



Universidad Autónoma de Querétaro  
Facultad de Ingeniería.

**“MAPEO 3D A PARTIR DE IMÁGENES RGB-D POR MEDIO  
DE UNA ARQUITECTURA CNN PARA UN SISTEMA SLAM”.**

**TESIS**

Que como parte de los requisitos para obtener el grado de  
Maestro en Ciencias en Inteligencia Artificial.

Presenta:

Víctor Beltrán Barrera.

Dirigido por:

Dr. Jesús Carlos Pedraza.

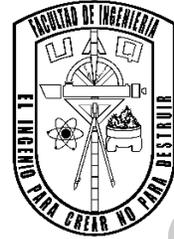
Querétaro, Qro. a 06 de septiembre del 2021.



Universidad Autónoma de Querétaro

Facultad de Ingeniería.

Maestría en Ciencias en Inteligencia Artificial.



**“MAPEO 3D A PARTIR DE IMÁGENES RGB-D POR MEDIO  
DE UNA ARQUITECTURA CNN PARA UN SISTEMA SLAM”.**

**TESIS**

Que como parte de los requisitos para obtener el grado de  
Maestro en Ciencias en Inteligencia Artificial.

**Presenta:**

Víctor Beltrán Barrera.

**Dirigido por:**

Dr. Jesús Carlos Pedraza Ortega.

Dr. Jesús Carlos Pedraza Ortega  
Presidente

\_\_\_\_\_  
Firma

Dr. Marco Antonio Aceves Fernández  
Secretario

\_\_\_\_\_  
Firma

Dr. Saúl Tovar Arriaga  
Vocal

\_\_\_\_\_  
Firma

Dr. Juan Manuel Ramos Arreguín  
Suplente

\_\_\_\_\_  
Firma

Dr. Efrén Gorrostieta Hurtado  
Suplente

\_\_\_\_\_  
Firma

Centro Universitario  
Querétaro, QRO.  
Septiembre 2021  
México

*El presente es de ustedes, pero el futuro, por el que tanto he trabajado, me pertenece.*

*Nikola Tesla*

Dirección General de Bibliotecas UAQ

## RESUMEN

La reconstrucción 3D se ha convertido en uno de los problemas centrales en diversos campos de investigación como lo son visión por computadora, robótica, gráficos por computadora, procesamiento digital de imágenes, entre otros. La principal problemática de estudio en este proceso es la alineación consistente de nubes de puntos superpuestas, mejor conocida como registro de nubes de puntos. La utilización de algoritmos basados en métodos tradicionales es muy popular para realizar el registro de nubes de puntos debido a la simplicidad y fácil aplicación que presentan.

Sin embargo, todos estos algoritmos presentan diversas problemáticas como la susceptibilidad a mínimos locales, sensibilidad a una buena inicialización, tiempos de convergencia elevado, sensibilidad a valores atípicos y ruido. Esto ha llevado a desarrollar nuevos métodos para optimizar el registro de nubes de puntos. La incorporación de métodos basados en aprendizaje profundo ha tomado gran popularidad en diversos campos de investigación debido al alto rendimiento y robustez de estos modelos.

En esta tesis se presenta un sistema de reconstrucción 3D basado en algoritmos de aprendizaje profundo para escenas interiores. Las nubes de puntos son adquiridas mediante un sensor de profundidad. A estas nubes de puntos se les realiza un preprocesamiento para reducir el número de puntos contenido en cada conjunto y son normalizadas dentro de una caja unitaria. El proceso del registro de las nubes de puntos es llevado a cabo mediante una arquitectura basada en redes neuronales y el uso del método tradicional ICP punto a plano como refinamiento.

Los resultados demuestran que el modelo propuesto obtiene mejores resultados en términos de exactitud (*RMSE*) y tiempo de procesamiento que los modelos basados en métodos tradicionales, mejorando de esta manera el mapeo 3D de una escena interior.

**Palabras clave: Reconstrucción 3D, redes neuronales, registro de nubes de puntos, gráfico de pose, ICP.**

## SUMMARY

3D reconstruction has become one of the central problems in various fields of research such as computer vision, robotics, computer graphics, digital image processing, among others. The main study problem in this process is the consistent alignment of overlapping point clouds, better known as point cloud registration. The use of traditional-based methods algorithms is very popular to carry out point cloud registration due to their simplicity and easy application.

However, all these algorithms present various problems such as susceptibility to local minima, sensitivity to good initialization, high convergence times, sensitivity to outliers and noise. This had led to the development of new methods to optimize point cloud registration. Deep learning-based methods incorporation has become very popular in many research fields due the high performance and robustness of these models.

This thesis presents a deep learning-based 3D reconstruction system for indoor scenes. Point clouds are acquired by a depth sensor. These point clouds are pre-processed to reduce point number contained in each set and are normalized into a unit box. Point cloud registration process is carried out through a neural network-based architecture and traditional point-to-plane ICP method as refinement.

Results show that the proposed model obtains better results in terms of accuracy (*RMSE*) and processing time than models based on traditional methods, thus improving 3D mapping of an indoor scene.

**Keywords:** 3D reconstruction, neural network, point cloud registration, pose graph, ICP.

*A mi familia y todas las personas que siempre me han apoyado*

Dirección General de Bibliotecas UAQ

## AGRADECIMIENTOS

A la Universidad Autónoma de Querétaro y al CONACYT por darme la oportunidad de completar la maestría en ciencias en inteligencia artificial.

A toda mi familia, en especial a mi mamá, Josefina Barrera Vázquez, no estaría aquí de no ser por ti.

A amigos por brindarme tanto apoyo y darme consejos cuando más los necesitaba, hicieron este viaje más liviano.

A mis compañeros de generación por su disposición a ayudarnos mutuamente siempre que se complicaban las cosas y por los buenos momentos que compartimos.

A todos mis profesores por guiarme y orientarme a lo largo de este proceso, gracias por transmitirnos todo su conocimiento y el tiempo que han dedicado a nuestra preparación y a las generaciones futuras. A mi profesor y tutor, Dr. Marco Antonio Aceves Fernández, gracias por sus consejos y su dedicación como investigador y docente. Esto no sería posible sin ustedes.

Especialmente a mi profesor y director de tesis, Dr. Jesús Carlos Pedraza Ortega. Gracias por acoplarse a mi forma de trabajar y por confiar en lo que hacía, fue mi única y mejor opción para realizar este trabajo. Por su esfuerzo como docente e investigador, siempre tuvo tiempo para resolver nuestras dudas, aconsejarnos y orientarnos en nuestra preparación además de inculcar siempre el respeto y lo importante que es el trabajo en equipo.

*“... también hay que buscar divertirnos en nuestro trabajo”*

*Dr. Jesús Carlos Pedraza Ortega*

# Índice general

Resumen .....	III
Summary .....	IV
1. Introducción .....	1
1.1. Antecedentes .....	3
1.1.1. Métodos tradicionales .....	3
1.1.2. Métodos basados en aprendizaje profundo .....	5
1.2. Planteamiento del problema .....	8
1.3. Justificación .....	9
1.4. Hipótesis .....	10
1.5. Objetivos .....	10
1.5.1. Objetivo general .....	10
1.5.2. Objetivos específicos .....	10
2. Marco teórico .....	11
2.1. Inteligencia artificial .....	11
2.1.1. Redes neuronales artificiales .....	12
2.2. Reconstrucción 3D .....	15
2.3. Adquisición y preprocesamiento .....	16
2.3.1. Datos de entrada .....	16
2.3.2. Tipos de sensores para la adquisición de datos .....	17
2.3.3. Preprocesamiento de los datos .....	18
2.4. Registro de las nubes de puntos .....	18
2.4.1. Registro por pares .....	19
2.4.1.1. Métodos basados en distancia .....	20
2.4.1.2. Métodos basados en filtros .....	20
2.4.1.3. Métodos basados en probabilidad .....	21
2.4.2. Registro por grupos .....	21
2.5. Algoritmos para el registro de nubes de puntos .....	22
2.5.1. Métodos tradicionales .....	22
2.5.2. Métodos basados en aprendizaje profundo .....	23
3. Metodología .....	25

