapas tridimensionales coherentes y precisos.

El objetivo fundamental del trabajo es desarrollar un algoritmo de *stacking* que resuelva el problema planteado, considerando sus limitaciones. El producto de *software* se centra en implementar un método para combinar nubes de puntos sin utilizar información de la pose de la cámara, logrando así una nube de puntos global de salida que represente de manera precisa y detallada el entorno capturado. Esta solución es particularmente relevante en aplicaciones como la exploración

de terrenos remotos, cartografía detallada y planificación urbana, donde la generación de representaciones tridimensionales precisas es crucial.

El alcance del trabajo se delimita a la implementación de un Generador de Nubes de Puntos como parte del proyecto mayor titulado "Desarrollo de Aplicaciones de Procesamiento de Vídeo e Imágenes Orientadas hacia Vehículos Autónomos bajo condiciones de visión no ideales para la Realidad Costarricense". Este proyecto implica el uso de drones para capturar vídeos y realizar estimaciones de fondo monocular, generando nubes de puntos individuales por cada imagen. La contribución clave del presente producto radica en la fase final de este proceso, donde las nubes de puntos se combinan para proporcionar un modelo tridimensional consolidado.

El enfoque del presente trabajo no incluye la generación directa de nubes de puntos a partir de imágenes ni la manipulación o interacción directa con la cámara. Se asume que el *hardware* está en condiciones adecuadas y el desarrollo se centra en la combinación y procesamiento de conjuntos de nubes de puntos sin depender de datos específicos de la posición u orientación de la cámara en cada captura.

Este informe se estructura de la siguiente manera: en la sección II se describe el estado del arte y antecedentes del problema. Posteriormente, la sección III detalla la forma en que se resolvió el problema, incluyendo el diseño del algoritmo y su implementación. En la sección IV, se discuten los resultados obtenidos. Finalmente, se presentan las principales conclusiones del proyecto y recomendaciones para trabajos futuros, en las secciones V y VI respectivamente.

## II. ANTECEDENTES

En el ámbito de la generación de mapas tridimensionales a partir de nubes de puntos, especialmente sin información de la pose de la cámara, se han desarrollado múltiples enfoques y tecnologías para abordar los desafíos inherentes a este proceso. A continuación, se presentan los conceptos clave y métodos utilizados en el estado actual de la técnica.

Una nube de puntos es un conjunto de puntos en el espacio que representan la superficie de un objeto. Estas pueden ser obtenidas por medio de distintas técnicas y pueden ser útiles para mapeos de áreas. [1]

En este ámbito, se utilizan drones como herramientas de exploración y recolección de datos espaciales, siendo esencial la generación precisa de mapas tridimensionales. El mapeo

con drones es una tecnología perteneciente al área de la teledetección, que tiene como objetivo crear mapas en 2D y 3D de un área utilizando datos de sensores situados en un dron o vehículo aéreo no tripulado (VANT). Esta técnica busca generar mapas geoespaciales que contienen coordenadas de ubicación específicas del mundo real para cada punto de datos. Los datos de mapas embebidos permiten realizar mediciones del mundo real, facilitando aplicaciones en construcción, agricultura, planificación urbana y estudios topográficos.

La fotogrametría y el LiDAR (*Light Detection and Ranging*) son dos de las técnicas más comunes para mapeos tridimensionales con drones. La segunda corresponde a una tecnología que utiliza láseres para medir distancias desde el sensor en el dispositivo LiDAR hasta los objetos en el entorno, tal como se visualiza en la figura 1. Es un método geomático, ya que mide la distancia a un objeto, ya sea la superficie de la Tierra o un edificio de apartamentos, utilizando luz láser en forma de un haz pulsado estrecho emitido desde una única fuente láser. [1]

La principal ventaja del mapeo con LiDAR es que opera sin depender de la visibilidad del entorno. Además, los sensores LiDAR pueden atravesar áreas con vegetación densa, lo que los hace perfectos para aplicaciones en bosques. Sin embargo, estos sensores son más costosos y pesados en comparación con las cámaras. [2]

Por otro lado, la fotogrametría consiste en crear un modelo tridimensional a escala a partir de un conjunto de fotografías tomadas desde diferentes ángulos. Como se visualiza en la figura 1, el método utiliza fotografías tomadas desde varios ángulos de un objeto para calcular sus dimensiones y generar un modelo 3D.

La fotogrametría es un método más accesible y económico debido a la gran disponibilidad de cámaras. Sin embargo, depende en gran medida de la visibilidad de las características de los datos capturados con cámaras. La visibilidad se ve afectada por la altura a la que navegue el dron y por aspectos ambientales tales como la oscuridad, nubes y niebla. [2]

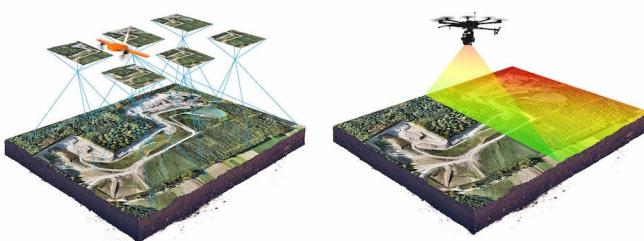


Figura 1. Dos formas de capturar datos desde un dron: fotogrametría (izquierda) y LiDAR (derecha) [1]

En escenarios donde se requiere capturar objetos en movimiento en 3D desde múltiples ángulos o a lo largo del tiempo, se han implementado diversos algoritmos temporales. Un enfoque común es el uso de representaciones en formato de mallas, pero estos métodos presentan limitaciones significativas. Los sensores 3D rara vez proporcionan datos en formato de malla y, además, los algoritmos de mallas asumen un mapa

de profundidad continuo que no mantiene correspondencias fundamentales para el apilamiento de información. [3]

Por otro lado, algunas arquitecturas de redes neuronales utilizan representaciones de nubes de puntos para generar mapas globales. Un ejemplo destacado es el *SyNoRiM*, una tecnología que produce resultados positivos pero enfrenta dificultades para generalizar a nuevos dominios. Además, sus conjuntos de datos de entrenamiento suelen tener cambios de vista relativamente pequeños, lo que limita su aplicabilidad en escenarios más diversos. [3]