3.1. Sistema de reconstrucción 3D basado en métodos tradicionales . . . . .	25
3.1.1. Adquisición y preprocesamiento de los datos . . . . .	26
3.1.2. Registro de las nubes de puntos . . . . .	28
3.1.2.1. Prealineación (Fast Global Registration) . . . . .	28
3.1.2.2. Transformación afín (ICP punto a plano) . . . . .	29
3.1.2.3. Optimización del gráfico de pose . . . . .	30
3.2. Sistema de reconstrucción 3D basado en aprendizaje profundo . . . . .	33
3.2.1. Fase de entrenamiento . . . . .	35
3.2.1.1. Base de datos . . . . .	35
3.2.1.2. Arquitectura de la red neuronal . . . . .	37
3.2.2. Fase de reconstrucción . . . . .	38
3.2.2.1. Adquisición y preprocesamiento de los datos . . . . .	38
3.2.2.2. Algoritmo inteligente . . . . .	39
4. Resultados . . . . .	40
4.1. Sistema de reconstrucción basado en métodos tradicionales . . . . .	41
4.1.1. Adquisición y preprocesamiento . . . . .	42
4.1.2. Métodos tradicionales . . . . .	43
4.1.3. Inicialización y gráfico de pose . . . . .	44
4.1.4. Reconstrucción 3D . . . . .	45
4.2. Sistema de reconstrucción basado en aprendizaje profundo . . . . .	48
4.2.1. Fase de entrenamiento . . . . .	48
4.2.2. Refinamiento de la red neuronal . . . . .	50
4.2.3. Preprocesamiento de los datos reales . . . . .	56
4.2.4. Reconstrucción 3D . . . . .	60
4.3. Requerimientos para la implementación en un sistema SLAM . . . . .	63
5. Conclusiones y trabajo futuro . . . . .	66
6. Referencias . . . . .	70
7. Anexos . . . . .	75
7.1. Artículo publicado . . . . .	75
7.2. Constancia de lengua extranjera . . . . .	76

## Índice de figuras

2.1. Inteligencia Artificial y sus principales áreas (Goodfellow & Bengio, 2017) . . . . .	12
2.2. Esquema de una neurona artificial (Izaurieta & Saavedra, 2015) . . . . .	13
2.3. Función escalón (Izq.), función sigmoidea (Der.), (Izaurieta & Saavedra, 2015) . . . . .	13
2.4. Red neuronal unicapa (Izaurieta & Saavedra, 2015) . . . . .	14
2.5. Red neuronal multicapa (Izaurieta & Saavedra, 2015) . . . . .	15
2.6. Proceso de Reconstrucción 3D. (Intwala & Magikar, 2016) . . . . .	15
2.7. Nubes de puntos de una escena tomadas de distintos ángulos . . . . .	16
2.8. Ejemplo de una nube de puntos desde una vista latera . . . . .	16
2.9. Taxonomía de las principales técnicas utilizadas para la adquisición de imágenes en 3D (Pedraza et al. 2011) . . . . .	17
2.10. Sensores RGB-D visuales más populares en el mercado . . . . .	17
2.11. Registro de nubes de puntos por pares con búsqueda de correspondencias (Zhu et al., 2019) . . . . .	19
2.12. Esquema general del proceso de registro de nubes de puntos . . . . .	22
2.13. Métodos basados en aprendizaje profundo (Zhang et al., 2020) . . . . .	24
3.1. Modelo del sistema de reconstrucción 3D . . . . .	26
3.2. Sensor de profundidad Intel RealSense D435 ( <a href="https://www.intelrealsense.com/depth-camera-d435">https://www.intelrealsense.com/depth-camera-d435</a> ) . . . . .	26
3.3. Montaje de la cámara de profundidad para el muestreo de las nubes de puntos . . . . .	28
3.4. Representación de un gráfico de pose (Stachniss, 2020) . . . . .	31
3.5. Modelo del sistema de reconstrucción 3D basado en aprendizaje profundo . . . . .	34
3.6. Objeto extraído de la base de datos ModelNet40 . . . . .	35
3.7. Nube de puntos obtenida de ModelNet40 . . . . .	36
3.8. Nube de puntos normalizada . . . . .	36
3.9. Nube de puntos objetivo y fuente . . . . .	37
3.10. Arquitectura de la red neuronal PCRNNet (Sarode et al., 2019) . . . . .	38
3.11. Diagrama del algoritmo inteligente . . . . .	39
4.1. Nubes de puntos obtenidas con el sensor de profundidad . . . . .	40
4.2. Nubes de puntos capturadas con el sensor de profundidad con diferente ángulo y distancia . . . . .	41
4.3. Nubes de puntos de entrada . . . . .	41

4.4. Muestreo de nubes de puntos . . . . .	42
4.5. Alineación obtenida para dos nubes de puntos . . . . .	43
4.6. Proceso de registro de las nubes de puntos . . . . .	45
4.7. Reconstrucción 3D con métodos tradicionales, escena 1 . . . . .	46
4.8. Reconstrucción 3D con métodos tradicionales, escena 2 . . . . .	46
4.9. Nivel de detalles en el sistema de reconstrucción 3D . . . . .	47
4.10. Resultados obtenidos por el modelo pre entrenado en (Sarode et al., 2019) . . . . .	49
4.11. Resultados obtenidos por el modelo entrenado . . . . .	49
4.12. Resultado del registro de puntos obtenido por ambos modelos . . . . .	50
4.13. Resultados del registro de nubes de puntos en objeto 1 . . . . .	51
4.14. Resultados del registro de nubes de puntos en objeto 2 . . . . .	52
4.15. Resultados del registro de nubes de puntos en objeto 3 . . . . .	52
4.16. Resultados del registro de nubes de puntos en objeto 4 . . . . .	53
4.17. Resultados del registro de nubes de puntos en objeto 5 . . . . .	53
4.18. Resultados cuantitativos totales obtenidos por cada algoritmo . . . . .	54
4.19. Resultados del valor RMSE totales obtenidos por cada algoritmo . . . . .	54
4.20. Resultados del valor FITNESS SCORE totales obtenidos por cada algoritmo . . . . .	55
4.21. Resultados del valor RMSE obtenido por cada algoritmo en 100 objetos . . . . .	55
4.22. Resultados del valor FITNESS SCORE obtenido por cada algoritmo en 100 objetos . . . . .	56
4.23. Nube de puntos original (Izq.), nube de puntos con down sampling (Der.) . . . . .	57
4.24. Vista con zoom de la parte frontal (Izq.) y angular (Der.) de la nube de puntos con down sampling . . . . .	57
4.25. Nube de puntos convertida en arreglo (Der.) y nube de puntos normalizada (Izq.) . . . . .	58
4.26. Forma del arreglo de la nube de puntos ( $n, d$ ) . . . . .	58
4.27. Forma del tensor de la nube de puntos ( $[n, d]$ ) . . . . .	58
4.28. Creación del conjunto de pruebas con forma $[T([b, n, d]), S([b, n, d])]$ . . . . .	59
4.29. Proceso del sistema de reconstrucción 3D basado en algoritmos de aprendizaje profundo . . . . .	61
4.30. Reconstrucción 3D con métodos de aprendizaje, escena 1 . . . . .	61
4.31. Comparación de los resultados cualitativos entre ambos sistemas de reconstrucción 3D . . . . .	62
4.32. Modelo del sistema SLAM visual . . . . .	64

## Índice de tablas

1.1. Algoritmos para el registro de nubes de puntos basados en métodos tradicionales . . . .5	
1.2. Algoritmos para el registro de nubes de puntos basados en aprendizaje profundo . . . .8	
2.1. Características de algunos sensores RGB-D visuales encontrados en el mercado . . . .18	
3.1. Especificaciones cámara Intel RealSense D435 . . . . .27	
4.1. Resultados obtenidos en el registro de nubes de puntos . . . . .44	
4.2. Resultados obtenidos registro de puntos con optimización global . . . . .45	
4.3. Resultados finales obtenidos por el sistema de reconstrucción 3D . . . . .47	
4.4. Comparación de resultados del modelo pre entrenado y el modelo entrenado . . . . .49	
4.5. Comparación de resultados objeto 1 . . . . .51	
4.6. Comparación de resultados objeto 2 . . . . .52	
4.7. Comparación de resultados objeto 3 . . . . .52	
4.8. Comparación de resultados objeto 4 . . . . .53	
4.9. Comparación de resultados objeto 5 . . . . .53	
4.10. Comparación de los resultados finales obtenidos . . . . .62	

# 1. INTRODUCCIÓN

Hoy por hoy, la reconstrucción 3D abarca diversas aplicaciones en distintas áreas. En visión por computadora se han desarrollado sistemas de realidad aumentada utilizando este proceso (Wu et al., 2019). En la robótica ha permitido desarrollar grandes avances en sistemas de localización y mapeo simultaneo (SLAM) (Durrant-Whyte & Bailey, 2006), así como en vehículos de exploración y rescate (Chen, Ihara & Hasegawa, 2020). En el sector automotriz su mayor auge se está alcanzando con los vehículos autónomos utilizando sensores LiDAR para obtener los datos de profundidad y realizar un mapeo de los alrededores (Lee, Song & Jo, 2016).

La principal problemática dentro del proceso de reconstrucción 3D es el registro de nubes de puntos, cuyo papel es fundamental dentro del proceso de reconstrucción ya que una buena alineación permitirá obtener un buen efecto (Furukawa & Hernández, 2013). El objetivo del registro de las nubes de puntos es encontrar correspondencias entre dos o más conjuntos de puntos y obtener una matriz de rotación junto con un vector de traslación de tal manera que los puntos se vayan alineando unos con otros.