El algoritmo *Iterative Closest Point* (ICP) es uno de los métodos más utilizados para el registro rígido de nubes de puntos. Este algoritmo alterna entre la búsqueda de puntos más cercanos en el conjunto objetivo y la minimización de la distancia entre puntos correspondientes, garantizando la convergencia a un alineamiento óptimo local. Sin embargo, el ICP clásico puede experimentar una convergencia lenta debido a su tasa de convergencia lineal. Para mejorar esto, se han desarrollado variantes del ICP que emplean técnicas como la minimización de la distancia de punto a plano y la aproximación cuadrática local de la función de distancia al cuadrado. [4]

Un método destacado que mejora la convergencia del ICP es el presentado por Zhang, Yao y Deng, quienes utilizan la aceleración de Anderson (AA), una técnica numérica eficaz para problemas de optimización en gráficos por computadora. Esta solución tiene la limitación de que depende de una buena alineación inicial. [4]

Otro de los estudios presentados sobre el tema fue realizado por Chen y Medioni, quienes publicaron un esquema de ICP similar utilizando un procedimiento de emparejamiento diferente basado en el vector normal de la superficie. Esta formulación solo es aplicable a puntos en superficies. [5]

Otra variante importante es el *Trimmed ICP* (TrICP), que utiliza de manera consistente el enfoque de Mínimos Cuadrados Ajustados (LTS, por sus siglas en inglés) en todas las fases de la operación, lo que mejora la robustez del algoritmo frente a datos ruidosos o incompletos. [6]

El trabajo previo más completo en relación con el objetivo de este proyecto es el denominado *PCStacking*. Este método aprovecha soluciones iterativas tanto en la estimación de la posición de la cámara como en los parámetros de calibración interna obtenidos durante el ajuste. El principio básico del algoritmo de apilamiento es calcular la mediana de las coordenadas Z de cada punto para múltiples modelos fotogramétricos, resultando en una nube de puntos con una precisión superior a cualquiera de las nubes de puntos individuales. [7]

### III. DESCRIPCIÓN DE LA SOLUCIÓN DESARROLLADA

En la presente sección, se detalla la solución implementada según diversas perspectivas que guiaron el desarrollo del algoritmo de *stacking*.

#### III-A. Casos de uso

A continuación, se detalla la perspectiva contextual del producto desarrollado mediante el diagrama de casos de uso

según el estándar UML, que se visualiza en la figura 2. De esta figura se derivan los siguientes casos de uso:

■ **Caso de uso 1:**

- **Nombre:** Aplicación del algoritmo de *stacking* por parte del Sistema de Mapeo 3D con Vehículos Autónomos.
- **Actores:**
  - Sistema de Mapeo 3D con Vehículos Autónomos: Corresponde al sistema del cual es parte el presente producto.
  - Sistema de archivos: Corresponde al sistema que gestiona y almacena los archivos.
- **Entradas:** Nubes de puntos en archivos de formato *pcd*.
- **Salidas:** Nube de puntos global en un archivo de formato *pcd*.
- **Descripción:** Este caso de uso representa la aplicación del algoritmo de *stacking* por parte del Sistema de Mapeo 3D con Vehículos Autónomos. Para llevarlo a cabo, dicho sistema brinda como insumo las puntos individuales generadas en etapas previas y utiliza el presente producto para realizar la combinación de las mismas. Finalmente, se produce una nube de puntos global que representa la escena tridimensional capturada.

■ **Caso de uso 2:**

- **Nombre:** Aplicación del algoritmo de *stacking* por parte del usuario.
- **Actores:**
  - Usuario: corresponde a la persona usuaria del producto. La persona supervisora, en este caso, tendría la misma interacción de caso de uso que un usuario convencional.
  - Sistema de archivos.
- **Entradas:**
  - Nubes de puntos en archivos de formato *pcd*.
  - Archivo de configuración.
- **Salidas:** Nube de puntos global en un archivo de formato *pcd*.
- **Descripción:** Este caso de uso implica la aplicación directa del algoritmo de *stacking* por parte del usuario, quien selecciona las nubes de puntos individuales y, opcionalmente, ajusta el proceso mediante un archivo de configuración. La ejecución del algoritmo resulta en la generación de una nube de puntos global que representa la combinación coherente de las nubes de puntos de entrada.

■ **Caso de uso 3:**

- **Nombre:** Visualización y evaluación de resultados.
- **Actores:**
  - Usuario.
  - Persona supervisora: Corresponde al profesor asesor encargado de la evaluación del producto.
- **Entradas:** Nube de puntos global en un archivo de formato *pcd*.

- **Salidas:** Visualización y/o evaluación de los resultados.

- **Descripción:** Este caso de uso implica que el usuario visualice los resultados generados por el algoritmo de *stacking*. Si el usuario es la persona supervisora, se encarga de visualizar el resultado y realizar una evaluación del mismo.

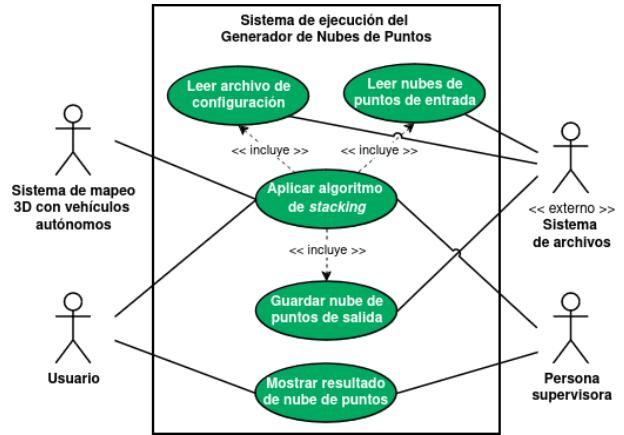


Figura 2. Diagrama de casos de uso UML del producto de *software* desarrollado.