El registro de nubes de puntos se divide en dos categorías: registro por pares y registro por grupos. En la primera categoría se toman dos nubes de puntos únicamente durante el proceso mientras que en la segunda se pueden tomar dos o más nubes de puntos.

Dentro del proceso de reconstrucción 3D el registro de nubes de puntos está conformado generalmente por dos fases: la pre-alineación y la transformación afín. La pre-alineación tiene dos enfoques: a nivel local utiliza descriptores como los histogramas de características de puntos (PFH) (Rusu et al., 2008) para codificar la variación de la forma en el vecindario de puntos y a nivel global toman todos los puntos en una sola cuenta (Tao & Hasegawa, 2015). Su principal objetivo consiste en realizar la pre alineación de las nubes de puntos generando una transformación inicial para la fase de la transformación afín, donde se encuentra la transformación de cuerpo rígido que mejor alinee ambas nubes de puntos.

Los métodos para resolver el problema del registro de nubes de puntos pueden ser clasificados dentro de dos grupos: los métodos tradicionales y los métodos basados en aprendizaje profundo. Cada grupo se clasifica en dos categorías: Los métodos locales o

métodos basados en correspondencias, y los métodos globales o métodos libres de correspondencias.

La familia de algoritmos conocidos como métodos locales requieren una alineación aproximada como inicialización. Debido a esto, uno de sus principales problemas es que pueden llegar a estancarse en un mínimo local (Deng et al, 2019). Por el lado contrario, los métodos globales no requieren una inicialización ya que se basan en extraer las características geométricas de las nubes de puntos. Es por esto que los métodos globales son utilizados generalmente como inicialización para los métodos locales con el fin de obtener mejores resultados (Zhou, Park & Koltun, 2016).

En la actualidad, el registro de nubes de puntos basado en aprendizaje profundo ha tomado gran popularidad debido al alto rendimiento y la robustez que presentan estos modelos ante problemas que afectan de manera significativa a los métodos tradicionales como el ruido, valores atípicos, tiempos de convergencia elevados, susceptibilidad a mínimos locales, sensibilidad a una buena inicialización, entre otros.

En esta tesis se presenta un sistema de reconstrucción 3D basado en algoritmos de aprendizaje profundo para escenas interiores. El modelo está constituido por tres secciones. En la primera sección se realiza la adquisición y el preprocesamiento de los datos. Por medio de un sensor de profundidad se obtienen las nubes de puntos que constituyen la escena a reconstruir, se realiza un muestreo para reducir el número de puntos contenido en cada conjunto y se normalizan dentro de una caja unitaria. En la segunda sección se realiza el registro de las nubes de puntos. Este proceso es llevado a cabo por el algoritmo inteligente propuesto. Este algoritmo se divide en dos partes: la inicialización inteligente; donde por medio de una arquitectura de redes neuronales (Neural Networks, NN) se obtiene la inicialización de la transformación de cuerpo rígido, y el refinamiento iterativo; donde se introduce el algoritmo Iterative Closest Point (ICP) punto a plano como refinamiento en el resultado preliminar de la arquitectura NN. Finalmente en la tercera sección se realiza el mapeo 3D aplicando la transformación de cuerpo rígido a la nube de puntos original. Se implementaron y evaluaron diferentes algoritmos para el registro de nubes de puntos como comparativa al modelo propuesto usando métricas como la raíz del error cuadrático medio, área de superposición y el tiempo de procesamiento.

## **1.1 ANTECEDENTES.**

El registro de nubes de puntos tridimensionales tiene como objetivo estimar una transformación rígida entre varias nubes de puntos que se superponen parcialmente para asociar conjuntos de datos en un sistema de coordenadas común. Esto permite integrar datos de múltiples sensores y puntos de vista en un modelo más grande, como un mapa de nube de puntos de alta precisión y una malla completa de objetos (Zhang, Dai & Sun, 2020).

Este proceso ha sido bien estudiado a lo largo de los años y se han desarrollado numerosos métodos para resolverlo. En esta sección se presenta una breve descripción de los dos grupos de métodos para el registro de nubes de puntos: los métodos tradicionales y los métodos basados en aprendizaje profundo.

### **1.1.1 Métodos tradicionales.**

Desde su creación el algoritmo ICP ha sido ampliamente utilizado para realizar la tarea de registro de conjuntos de puntos. En (Besl & McKay, 1992) cada uno de los puntos dentro de un conjunto son emparejados con el punto más cercano del otro conjunto. La métrica del error se basa en la minimización de la distancia euclidiana entre cada uno de estos pares de puntos. Este proceso se repite hasta alcanzar el criterio de término que puede ser por número de iteraciones o cierta magnitud de error alcanzada lo cual hace que el tiempo de procesamiento sea elevado.

Por otro lado, en (Chen & Medioni, 1991) se utiliza una métrica de error basada en minimizar la distancia euclidiana entre un punto y un plano tangente correspondiente a este punto por medio de métodos de mínimos cuadrados no lineales como en (Wu et al., 2015). Aunque el tiempo de convergencia del algoritmo ICP punto a plano se incrementa un poco con respecto a ICP punto a punto en cada iteración se logran obtener mejores resultados y una convergencia más rápida.

Para enfrentar los principales problemas que sufre este algoritmo ante factores como puntos sin correspondencias, poca superposición entre ambos conjuntos de puntos, el pre-

alineamiento y la susceptibilidad a converger hacia un mínimo local han sido propuestos diferentes métodos y variantes de ICP.

En (Agamennoni, Fontana & Sorrenti, 2016) se hace uso de técnicas probabilísticas en el emparejamiento de puntos para reducir los puntos sin correspondencias, (Liu et al., 2018) propone un método basado en asignar pesos a los puntos con menor distancia euclidiana para reducir el error de medida en puntos sin correspondencias. En (Seal & Bhowmick, 2017) se presenta una variante de ICP junto con el algoritmo RANSAC para remover puntos sin correspondencias y el uso de técnicas de aceleración de búsqueda como *kd-Tree* (Miller, Vandome & Mcbrewster, 2009). En (Wu et al., 2015) se propone el uso de mínimos cuadrados no lineales junto con ICP para conjuntos de puntos con bajo nivel de superposición.

Debido a la necesidad de una buena inicialización, se han implementado técnicas como el uso de descriptores (Rusu, Blodow & Beetz, 2009) para codificar la variación de la forma en los conjuntos de puntos, y encontrar un movimiento inicial adecuado por medio de RANSAC (Chen, Hung & Cheng, 1999), o su variante (Zhou, Park & Koltun, 2016), cuya principal ventaja es que no necesitan un alineamiento inicial, sin embargo, el tiempo de procesamiento se incrementa.

Para evitar caer en mínimos locales (Tao & Hasegawa, 2015) propone un modelo global con baja complejidad computacional basado en ICP utilizando árboles dimensionales (*kd-Tree*) para búsqueda de vecinos cercanos y los vectores normales para realizar la rotación. Otras variantes a nivel global como (Koguciuk, 2017) emplean métodos heurísticos. Go-ICP (Yang et al., 2016) obtiene la posición global óptima basada en una estrategia de *Branch and bound* para encontrar la transformada rígida, sin embargo, tiene un tiempo de convergencia muy alto debido a la técnica utilizada.

El uso de optimizadores para resolver problemas de mínimos cuadrados no lineales es aplicado como refinamiento en el registro de conjuntos de puntos a nivel global como Levenberg Marquardt (LM) y Gauss Newton (GN). Este tipo de métodos en conjunto con métodos de registro a nivel local hacen una buena combinación para obtener mejores resultados.

Una de las técnicas que facilita el proceso de reconstrucción son los gráficos de pose ya que conducen a una estructura fácil de procesar basada en nodos y bordes además de ofrecer una mayor flexibilidad en términos de observación ante la deformación de las nubes de puntos al utilizar sensores RGB-D o LiDAR para la adquisición de datos. En (Yan et al., 2019) se presenta un sistema de SLAM con un sensor LiDAR basado en gráficos de pose, en este trabajo se utiliza el algoritmo ICP punto a plano para el registro de puntos y se optimiza a nivel global por medio del algoritmo Levenberg Marquardt.

**Tabla 1.1.** Algoritmos para el registro de nubes de puntos basados en métodos tradicionales.

<b>Autores</b>	<b>Algoritmo</b>	<b>Método</b>	<b>Año</b>	<b>Características</b>
Chen & Mendioni	ICP punto – plano	Local / Distancia	1991	Toma la distancia entre un punto y un plano / sensible a inicialización y mínimos locales
Besl & McKay	ICP punto – punto	Local / Distancia	1992	Toma la distancia entre dos puntos / sensible a inicialización y mínimos locales
Chen, et al.	RANSAC	Global / Distancia	1999	Emparejamiento parcial utilizando puntos preseleccionados para optimizar el proceso / número elevado de iteraciones para resultados óptimos - tiempo de procesamiento elevado
Zhou, et al.	FGR	Global / Distancia	2016	Toma un conjunto de correspondencias pequeño para optimizar el tiempo de convergencia / requiere cierto número de correspondencias – sensible a puntos fuera de línea.
Yan, et al.	Registro fino global	Local-Global / Distancia	2019	Utiliza ICP punto a plano junto con métodos de optimización globales / sensible a inicialización.

### 1.1.2 Métodos basados en aprendizaje profundo.

En los últimos años se ha desarrollado una amplia clase de arquitecturas profundas para datos geométricos, denominadas arquitecturas de aprendizaje profundo geométrico, que incluye métodos recientes de aprendizaje en gráficos y nubes de puntos (Wang & Solomon, 2019).

El primer modelo en abrir las puertas dentro del aprendizaje profundo a las nubes de puntos fue PointNet (Qi et al., 2017). Esta arquitectura ha servido de inspiración a varios trabajos dentro del estado del arte dedicados a realizar tareas como la segmentación y la clasificación de objetos en nubes de puntos principalmente.

En años recientes se ha incrementado el desarrollo de algoritmos para el registro de nubes de puntos basados en este trabajo. Estos algoritmos pueden ser clasificados dentro de dos categorías: los métodos basados en correspondencias, también llamados métodos locales, y los métodos libres de correspondencias o métodos globales.

Los métodos locales se basan en obtener correspondencias entre las dos nubes de puntos de entradas, son los métodos más comunes dentro del campo del registro de nubes de puntos y gran parte de ellos están basados en los métodos tradicionales.

A diferencia de los métodos globales que extraen características por nube de puntos, los métodos locales se basan en extraer características por puntos para generar un mapeo y extraer la transformación rígida.

Debido a esto, PointNet es raramente usada dentro de esta categoría ya que se centra en obtener las características globales. Una versión mejorada es PointNet++ (Qi et al., 2017) capaz de extraer características locales dentro de una nube de puntos.

Uno de los trabajos más representativos dentro de este tipo de métodos es la red neuronal convolucional Virtual Corresponding Points (DeepVCP) (Lu et al., 2019) el cual aplica PointNet++ para la extracción de un descriptor de características, selecciona los puntos clave dentro de la nube de puntos que son más significantes para el registro por medio de una capa de asignación de pesos y por medio de una modificación de PointNet++ llamada Mini-PointNet (Qi et al., 2017) incrementa la información de las características locales para después utilizar SVD (*Singular Value Decomposition*) y obtener la transformada de cuerpo rígido.

La red neuronal convolucional Point Pair Feature Network (PPFNet) (Deng, Birdal & Ilic, 2018) toma parte de la estructura de PointNet y se centra en los descriptores invariantes de rotación dentro de las correspondencias, por medio de PointNet obtiene las características

globales y las concatena con las características locales obtenidas por medio del módulo mini-PointNet.

Como alternativa a PointNet, DGCNN (Wang et al., 2019) recupera explícitamente la estructura gráfica tanto en el espacio euclidiano como en el espacio de características y aplica redes neuronales gráficas al resultado. PCNN (Atzmon et al., 2018) usa un operador de extensión para definir la convolución en las nubes de puntos, mientras que PointCNN (Le et al., 2018) aplica la convolución euclidiana después de aplicar una transformación aprendida.

DCP (*Deep Closest Point*) (Wang et al., 2019) consta de tres partes: una red de incrustación de nube de puntos, un módulo basado en la atención combinado con una capa de generación de puntero, para aproximar el emparejamiento combinatorio, y una capa de descomposición de valor singular diferenciable (SVD) para extraer la transformación rígida final.

Los métodos globales son menos habituales dentro del campo del registro de nubes de puntos. Se basan en obtener el movimiento rígido o transformada de cuerpo rígido mediante la diferencia de las características globales de las nubes de puntos de entrada.

PointNetLK (Aoki et al., 2019) se obtienen las características globales de las nubes de puntos de entrada mediante la red de PointNet y mediante una modificación del algoritmo de Lucas & Kanade (LK) (Baker & Matthews, 2004) obtienen el Jacobiano de la nube de puntos modelos y la transformación rígida en un proceso repetitivo.

Feature-metric registration (Huang, Mei & Zhang, 2020) realiza un proceso similar al de PointNetLK pero incrementa la optimización del proceso mediante la disminución en las métricas de las características, utiliza un sistema de *encoder-decoder* para extraer las características. El codificador genera características diferentes y el decodificador se encarga de recuperar dichas características para cada una de las correspondientes copias previamente rotadas bajo un sistema de entrenamiento semi supervisado.

Al igual que los métodos anteriores, PCRNet (Sarode et al., 2019) extrae las características globales de las nubes de puntos utilizando PointNet, las concatena y aplica cinco capas *fully connected* y arroja como salida un vector de dimensión siete, los primeros tres valores representan la translación y los últimos cuatro el cuaternión de rotación.

Aunque todos estos métodos tienen un alto rendimiento en cuanto al alineamiento, la exactitud de su resultado depende mucho de la forma de las nubes de puntos de entrada ya que, si presentan formas muy diferentes, su rendimiento disminuirá.

**Tabla 1.2.** Algoritmos para el registro de nubes de puntos basados en aprendizaje profundo.

<b>Autores</b>	<b>Título</b>	<b>Año</b>	<b>Técnica</b>	<b>Implementación</b>	
Lu, et al.	DeepVCP: An End-to-End Deep Neural Network for point cloud registration	2019	PointNet++ / mini-PointNet	KITTI	Nvidia GTX 1080
Aoki, et al.	PointNet LK Robust & Efficient Point Cloud Registration using PointNet	2019	PointNet + Lucas-Kanade	ModelNet	Intel Xeon 2GHz
Wan & Solomon	Deep Closest Point: Learning Representations for Point Cloud Registration	2019	DGCNN+ICP	ModelNet	Nvidia GTX 1070
Sarode, et al.	PCRNNet Point Cloud Registration Network using PointNet Encoding	2019	PointNet + MLP FC	ModelNet	Nvidia GTX 1070

## 1.2 PLANTEAMIENTO DEL PROBLEMA.

En aplicaciones reales los métodos para el registro de las nubes de puntos sufren diversos retos debido a la calidad de los datos de entrada. Para realizar la reconstrucción de una escena se requiere tomar capturas de distintas poses, esto puede causar la deformación entre las nubes de puntos.

Otro factor importante es el ruido ocasionado por los sensores utilizados para la adquisición de los datos ya que pueden ocasionar la aparición de puntos fuera de línea o valores atípicos, los cuales no tienen correspondencias con otras nubes de puntos (Zhu et al., 2019).

Una alta dimensionalidad y los conjuntos de puntos masivos son encontrados comúnmente en el mundo real, un sensor LiDAR obtiene cerca de un millón de puntos mientras escanea la superficie por lo que se requiere una gran capacidad computacional en estos sistemas para procesar, guardar y modificar todos estos datos.

Además de esto, los métodos locales presentan dos grandes problemáticas. Por una parte, requieren de una buena inicialización ya que es indispensable para obtener óptimos resultados y de esto dependerá su tiempo de convergencia, por otra parte, estos métodos son susceptibles a caer en un mínimo local (Zhang, Dai & Sun, 2020).

Para solucionar estos problemas se han desarrollado diferentes técnicas y variaciones de los algoritmos tradicionales. Sin embargo, su tiempo de convergencia se incrementa de manera considerable y aun así no se garantiza una buena alineación en todos los casos.

### 1.3 JUSTIFICACIÓN.

En años recientes el campo de aplicación de la reconstrucción 3D ha ido incrementando y abarcando distintas áreas de estudio como el sector médico, la impresión 3D y la arquitectura, aunque su mayor auge lo presenta en la robótica, visión y gráficos por computadora y el sector automotriz.

En la robótica se han desarrollado grandes avances en los sistemas de localización y mapeo simultaneo o sistemas SLAM (*simultaneous localization and mapping*) (Durrant-Whyte & Bailey, 2006), así como en vehículos de exploración y vehículos de búsqueda y rescate (Chen, Ihara & Hasegawa, 2020).

En visión por computadora y procesamiento de imágenes digitales se han desarrollado sistemas de realidad aumentada utilizando este proceso (Wu, Chan & Lin, 2019).

En el sector automotriz su mayor aplicación está en los vehículos autónomos utilizando sensores LiDAR para obtener los datos de profundidad y realizar un mapeo de los alrededores (Lee, Song & Jo, 2016).