### III-B. Composición

Seguidamente, se detalla la composición del producto implementado. Esta se documenta mediante el diagrama de componentes UML, que se presenta en la figura 3. A continuación, se describen cada uno de los componentes del sistema:

- **Lector de configuraciones:** Este componente se encarga de leer y procesar las configuraciones necesarias para la ejecución del algoritmo de *stacking*. Incluye parámetros como umbrales de aceptación, rutas de entrada y de salida, entre otros parámetros de ajuste.
- **Lector de nubes de puntos:** Este componente se encarga de leer las nubes de puntos individuales provenientes de archivos en formato *pcd*. Su función principal es cargar los datos para que sean procesados por el resto del sistema.
- **Preprocesador de nubes de puntos:** El preprocesador se encarga de realizar operaciones iniciales sobre las nubes de puntos para mejorar de la calidad de los datos.
- **Submuestreo y detección de outliers:** Realiza el submuestreo de las nubes de puntos para reducir la cantidad de datos a procesar, eliminando puntos redundantes. Además, detecta y elimina puntos atípicos que puedan afectar la calidad de la nube de puntos combinada.
- **Analizador de combinabilidad por comparación directa:** Este subcomponente evalúa la posibilidad de combinar diferentes nubes de puntos mediante la comparación directa de la distancia entre ellas. Ayuda a identificar nubes de puntos que se puedan alinear adecuadamente.
- **Analizador de combinabilidad por ML:** Utiliza técnicas de aprendizaje automático como la regresión

logística para predecir la combinabilidad de las nubes de puntos. Aprende de datos previamente combinados y mejora la precisión de la predicción en el proceso de *stacking*.

- **Procesador de algoritmo de stacking:** Se encarga de apilar los conjuntos preprocesados.

- **ICP pairwise registration:** Realiza la alineación inicial de pares de nubes de puntos utilizando el algoritmo *Iterative Closest Point* (ICP). Este paso es crucial para obtener una alineación aproximada antes de la combinación total.
- **ICP full registration:** Refina la alineación de todas las nubes de puntos combinadas usando ICP, asegurando una integración más precisa y coherente en la nube de puntos global.
- **Optimizador de grafo de poses:** Optimiza el grafo de poses, que representa la relación espacial entre las nubes de puntos individuales. Este subcomponente ajusta las poses para minimizar los errores acumulados en el proceso de combinación.

- **Generador de archivo de salida:** Este componente se encarga de generar un archivo de salida en formato *pcd* que contiene la nube de puntos global combinada. Proporciona un formato accesible y ampliamente utilizado para su posterior análisis o visualización.
- **Visualizador de resultados:** Ofrece una interfaz gráfica o herramienta para visualizar los resultados del proceso de *stacking*. Permite a los usuarios evaluar la calidad y coherencia de la nube de puntos global generada.

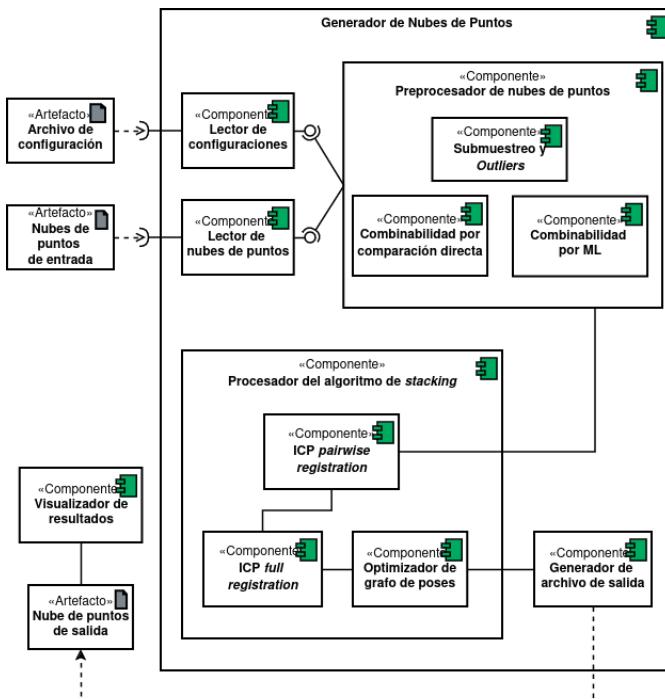


Figura 3. Diagrama de componentes UML del producto desarrollado.

### III-C. Algoritmo desarrollado

En la presente sección, por medio del diagrama de flujo que se visualiza en la figura 5, se describen las operaciones que llevan a cabo los componentes del producto previamente descritos.

#### III-C1. Lectura de archivo de configuración y carga de nubes de puntos

El algoritmo del producto desarrollado inicia cuando el usuario da la indicación de ejecutar la solución, y como se presenta en el diagrama, el archivo de configuración es la primera información que se procesa. Este contiene valores que ajustan el algoritmo de *stacking* que se realizará, así como rutas de los archivos de entrada y salida.

Dicho archivo de configuración fue diseñado de la siguiente manera:

```
{
    "input_path": ".../data/cloud-points",
    "config_params": {
        "voxel_size": 0.02,
        "remove_outliers_params": {
            "nb_neighbors": 20,
            "std_ratio": 2.0
        },
        "combinability_threshold": 0.5
    },
    "output_file": "nube_pcd_combinada.pcd"
}
```

- **input\_path:** Ruta al directorio donde se encuentran los archivos de nubes de puntos que se van a procesar.
- **voxel\_size:** Define el tamaño del *voxel* para el muestreo de las nubes de puntos mediante una técnica utilizada para reducir la cantidad de puntos en una nube de puntos.
- **nb\_neighbors:** Define el número de vecinos a considerar para cada punto al evaluar si es un *outlier*. Un valor más alto considera más puntos vecinos y puede resultar en una eliminación más estricta de puntos atípicos.
- **std\_ratio:** Consiste en el *ratio* estándar utilizado para definir el umbral de eliminación de *outliers*. Este valor determina cuán lejos de la media deben estar los puntos para ser considerados atípicos. Un valor más bajo es más estricto y eliminará más puntos que se consideran atípicos.
- **combinability\_threshold:** Umbral para determinar la combinabilidad de las nubes de puntos. Este parámetro define la sensibilidad del algoritmo para decidir si dos nubes de puntos pueden combinarse eficazmente. Un umbral más bajo puede resultar en una combinación más estricta, mientras que un umbral más alto permite una mayor flexibilidad.
- **output\_file:** Nombre del archivo *.pcd* donde se guardará la nube de puntos combinada. Este es el resultado final del procesamiento y debe especificar la ruta y el nombre del archivo de salida.

A partir de la indicación de ejecución, el producto se encarga de leer y procesar el archivo de configuración. Para eso, debe buscarse en una ruta definida y analizar su contenido. En caso de que dicho archivo cuente con un formato correcto,

se utilizarán los valores establecidos y se mostrará al usuario un mensaje de éxito en la lectura. Por el contrario, el programa seleccionará configuraciones por defecto.

Seguidamente, se buscan las nubes de puntos de entrada en las rutas establecidas en el archivo de configuración y las procesa para tomarlas como insumo para las siguientes etapas. Al realizar su lectura se presentan dos casos: si el formato es incorrecto, el programa debe mostrar un mensaje de error y finalizar su ejecución; en caso contrario, se continua con el flujo de funcionamiento.

**III-C2. Preprocesamiento:** En la siguiente etapa, el producto ejecuta un método de preprocesamiento de las nubes de puntos de entrada que fueron leídas. Este incluye pasos como el **submuestreo** por medio de la técnica de *voxel downsampling*, que se encarga de disminuir la densidad de las nubes por medio de regiones establecidas por el tamaño del *voxel*. Adicionalmente, se detectan **puntos atípicos** utilizando estadísticas de distancia local.

Como parte del preprocesamiento, el producto evalúa la posibilidad de combinar diferentes nubes de puntos, ya sea mediante la comparación directa de la distancia entre ellas, o por medio del aprendizaje automático de datos previamente combinados. Esto ayuda a identificar nubes de puntos que se puedan alinear adecuadamente, y así eliminar las que no tengan suficientes coincidencias.

Cabe destacar que el modelo de aprendizaje es útil únicamente cuando es entrenado y predice sobre conjuntos de datos en los que se sabe que las nubes de puntos no son muy diferentes entre sí. La comparación directa es un método más general y funcional con la mayoría de los conjuntos de datos al establecer de manera correcta los parámetros en el archivo de configuración.

**III-C3. Iterative Closest Point:** Seguidamente, se lleva a cabo el algoritmo principal del producto: el apilamiento de las nubes de puntos preprocesadas. Dicho algoritmo corresponde al *Iterative Closest Point (ICP)*. Este es uno de los métodos clásicos para el registro rígido, que es el problema de alinear un conjunto de puntos fuente con un conjunto de puntos objetivo. [8]

Se utiliza el proceso denominado **registro múltiple**, que implica la alineación de varias piezas de geometría en un espacio global. En este caso, la entrada consiste en un conjunto de nubes de puntos  $\{P_i\}$  obtenidas de diferentes capturas sin información sobre la pose de la cámara. El objetivo es obtener un conjunto de transformaciones rígidas  $\{T_i\}$ , de manera que las nubes de puntos transformadas  $\{T_i P_i\}$  se alineen correctamente en el espacio global.

La implementación de la registro múltiple se realiza haciendo uso de la biblioteca *Open3D*, que proporciona herramientas eficientes para el procesamiento de nubes de puntos.

**III-C3a. Registro por pares:** En su primera etapa, se aplica específicamente la modalidad de punto a plano (*Point-to-plane ICP*). El flujo general de este enfoque de ICP se presenta en el recuadro punteado en el diagrama de la figura 5. En general, este algoritmo itera sobre dos pasos: [9]

1. Encontrar el conjunto de correspondencias  $\kappa = \{(p, q)\}$  a partir de la nube de puntos objetivo  $P$  y la nube

de puntos fuente  $Q$  transformada con la matriz de transformación actual  $T$ .

2. Actualizar la transformación  $T$  minimizando la función objetivo  $E(T)$  definida sobre el conjunto de correspondencias  $\kappa$ , que se presenta en la ecuación 1.

$$E(T) = \sum_{(p,q) \in \kappa} ((\mathbf{p} - \mathbf{T}\mathbf{q}) \cdot n_p)^2 \quad (1)$$

En el proceso, se calculan los residuos y las matrices Jacobianas de la nube de puntos objetivo.

**III-C3b. Registro completo:** A continuación, una vez realizado el procesamiento por pares, se procede a hacerlo para todo el conjunto de nubes de puntos de entrada. Es en este punto donde se realiza el *stacking* de todas las nubes compatibles. Para esto, es necesario estimar la pose de la cámara, y esto se lleva a cabo por medio de una estructura denominada **grafo de poses**.

Como se presenta en la figura 4, un grafo de poses tiene dos elementos clave: nodos y aristas. En este contexto, un nodo es una pieza de geometría (nube de puntos)  $P_i$ , asociada a una matriz de poses  $T_i$  que transforma  $P_i$  en el espacio global. El conjunto  $\{T_i\}$  son las variables desconocidas por optimizar. Inicialmente, se define el espacio global como el espacio de  $P_0$ , y  $T_0$  como la matriz identidad. Las demás matrices de poses se inicializan acumulando transformaciones entre nodos vecinos.

Una arista del grafo de poses conecta dos nodos (piezas de geometría) que se solapan. Cada arista contiene una matriz de transformación  $T_{i,j}$  que alinea la geometría fuente  $P_i$  con la geometría objetivo  $P_j$ . [10]

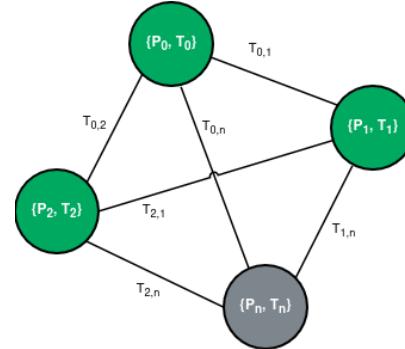


Figura 4. Estructura del grafo de poses.

El error calculado para una arista en el grafo de poses se basa en la ecuación 2, que corresponde al error cuadrático medio (RMSE) entre los conjuntos de datos correspondientes de los dos nodos conectados.

$$RMSE = \sqrt{\sum_{i=1}^n \frac{(P_i - P_j)^2}{n}} \quad (2)$$

**III-C3c. Optimización del grafo de poses:** La optimización global se realiza dos veces en el gráfico de poses. La primera pasada optimiza las poses para el gráfico de poses original, tomando en cuenta todas las aristas y haciendo su

mejor esfuerzo para distinguir las alineaciones falsas entre las aristas inciertas. Estas alineaciones falsas tienen pesos de proceso lineales pequeños y se eliminan después de la primera pasada. La segunda pasada se ejecuta sin ellas y produce una alineación global precisa. Esta técnica se basa en el trabajo presentado por Choi et al. [11]

*III-C4. Visualización de la salida:* Para finalizar el proceso, el producto cuenta con un método que se encarga de generar un archivo *pcd*, en el cual se almacena la información relacionada con la nube de puntos global de salida y despliega un mensaje de éxito en el proceso.