El registro de nubes de puntos impacta directamente a todas estas aplicaciones por lo que el desarrollo de nuevas técnicas y algoritmos cada vez más robustos y de alto rendimiento se ha vuelto algo imprescindible y, por lo tanto, todo un objeto de investigación.

## **1.4 HIPÓTESIS.**

Es posible obtener una reconstrucción 3D más exacta y precisa por medio de una cámara RGB-D y la implementación de algoritmos de aprendizaje profundo.

## **1.5 OBJETIVOS.**

### **1.5.1 Objetivo general.**

Desarrollar, implementar y evaluar un algoritmo de reconstrucción 3D para un entorno cerrado por medio de una cámara RGB-D mediante el uso de algoritmos de aprendizaje profundo.

### **1.5.2 Objetivos específicos.**

- Implementar un algoritmo basado en aprendizaje profundo para el registro de nubes de puntos.
- Desarrollar un algoritmo para la adquisición de datos y la reconstrucción 3D.
- Implementar ambos algoritmos en conjunto con un sensor RGB-D y evaluar su funcionamiento en una PC.
- Comparar los resultados obtenidos con algún método del estado del arte con fines de validación.

## 2. MARCO TEÓRICO

### 2.1 INTELIGENCIA ARTIFICIAL.

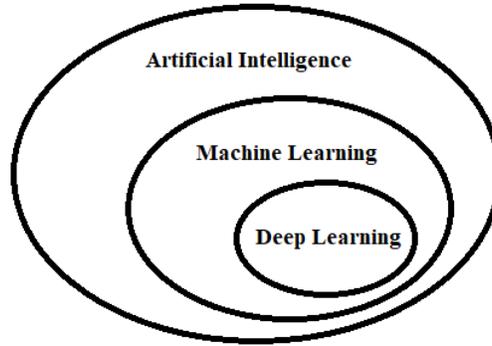
Dar una definición exacta de lo que es la inteligencia artificial es una tarea muy complicada, sobre todo porque depende de la propia palabra “inteligencia”, que al día de hoy sigue teniendo múltiples interpretaciones, tomando una definición, se puede considerar como inteligencia artificial (IA) a toda aquella máquina capaz de imitar las funciones cognitivas que los humanos asocian con otras mentes humanas, por ejemplo: percibir, razonar, aprender y resolver problemas (Russell & Norvig, 2009).

Dentro del campo de la inteligencia artificial, podemos encontrar diferentes subcategorías que responden a diferentes comportamientos inteligentes, por ejemplo: el campo de la robótica con robots capaces de adaptarse en un entorno; el campo del procesamiento de lenguaje natural con la capacidad de entender el lenguaje, entre otros. Sin embargo, si existe una capacidad que nos define como agentes inteligentes es la capacidad de aprender.

El Machine Learning es una rama del campo de la inteligencia artificial que le da a las máquinas la habilidad de aprender sin ser explícitamente programadas para ello. Dentro del Machine Learning existen diferentes técnicas para cubrir distintos tipos de aplicaciones, por ejemplo: árboles de decisión, modelos de regresión, modelos de clasificación, técnicas de caracterización, entre otras, sin embargo, la que más se ha destacado dentro de esta rama de la inteligencia artificial es el Deep Learning.

El Deep Learning, o aprendizaje profundo, constituye un conjunto particular de algoritmos de Machine Learning que utilizan estructuras profundas, mejor conocidas como redes neuronales, para encontrar patrones en los datos. Estos tipos de algoritmos cuentan actualmente con un gran interés, ya que han demostrado ser sumamente exitosos para resolver determinados tipos de problemas como el reconocimiento de imágenes y la detección de objetos. Muchos consideran que este tipo de algoritmos son los que nos llevarán a resolver definitivamente el problema de la Inteligencia Artificial.

En la Fig. 2.1 se muestra un esquema de la inteligencia artificial y sus principales áreas.



**Fig. 2.1.** Inteligencia Artificial y sus principales áreas (Goodfellow & Bengio, 2017).

### **2.1.1 Redes neuronales artificiales.**

Las redes neuronales artificiales están inspiradas en las redes neuronales biológicas del cerebro humano y están constituidas por elementos que se comportan de forma similar a las neuronas biológicas en sus funciones más comunes y se organizan de una forma parecida a la que presenta el cerebro humano (Basogain, 2012).

Las redes neuronales artificiales presentan una serie de características propias del cerebro, las más destacadas son:

- **Aprendizaje:** adquieren su conocimiento mediante el estudio, ejercicio o experiencia y, además, pueden cambiar su comportamiento en función del entorno. Se les muestra un conjunto de entradas y ellas mismas se ajustan para producir unas salidas consistentes.
- **Generalización:** generalizan automáticamente debido a su propia estructura y naturaleza. Estas redes pueden ofrecer, dentro de un margen, respuestas correctas a entradas que presentan pequeñas variaciones debido a los efectos de ruido o distorsión.
- **Abstracción:** aíslan mentalmente o consideran por separado las cualidades de un objeto. Algunas redes neuronales son capaces de abstraer la esencia de un conjunto de entradas que aparentemente no presentan aspectos comunes o relativos.

El elemento básico de un sistema neuronal biológico es la neurona. Un sistema neuronal biológico está compuesto por millones de neuronas organizadas en capas. En función de esto,

una red neuronal artificial trata de emular el funcionamiento y la estructura de una red neuronal biológica donde, por medio de un sistema neuronal artificial, se puede establecer una estructura jerárquica similar a la existente en el cerebro. El elemento esencial será la neurona artificial, la cual se organizará en capas, dichas capas constituirán la red neuronal (Larrañaga, Inza & Moujahid, 2011).

En esencia, se aplica un conjunto de entradas  $x_i$  a la neurona  $y_j$ , cada una de las cuales están enviando señales de entradas o valores numéricos. Cada entrada se multiplica por su peso  $w_{ji}$  o ponderación correspondiente, análogo al grado de conexión de la sinapsis en la red neuronal biológica. Todas las entradas ponderadas se suman y se determina el nivel de excitación o activación de la neurona por medio de una función de activación, dicha función de activación puede ser una función de escalón o una función sigmoidea, que simulan con mayor exactitud las características de transferencia no lineales de las neuronas biológicas. En la Fig. 2.2 se muestra el esquema de una red neuronal y en la Fig. 2.3 algunas funciones de activación típicas no lineales (Izaurieta & Saavedra, 2015).

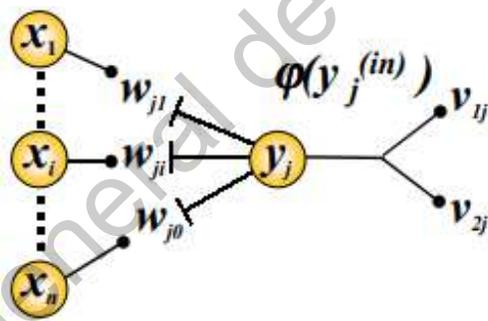


Fig. 2.2. Esquema de una neurona artificial (Izaurieta & Saavedra, 2015).

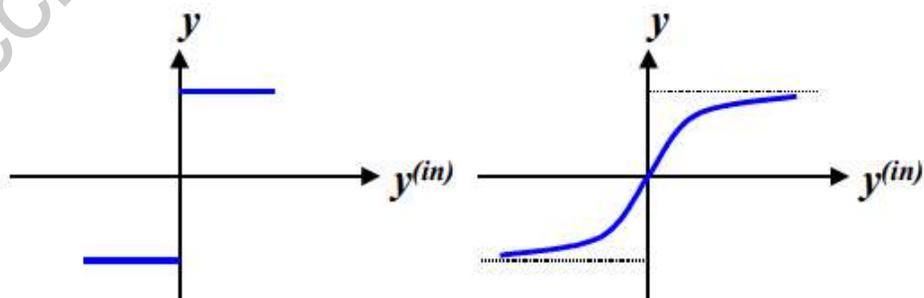


Fig. 2.3. Función escalón (Izq.), función sigmoidea (Der.), (Izaurieta & Saavedra, 2015).

La capacidad de cálculo y potencia de la computación neuronal proviene de las múltiples conexiones de las neuronas artificiales que constituyen una red neuronal artificial. La red más simple es un grupo de neuronas ordenadas en una capa como se muestra en la Fig. 2.4.

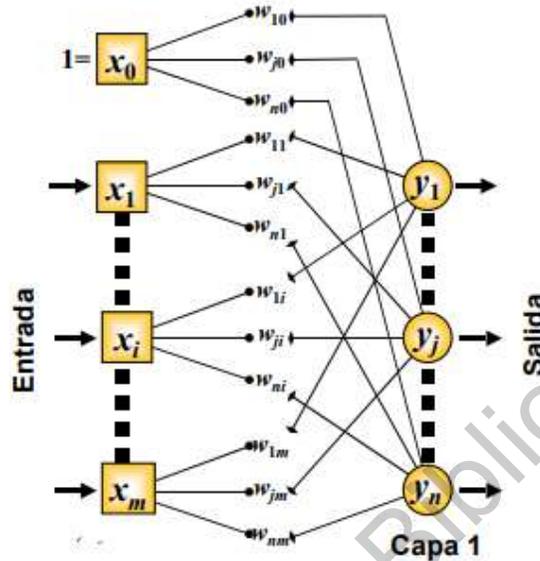


Fig. 2.4. Red neuronal unicapa (Izaurieta & Saavedra, 2015).

Normalmente las redes más complejas y más grandes ofrecen mejores prestaciones en el cálculo computacional que las redes simples. Las configuraciones de las redes construidas presentan aspectos muy diferentes, pero tienen un aspecto común, el ordenamiento de las neuronas en capas o niveles imitando la estructura de capas que presenta el cerebro en algunas partes. Las redes multicapa se forman con un grupo de capas simples en cascada. La salida de una capa es la entrada de la siguiente capa. Se ha demostrado que las redes multicapa presentan cualidades y aspectos por encima de las redes de una capa simple. En la Fig. 2.5 se muestra el esquema de una red multicapa (Larrañaga, Inza & Moujahid, 2011).

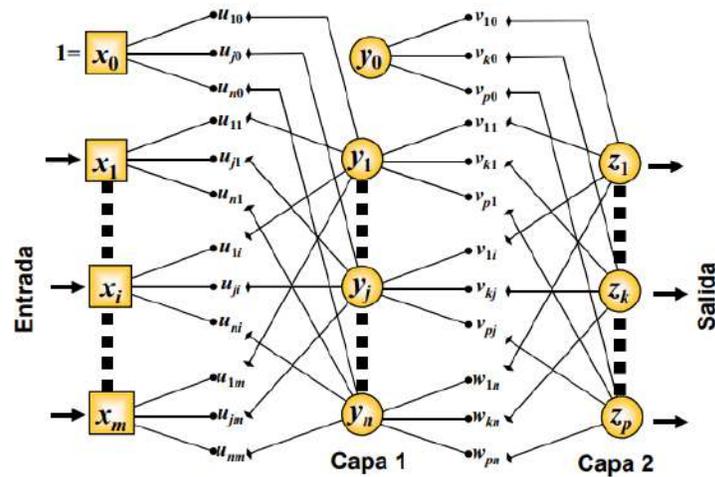


Fig. 2.5. Red neuronal multicapa (Izaurieta & Saavedra, 2015).

## 2.2 RECONSTRUCCIÓN 3D.

En visión por computadora, la reconstrucción 3D es el proceso mediante el cual se obtienen las propiedades geométricas de una escena real por medio del procesamiento y combinación de datos. El objetivo de la reconstrucción 3D basada en imágenes es inferir la geometría 3D y la estructura de objetos y escenas a partir de una o múltiples imágenes en 2D. Este problema es fundamental para muchas aplicaciones, como la navegación de robots, el reconocimiento de objetos y la comprensión de la escena, el modelado 3D y la animación, el control industrial y el diagnóstico médico, entre otros (Han, Laga & Bennamoun, 2019).

El proceso de reconstrucción 3D está constituido por tres secciones: la adquisición y pre procesamiento de los datos, el registro de las nubes de puntos y finalmente el mapeo 3D de una escena. En la Fig. 2.6 se muestra el diagrama del proceso de reconstrucción 3D.

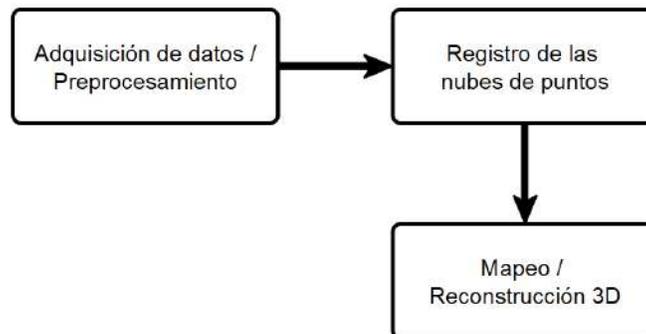


Fig. 2.6. Proceso de Reconstrucción 3D. (Intwala & Magikar, 2016).

## 2.3 ADQUISICIÓN DE DATOS Y PREPROCESAMIENTO.

Para realizar la reconstrucción de una escena es necesario realizar múltiples escaneos desde diferentes posiciones y ángulos para capturar distintas instancias que describen la superficie dicha escena, mejor conocidas como nubes de puntos. Una vez que se adquieren las nubes de puntos se les realiza un preprocesamiento, generalmente constituido por la normalización de los datos y el muestreo o reducción de puntos. En la Fig. 2.7 se muestran dos nubes de puntos tomadas de una escena desde distintos ángulos.



Fig. 2.7. Nubes de puntos de una escena tomadas de distintos ángulos.

### 2.3.1 Datos de entrada.

Los datos de entrada para un sistema de reconstrucción 3D son generalmente conocidos como nubes de puntos. Una nube de puntos es un conjunto de puntos dentro de un sistema de coordenadas tridimensionales donde cada punto tiene coordenadas  $x, y, z$ . En la Fig. 2.8 se muestra un ejemplo de una nube de puntos desde una vista lateral.



Fig. 2.8. Ejemplo de una nube de puntos desde una vista lateral.

### 2.3.2 Tipos de sensores para la adquisición de datos.

Existen diferentes técnicas para realizar la adquisición de datos en la reconstrucción 3D de un objeto, en la Fig. 2.9 se muestra la taxonomía de las principales técnicas utilizadas en la adquisición de datos 3D.



**Fig. 2.9.** Taxonomía de las principales técnicas utilizadas para la adquisición de imágenes en 3D (Pedraza et al. 2011).

Las cámaras 3D, o sensores RGB-D, son sensores matriciales que estiman la profundidad y capturan mapas de alcance. Al igual que las cámaras en color, proporcionan imágenes del entorno circundante desde un único punto de vista. Junto con la información de color, los sensores RGB-D proporcionan mediciones de profundidad al explotar la información visual. En la Fig. 2.10 se muestran algunos de los principales sensores RGB-D visuales que se encuentran en el mercado y en la Tabla 2.1 se enlistan algunas de sus características.



**Fig. 2.10.** Sensores RGB-D visuales más populares en el mercado. Kinect V1 (Izq.), RealSense D435 (Centro) y StereoLabs ZED (Der.).

**Tabla 2.1.** Características de algunos sensores RGB-D visuales encontrados en el mercado.

<b>Dispositivo</b>	<b>Tecnología</b>	<b>Rango (m)</b>	<b>Resolución</b>	<b>FPS</b>	<b>Campo de visión</b>
Kinect V1	Luz estructurada	0.8 – 4.0	640 x 480	30	57° x 43°
Kinect V2	Tiempo de vuelo	0.5 – 4.5	512 x 424	30	70° x 60°
Intel D435	Estereoscópico activo	0.2 – 10	1280 x 720	90	85.2° x 58°
StereoLabs ZED	Estereoscópico pasivo	0.5 – 20	4416 x 1242	100	110° (Diagonal)

### 2.3.3 Preprocesamiento de los datos.

Una parte muy importante dentro del sistema de reconstrucción 3D es el preprocesamiento de los datos. Generalmente los modelos del estado del arte trabajan con información obtenida directamente de una base de datos, esto causa que al llevar dichos algoritmos a aplicaciones con datos del mundo real tengan problemas de funcionamiento y resultados poco eficientes por lo que el preprocesamiento de los datos de entrada es parte fundamental del proceso.

Generalmente en esta parte del proceso se realiza un filtrado para reducir el nivel de ruido, la detección y eliminación de puntos fuera de línea. También se realiza un muestreo para reducir la cantidad de puntos contenido en cada conjunto de puntos para que, de esta manera, se reduzca el tiempo de procesamiento y el proceso se optimice. Por último, se realiza la normalización de los datos en una esfera o caja unitaria.

## 2.4 REGISTRO DE LAS NUBES DE PUNTOS.

El registro de nubes de puntos tridimensionales se centra en la estimación de una transformación de cuerpo rígido entre varias nubes de puntos superpuestas para asociarlas en un sistema de coordenadas común. Esto permite integrar datos de múltiples sensores y puntos de vista en un modelo mayor, como un mapa tridimensional más preciso y una malla completa de objetos (Zhang et al., 2020).