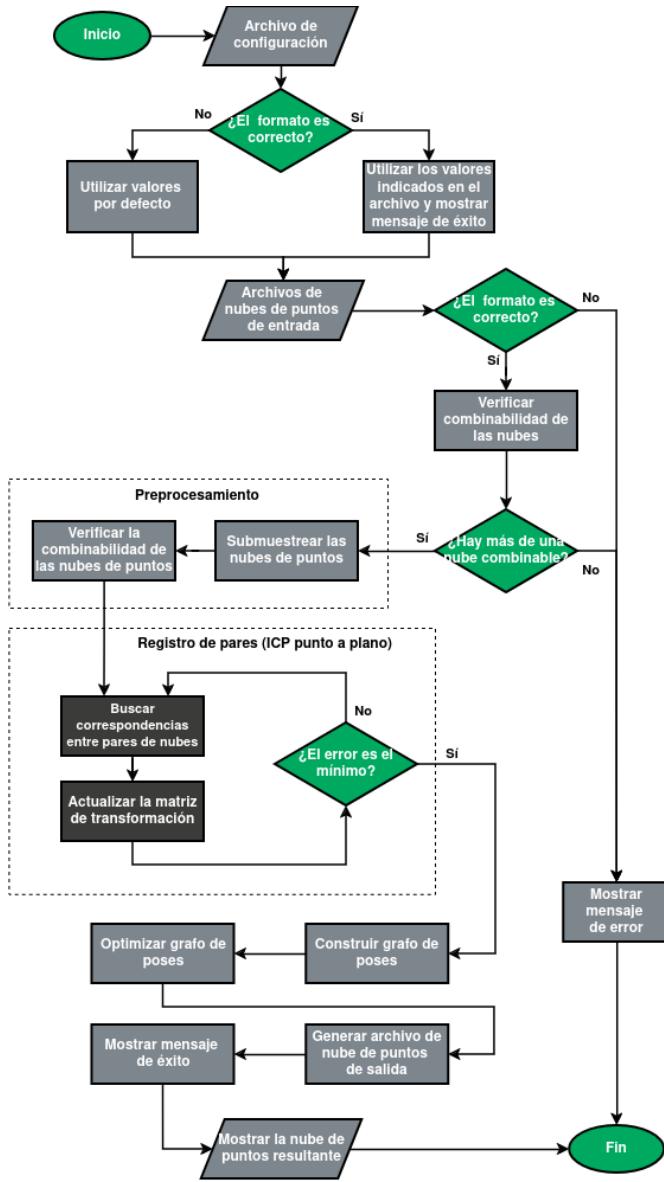


Figura 5. Diagrama de flujo de la perspectiva de algoritmo del producto desarrollado.

#### IV. RESULTADOS

En esta sección se presentan los resultados obtenidos al implementar y evaluar el algoritmo de registro múltiple para la generación de mapas tridimensionales a partir de nubes de

puntos individuales. La evaluación de la solución desarrollada se centra en una comparación visual de las nubes de puntos generadas, contrastando los resultados obtenidos con aquellos presentados en estudios previos y métodos alternativos.

##### IV-A. Resultados parciales y final

A continuación, se presentan los resultados parciales obtenidos del flujo de ejecución del algoritmo. En primer lugar, en la figura 6 se visualiza el archivo de configuración utilizado. Es posible apreciar los valores de parámetros establecidos específicamente para este conjunto de datos.

```

data > { config.json > ...
1  {
2    "input_path": ".../data/7-scenes-redkitchen/test",
3    "config_params": {
4      "voxel_size": 0.05,
5      "remove_outliers_params": {
6        "nb_neighbors": 20,
7        "std_ratio": 2.0
8      },
9      "combinability_threshold": 0.17
10    },
11    "output_file": ".../results/nube_pcd_salida_5.pcd"
12  }
  
```

Figura 6. Ejemplo de archivo de configuración.

En las figuras 7 y 8, es posible visualizar que las nubes de puntos de entrada están desalineadas. La segunda imagen muestra dichas nubes de puntos una vez fueron aplicadas las técnicas de preprocesamiento, de ahí que la densidad de los puntos es menor.

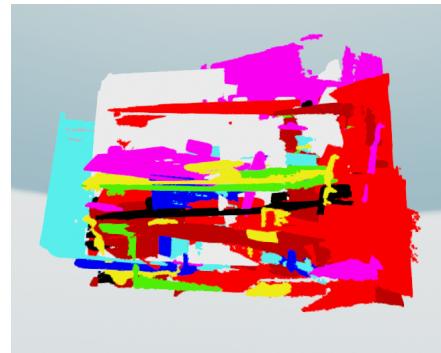


Figura 7. Nubes de puntos sin combinar.

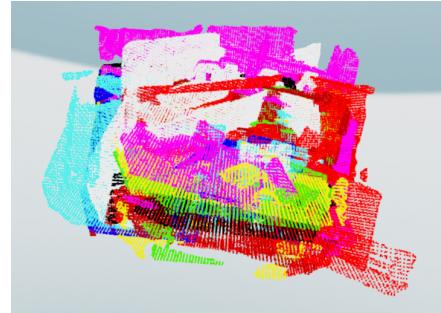


Figura 8. Nubes de puntos preprocesadas según los parámetros del archivo de configuración.

Al aplicar el algoritmo implementado, se genera en un archivo *pcd*, la fusión coherente de las nubes de puntos que fueron seleccionadas como combinables.

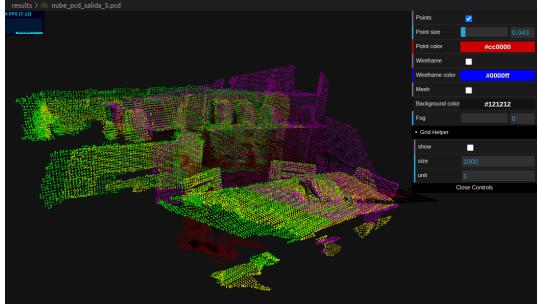


Figura 9. Resultado final de la ejecución del programa.

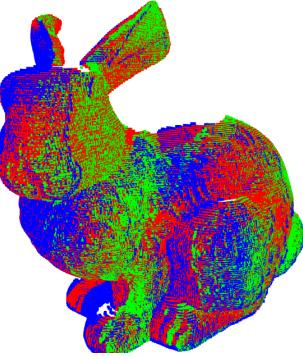


Figura 12. Resultado final de la ejecución del programa con el set de datos de *Stanford Bunny*.

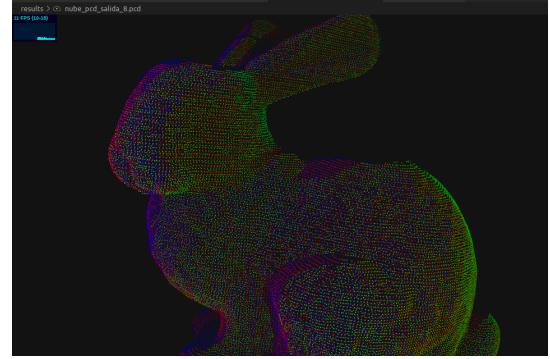


Figura 13. Archivo de salida final de la ejecución del programa con el set de datos de *Stanford Bunny*.

#### IV-B. Comparativa con otros métodos

A continuación, se presenta el proceso para obtener un resultado con el set de datos de *Stanford Bunny*.

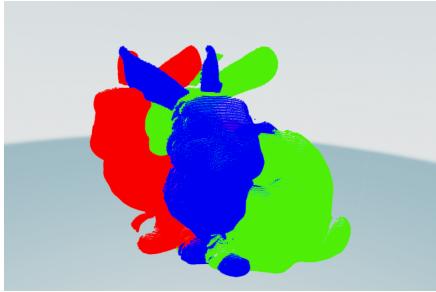


Figura 10. Nubes de puntos del set de datos de *Stanford Bunny* sin combinar.

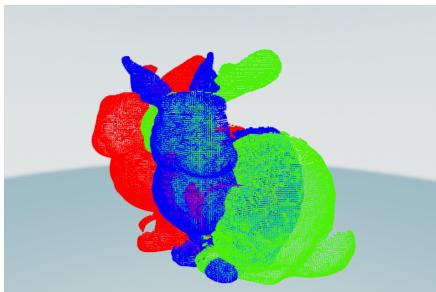


Figura 11. Nubes de puntos preprocesadas según los parámetros del archivo de configuración.

Zhang, Yao y Deng [4] presentan una serie de resultados obtenidos al aplicar otros métodos de registro sobre conjuntos de datos de entrada. Estos pueden ser visualizados en la figura 14.

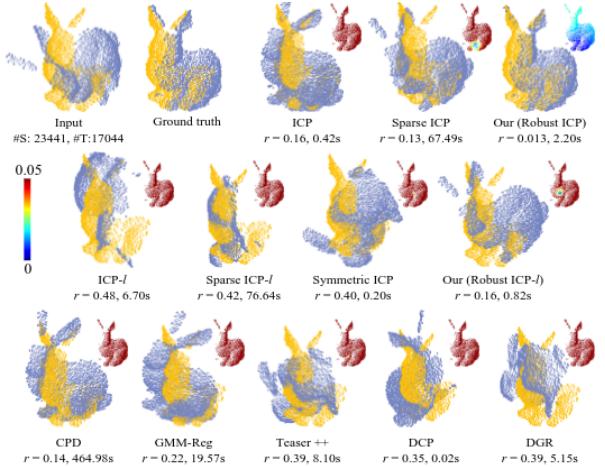


Figura 14. Resultados de otros métodos de combinación de nubes de puntos. [4]

La imagen 14 muestra casos donde la figura del conejo fue reconstruida de manera óptima y otros casos en los que esto no fue posible. Por ejemplo, algunas variantes del ICP no logran una convergencia suficiente para brindar una combinación correcta, mientras que la implementación robusta (arriba a la

derecha de la imagen) sí lo logra. Al realizar una comparación visual de los resultados obtenidos en el presente proyecto contra los otros métodos, es posible analizar la efectividad de la solución desarrollada, ya que se obtiene una fusión de las nubes de puntos coherente, como la obtenida con los métodos robustos.

#### IV-C. Otros resultados exitosos

A continuación, se presentan dos ejemplos de combinaciones exitosas por parte del algoritmo implementado.

*IV-C1. Set de datos de exteriores:* Para conjuntos de datos de exteriores se obtuvo lo siguiente:

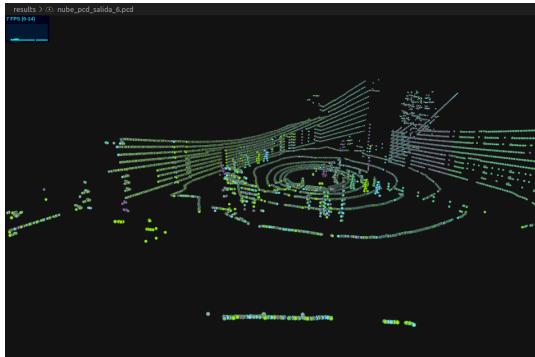


Figura 15. Resultado exitoso con nubes de puntos de exteriores.

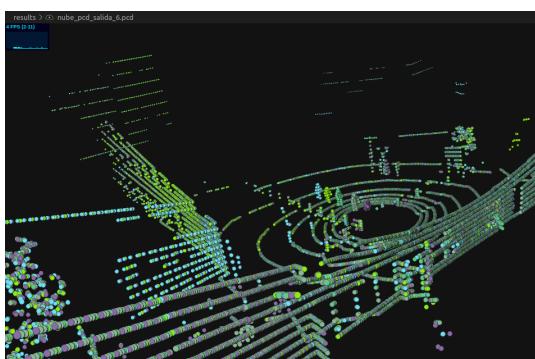


Figura 16. Otra perspectiva del resultado exitoso con nubes de puntos de exteriores.

*IV-C2. Set de datos de interiores:* Para conjuntos de datos de interiores se obtuvo lo mostrado previamente en los resultados, así como lo siguiente:

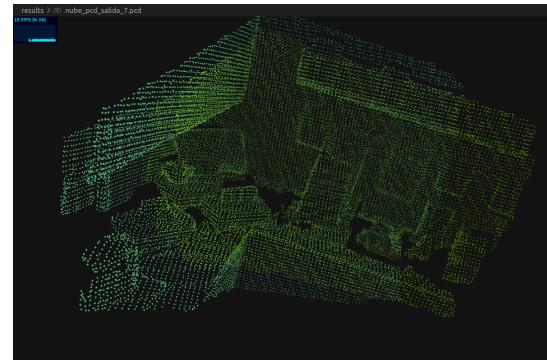


Figura 17. Resultado exitoso con nubes de puntos de una sala de una casa.