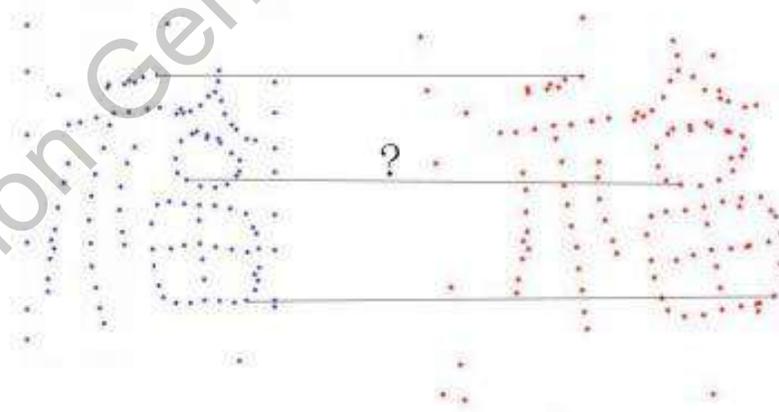
A grandes rasgos, su objetivo principal es encontrar correspondencias entre dos o más conjuntos de puntos y obtener una matriz de rotación junto con un vector de traslación de tal manera que los puntos se vayan alineando unos con otros. De manera formal, se tienen dos nubes de puntos  $X$  (fuente) y  $Y$  (objetivo), con  $N_x$  y  $N_y$  puntos, respectivamente.  $X$  representa la misma forma o escena que  $Y$ , pero presenta ciertas variaciones por ruido o ángulo. Se desea encontrar una transformación 3D de cuerpo rígido para transformar a  $X$  en  $X'$ , de tal manera que  $X'$  se alinee de mejor manera con  $Y$  (Zhang et al., 2020).

Normalmente el registro de nubes de puntos se divide en dos categorías: registro por pares y el registro por puntos.

#### 2.4.1 Registro por pares.

En el registro por pares sólo se consideran dos conjuntos o nubes de puntos durante el proceso de registro. En la Fig. 2.11 se muestra un ejemplo de registro de nubes de puntos por pares. El objetivo del registro de nubes de puntos por pares es encontrar la transformación adecuada y establecer las correspondencias correctas entre ambas nubes de puntos (Zhu et al., 2019).

Varios métodos han sido desarrollados para resolver este problema. Estos métodos pueden ser clasificados en tres categorías: métodos basados en distancia, filtrado y probabilidad.



**Fig. 2.11.** Registro de nubes de puntos por pares con búsqueda de correspondencias (Zhu et al., 2019).

### 2.4.1.1 Métodos basados en distancia.

Los métodos para el registro de nubes de puntos basados en distancia trabajan bajo un esquema de dos pasos. El primero es calcular la distancia entre dos nubes de puntos y encontrar sus puntos de correspondencia. Después, la distancia entre ambas nubes de puntos con sus determinados puntos de correspondencia es minimizada en el segundo paso (Zhu et al., 2019).

Uno de los métodos más representativos de este grupo es el algoritmo iterativo del punto más cercano, ICP por sus siglas en inglés (*iterative closest point*). ICP puede ser expresado como un problema de optimización como se muestra en la ecuación 2.1.

$$\arg \min (R, t) \left\{ \frac{1}{M} \sum_{j=1}^M \|y_j - (Rx_j + t)\|_2 \right\} \quad (2.1)$$

Donde  $y_j$  y  $x_j$  es un par de puntos correspondientes entre sí,  $\|\cdot\|_2$  es la norma euclidiana,  $R$  y  $t$  son la matriz de rotación y el vector de traslación respectivamente y  $M$  el número de pares de correspondencia.

### 2.4.1.2 Métodos basados en filtros.

Los métodos para el registro de nubes de puntos basados en filtrado utilizan un modelo espacial de estados, SSM por sus siglas en inglés (*state space model*). De manera general un SSM es formulado como se muestra en la ecuación 2.2.

$$x'_k = x'_{k-1} + v_k \quad , \quad y_k = f(x'_k, x_k) + w_k \quad (2.2)$$

Donde  $y_k$  y  $x_k$  son los puntos de dos conjuntos,  $x'_k$  es el estado en el instante  $k$ , y puede ser descrito como  $x'_k = [t_k^x, t_k^y, \theta_k]^T$  en conjuntos 2D;  $t_k^x$  y  $t_k^y$  son los parámetros de traslación en los ejes  $x$ ,  $y$ ; y  $\theta_k$  el parámetro de rotación en el instante  $k$ . Para conjuntos 3D, el estado es denotado como  $x'_k = [t_k^x, t_k^y, t_k^z, \theta_k^x, \theta_k^y, \theta_k^z]^T$ ;  $t_k^z$  es el parámetro de traslación en el eje  $z$  y  $\theta_k^x, \theta_k^y, \theta_k^z$  son los parámetros de rotación en los ejes  $x$ ,  $y$ ,  $z$  en el instante  $k$ , respectivamente,  $f(\cdot)$  es la función de medida y  $v_k, w_k$  son el ruido del proceso y ruido gaussiano respectivamente.

### 2.4.1.3 Métodos basados en probabilidad.

La deriva de punto coherente, CPD por sus siglas en inglés (*coherent point drift*), es un método popular en el campo del registro de conjuntos de puntos basado en probabilidades. En el método CPD, un registro de conjunto de puntos rígido y no rígido se formula como un problema de estimación de máxima verosimilitud (ML) utilizando el método GMM. Un conjunto de puntos está representado por centroides GMM, y el otro conjunto de puntos se ajusta a los del primer conjunto de puntos moviéndose coherentemente. En la ecuación 2.3 se describe este método.

$$\rho(Y) = \prod_{j=1}^M \prod_{i=1}^N \pi_i \eta(y_j | \delta(x_i), \sigma^2 I) \quad (2.3)$$

Donde  $\eta(\cdot)$  es la distribución gaussiana,  $\delta(\cdot)$  es la transformación rígida o no rígida,  $\sigma^2$  las covarianzas isotrópicas iguales,  $I$  la matriz identidad y  $\pi_i$  el coeficiente de mezcla. Luego, se aplica un algoritmo de maximización de expectativas (EM) para realizar esta optimización de ML.

### 2.4.2 Registro por grupos.

En el registro por grupos se consideran dos o más nubes de puntos durante el proceso de registro. Tradicionalmente, este problema es realizado utilizando el registro por pares de manera repetitiva como una estrategia secuencial o una estrategia de uno contra todos. En la estrategia de registro secuencial por pares, los parámetros se actualizan mediante un método basado en distancia o un método probabilístico cuando se dispone de conjuntos de puntos adicionales. El principal inconveniente de la estrategia de registro secuencial por pares consiste en la propagación del error en los pasos siguientes. Para la estrategia de registro por pares de uno contra todos, el conjunto de puntos de referencia debe elegirse de antemano. Los otros conjuntos de puntos se utilizan para registrarse con el conjunto de puntos de referencia.

## 2.5 ALGORITMOS PARA EL REGISTRO DE NUBES DE PUNTOS.

El proceso de registro de las nubes de puntos se divide generalmente en dos partes: inicialización o pre-alineación, y el emparejamiento fino o transformación afín. En la pre-alineación se busca una alineación inicial rígida de las nubes de puntos sin alguna transformación inicial dada. En la literatura, estos tipos de métodos son comúnmente llamados métodos globales o libres de correspondencias. Por el otro lado, el emparejamiento fino incluye métodos que comienzan con una transformación inicial aproximada y tratan de encontrar una alineación tan exacta como sea posible. Este tipo de métodos son conocidos como métodos locales o métodos basados en correspondencias (García, 2017).

En la Fig. 2.12 se muestra un esquema general del proceso de registro de nubes de puntos.



**Fig. 2.12.** Esquema general del proceso de registro de nubes de puntos.

### 2.5.1 Métodos tradicionales.

Los algoritmos basados en métodos tradicionales pueden ser divididos dentro de dos categorías: los métodos locales y los métodos globales. ICP es un método local para el registro de conjuntos de puntos que se basa en la distancia euclidiana de pares de puntos entre ambos conjuntos.

En su forma básica ICP empieza con una alineación inicial sin conocimiento alguno de correspondencias entre ambos conjuntos de puntos y después opera de manera iterativa bajo dos pasos: 1) buscar y establecer correspondencias entre los puntos cercanos de ambos conjuntos, y 2) recalculan la estimación de la transformada por medio de mínimos cuadrados para encontrar la matriz de rotación, así como el vector de traslación que permita alinear de mejor manera ambas nubes de puntos. Una buena inicialización del algoritmo ICP es

indispensable para obtener óptimos resultados y de ello dependerá su convergencia y susceptibilidad de caer en un mínimo local (Zu et al., 2019). Es por esto que este algoritmo debería ser aplicado sólo en la transformación afín.

Algunas de las soluciones más comunes ante esta situación son el uso de optimizadores globales como refinamiento y la utilización de métodos de registro globales, los cuales se basan en extraer las características geométricas de las nubes de puntos para realizar el registro. Debido a que los métodos de registro global no requieren una aproximación inicial, usualmente son utilizados como inicialización para los métodos locales con el fin de obtener mejores resultados (Zhou, Park & Koltun, 2016).