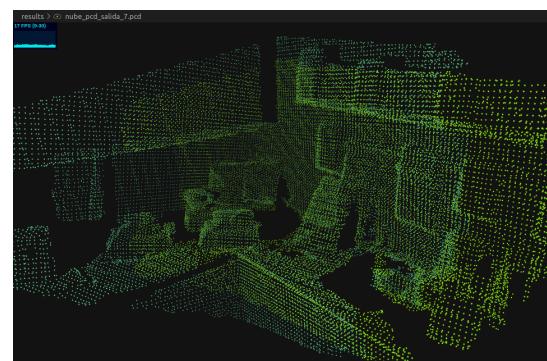


Figura 18. Otra perspectiva del resultado exitoso con nubes de puntos de una sala de una casa.

#### IV-D. Casos no óptimos

Finalmente, se presentan algunos casos en los que el funcionamiento del algoritmo no fue el esperado, ya sea debido a la sensibilidad del mismo respecto a los parámetros de configuración o la naturaleza del set de datos.

En la figura 19, se muestra un cambio mínimo del archivo de configuración que se visualiza en la figura 6. El parámetro cambiado fue el *voxel\_size*, de 0,05 a 0,08.

```
data > {} config.json > ...
1  {
2    "input_path": "../data/7-scenes-redkitchen/test",
3    "config_params": {
4      "voxel_size": 0.08,
5      "remove_outliers_params": {
6        "nb_neighbors": 20,
7        "std_ratio": 2.0
8      },
9      "combinability_threshold": 0.17
10 },
11   "output_file": "../results/nube_pcd_salida_9.pcd"
12 }
```

Figura 19. Archivo de configuración modificado.

Seguidamente, en la figura 20, se presenta el resultado obtenido con dicha configuración. Es posible observar que se presentan algunas nubes desalineadas. Esto demuestra la dependencia y la sensibilidad del algoritmo a los valores de los parámetros establecidos, ya que una leve variación de estos pueden cambiar completamente el resultado.

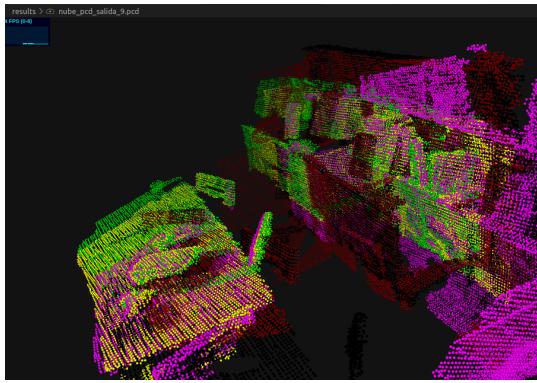


Figura 20. Resultado final no óptimo por las configuraciones utilizadas.

Ahora, en la figura 21, se presenta un caso en el cual se agrega una nube de puntos al set de prueba. Es posible observar que la nube de color amarillo no fue combinada correctamente. Esto pudo ser debido a que su alineación inicial no era suficiente para que el algoritmo fuera capaz de fusionarla con las demás, o los parámetros de configuración no fueron los adecuados para llevar a cabo el proceso de manera correcta.

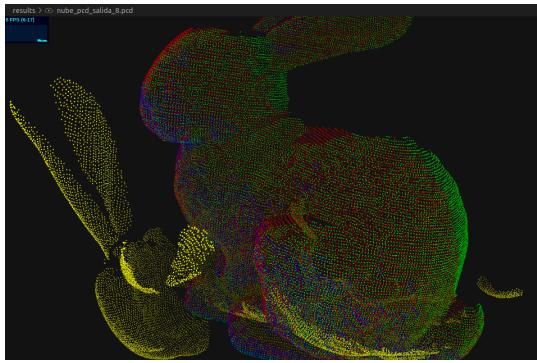


Figura 21. Resultado final no óptimo por las configuraciones utilizadas con el set de datos de *Stanford Bunny*.

#### IV-E. Fortalezas

- Alineación Precisa: La solución desarrollada demostró una capacidad notable para alinear nubes de puntos capturadas desde diferentes ángulos y posiciones. Las transformaciones rígidas calculadas mediante el algoritmo de registro múltiple permiten una integración fluida de los fragmentos, resultando en modelos tridimensionales coherentes y bien definidos. Esta precisión se refleja en la claridad y continuidad de las estructuras representadas.
- Adaptabilidad a conjuntos de datos: Debido a la utilización de un archivo de configuración personalizable, el algoritmo puede ajustarse para manejar diversas condiciones de entrada y requisitos específicos del proyecto. Esto permite a los usuarios adaptar el procesamiento de nubes de puntos a sus necesidades particulares sin necesidad de modificar el código base del programa.

#### IV-F. Debilidades

- Dependencia de la Alineación Inicial: A pesar de las mejoras en la optimización del grafo de poses, la solución

aún muestra cierta dependencia de una buena alineación inicial. En escenarios donde los fragmentos iniciales no están bien alineados, la calidad de la nube de puntos final puede verse comprometida. Esto sugiere la necesidad de mejorar los algoritmos de odometría o incorporar técnicas adicionales para asegurar una alineación inicial más precisa.

- Computacionalmente Intensivo: La implementación de registro múltiple y la optimización del grafo de poses son procesos computacionalmente intensivos. La solución desarrollada requiere una cantidad de recursos de procesamiento y tiempo de cálculo directamente proporcional a la cantidad de nubes de puntos en los conjuntos de datos, lo que puede limitar su aplicabilidad en tiempo real o en dispositivos con capacidades computacionales limitadas.
- Dependencia de valores de configuración: Aunque el archivo de configuración brinda adaptabilidad al producto, la calidad de la nube de puntos de salida depende en gran medida de los valores especificados en el mismo. Esto significa que si los parámetros no están bien ajustados, la precisión y coherencia de la nube de puntos resultante pueden verse seriamente comprometidas. Además, esta dependencia puede resultar en una necesidad constante de ajustes y pruebas adicionales para encontrar la configuración óptima, incrementando así el tiempo y esfuerzo necesario para obtener resultados satisfactorios.

## V. CONCLUSIONES

El presente trabajo ha logrado desarrollar una solución eficaz para la generación de mapas tridimensionales a partir de nubes de puntos individuales, abordando el desafío de la falta de información sobre la pose de la cámara. Se ha demostrado que el algoritmo de *stacking* implementado es capaz de fusionar las nubes de puntos de manera coherente y precisa, lo que permite obtener representaciones tridimensionales comprensibles del entorno mapeado.

Entre los aspectos positivos de la solución se destaca su eficacia en la combinación de datos, la utilización de técnicas de visión por computadora y fotogrametría, y la flexibilidad para trabajar con diversos conjuntos de datos en diferentes ambientes.

Sin embargo, se identifican algunas limitaciones que podrían afectar la aplicación práctica de la solución. Entre ellas, se observa que el proceso de combinación de algunas nubes de puntos puede verse afectado por problemas de alineación inicial o falta de relación con el conjunto de datos, lo que podría generar resultados subóptimos en ciertos escenarios. Asimismo, al realizar pruebas sobre el producto desarrollado, fue posible detectar la dependencia que tiene el algoritmo con los parámetros del archivo de configuración.

## VI. RECOMENDACIONES

En cuanto a las recomendaciones para futuras mejoras y trabajo futuro relacionado con este proyecto, se sugieren las siguientes acciones.

Se recomienda continuar trabajando en la optimización del algoritmo de *stacking* para mejorar su eficacia y precisión. Esto podría implicar la exploración de nuevas técnicas de alineación y fusión de nubes de puntos, así como la implementación de nuevas estrategias para preprocesar las nubes de puntos.

Como trabajo futuro que mejoraría el desempeño del producto, destaca el desarrollo de un modelo de *machine learning* que colabore a distinguir la combinabilidad de las nubes de puntos. Esto ayudaría a los usuarios a reducir su interacción con el archivo de configuración, y así lograr que el ajuste de parámetros se realice de manera automática.

Asimismo, como al utilizar el producto se logra obtener un archivo en formato *pcd* que contiene puntos en tres dimensiones, se recomienda el desarrollo de herramientas de visualización y análisis que permitan a los usuarios explorar y analizar de manera efectiva los mapas tridimensionales generados. Esto podría incluir la implementación de interfaces gráficas de usuario intuitivas, así como herramientas para medir, analizar y manejar las características específicas del entorno mapeado.

Finalmente, para continuar con la validación de la eficacia de la solución desarrollada, se sugiere realizar utilizar el producto y realizar evaluaciones en entornos del mundo real con diferentes combinaciones de parámetros de configuración. Esto podría implicar la colaboración con instituciones o empresas que requieran de la generación de mapas tridimensionales para aplicaciones prácticas, como la agricultura de precisión, la cartografía urbana o la inspección de infraestructuras.

## VII. BIBLIOGRAFÍA

- [1] Lynn. “*Lidar vs photogrammetry: Which is better for point cloud creation?*” [En línea]. Disponible en: <https://www.mosaic51.com/technology/lidar-vs-photogrammetry-which-is-better-for-point-cloud-creation/>. (2021).
- [2] MathWorks. “*What Is Drone Mapping?*” [En línea]. Disponible en: <https://mathworks.com/discovery/drone-mapping.html>. (2024).
- [3] J. Huang, T. Birdal, Z. Gojcic, L. J. Guibas y S. M. Hu, “Multiway Non-rigid Point Cloud Registration via Learned Functional Map Synchronization,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, n.º 2, págs. 2038-2053, 2022. dirección: <https://arxiv.org/pdf/2111.12878.pdf>.
- [4] J. Zhang, Y. Yao y B. Deng, “Fast and Robust Iterative Closest Point,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, n.º 7, págs. 3450-3466, 2021. dirección: <https://arxiv.org/pdf/2007.07627.pdf>.
- [5] Y. Chen y G. Medioni, “Object modelling by registration of multiple range images,” *Image and vision computing*, vol. 10, n.º 3, págs. 145-155, 1992. dirección: <http://graphics.stanford.edu/courses/cs348a-17-winter/Handouts/chen-medioni-align-rob91.pdf>.
- [6] D. Chetverikov, D. Svirko, D. Stepanov y P. Krsek, “The trimmed iterative closest point algorithm,” en *2002 International Conference on Pattern Recognition*, IEEE, vol. 3, 2002, págs. 545-548.
- [7] X. Blanch, A. Abellán y M. Guinau, “Point cloud stacking: A workflow to enhance 3D monitoring capabilities using time-lapse cameras,” *Remote Sensing*, vol. 12, n.º 8, pág. 1240, 2020. dirección: <https://www.mdpi.com/2072-4292/12/8/1240>.
- [8] P. J. Besl y N. D. McKay, “A Method for Registration of 3-D Shapes,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, n.º 2, págs. 239-256, 1992. dirección: <https://ieeexplore.ieee.org/document/121791>.
- [9] Open3D, *ICP Registration — Open3D latest documentation*, [En línea]. Disponible en: [https://www.open3d.org/docs/latest/tutorial/Basic/icp\\_registration.html](https://www.open3d.org/docs/latest/tutorial/Basic/icp_registration.html), 2024.
- [10] Open3D, *Multiway Registration — Open3D latest documentation*, [En línea]. Disponible en: [https://www.open3d.org/docs/latest/tutorial/Advanced/multiway\\_registration.html](https://www.open3d.org/docs/latest/tutorial/Advanced/multiway_registration.html), 2024.
- [11] S. Choi, Q.-Y. Zhou y V. Koltun, “Robust reconstruction of indoor scenes,” en *Proceedings of the IEEE conference on computer vision and pattern recognition*, [En línea]. Disponible en: [https://www.cv-foundation.org/openaccess/content\\_cvpr\\_2015/papers/Choi\\_Robust\\_Reconstruction\\_of\\_2015\\_CVPR\\_paper.pdf](https://www.cv-foundation.org/openaccess/content_cvpr_2015/papers/Choi_Robust_Reconstruction_of_2015_CVPR_paper.pdf), 2015, págs. 5556-5565.