Es común el uso de gráficos de pose en sistemas de reconstrucción 3D y sistemas SLAM. Este método hace la representación de una estructura de gráficos constituida por nodos y bordes que facilita el procesamiento y es flexible en términos de observación en sistemas que utilizan sensores RGB-D. Por estas razones son una herramienta muy práctica y fácil de implementar cuando se requiere optimizar el mapeo de una escena (Yan et al., 2019).

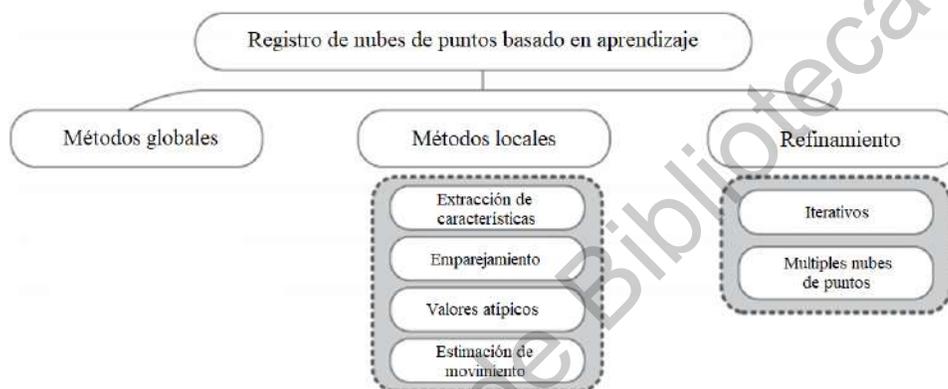
### **2.5.2 Métodos basados en aprendizaje profundo.**

El éxito del aprendizaje profundo en tareas de visión de alto nivel se ha extendido a varias tareas geométricas de visión por computadora, como la estimación del movimiento, la correspondencia estéreo, el flujo óptico y la múltiple vista estéreo. Sin embargo, la extensión a las nubes de puntos 3D no es sencilla debido a las diferencias significativas entre las imágenes 2D y 3D. La escasez y los patrones irregulares, las permutaciones desordenadas y una distribución desequilibrada constituyen los principales obstáculos (Zhang et al., 2020).

Los métodos basados en aprendizaje profundo pueden ser divididos en dos categorías: métodos basados en correspondencias y métodos libres de correspondencias. Los métodos basados en correspondencias ocupan una proporción significativa de los métodos para el registro de nubes de puntos profundo. Estos métodos están inspirados en los métodos tradicionales y se centran en encontrar correspondencias a nivel local. Los métodos libres de correspondencias se centran en encontrar los parámetros de movimiento rígido buscando la diferencia entre las características globales de las nubes de puntos (Zhang et al., 2020).

Aunque todos estos métodos tienen un alto rendimiento en cuanto al alineamiento, la exactitud que presentan en nubes de puntos relativamente diferentes, la sensibilidad al ruido y los puntos fuera de línea de las características representan un bajo límite en su rendimiento (Zhang et al., 2020).

En adición a estos dos métodos, otras técnicas se han propuesto para refinar el desempeño de todos estos algoritmos de manera significativa. Los más representativos son los métodos iterativos y el registro múltiple de nubes de puntos. En la Fig. 2.13 se muestra la clasificación de los métodos para el registro de las nubes de puntos basados en aprendizaje profundo.



**Fig. 2.13.** Métodos basados en aprendizaje profundo (Zhang et al., 2020).

### 3. METODOLOGÍA

Esta tesis propone el uso de algoritmos de aprendizaje profundo para el registro de nubes de puntos en un sistema de reconstrucción 3D. La metodología implementada se divide en dos secciones. En la primera parte se describen los pasos necesarios para la implementación de un sistema de reconstrucción 3D basado en métodos tradicionales, esto es, sin ningún módulo de aprendizaje de por medio.

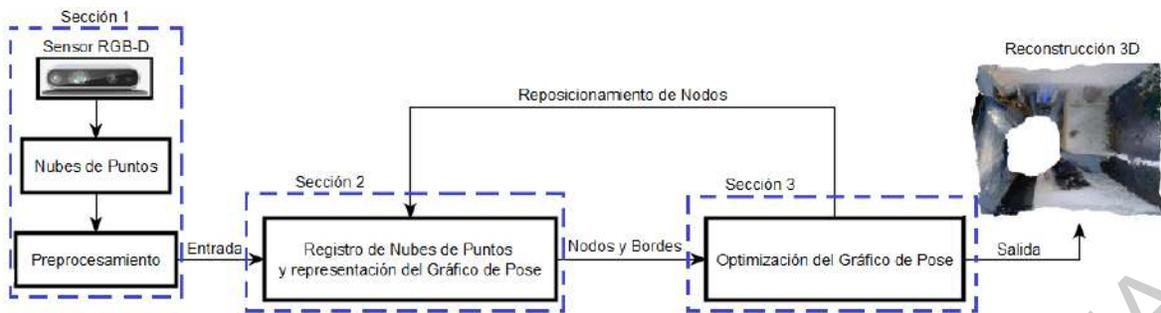
En la segunda parte se describe detalladamente los métodos y técnicas implementadas para la construcción de un sistema de reconstrucción 3D basado en algoritmos de aprendizaje profundo.

#### 3.1 SISTEMA DE RECONSTRUCCIÓN 3D BASADO EN MÉTODOS TRADICIONALES.

El modelo del sistema de reconstrucción 3D se muestra en la Fig. 3.1. El sistema está dividido en 3 secciones: En la primera parte se realiza la adquisición de las nubes de puntos por medio de un sensor RGB-D y se realiza su preprocesamiento.

En la segunda sección se detalla el proceso del registro de las nubes de puntos constituido por dos partes; la pre-alineación por medio del algoritmo propuesto en (Zhou et al., 2016) junto con la transformación afín de las nubes de puntos por medio de ICP punto a plano, y la representación de los nodos y bordes para la construcción del gráfico de pose.

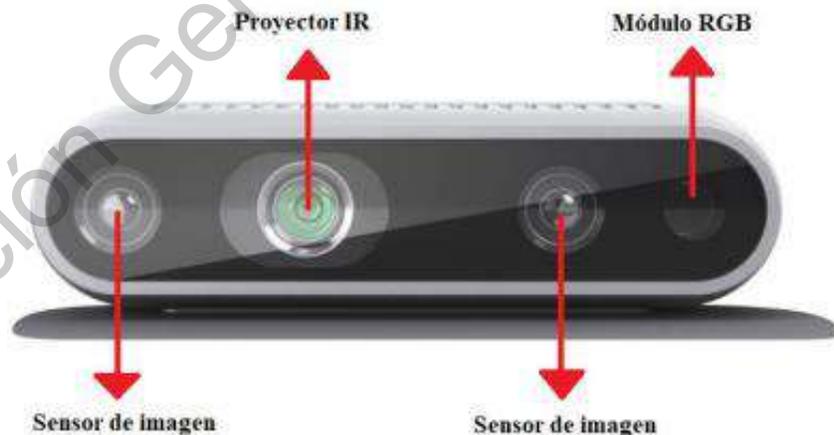
Finalmente, en la tercera sección se muestra la optimización a nivel global del gráfico sobre los nodos y bordes generados. Después de esto se retorna la nueva posición de los nodos al algoritmo de alineación para realizar el registro con una nueva nube de puntos, este proceso se va repitiendo con cada una de las nubes de puntos que se adquieran para el mapeo de la escena interior tomando como nuevo objetivo el resultado del proceso hecho previamente y como fuente una nueva nube de puntos previamente capturada por el sensor de profundidad.



**Fig. 3.1.** Modelo del sistema de reconstrucción 3D.

### 3.1.1 Adquisición y preprocesamiento de los datos.

Una parte muy importante dentro del sistema de reconstrucción 3D es la adquisición de los datos. Generalmente los modelos del estado del arte trabajan con información obtenida directamente de una base de datos, esto causa que al llevar dichos algoritmos a aplicaciones con datos del mundo real tengan problemas de funcionamiento y resultados poco eficientes por lo que el preprocesamiento de los datos de entrada es parte fundamental del proceso. Para este trabajo se realiza la adquisición de las nubes de puntos por medio del sensor de profundidad Intel RealSense D435, mostrado en la Fig. 3.2. El sensor está compuesto por un sensor RGB, un proyector infra rojo y dos sensores de imagen. En la Tabla 3.1 se presentan sus características principales.



**Fig. 3.2.** Sensor de profundidad Intel RealSense D435 (<https://www.intelrealsense.com/depth-camera-d435>).

**Tabla 3.1.** Especificaciones cámara Intel RealSense D435.

<b>Característica</b>	<b>Descripción</b>
Rango máximo	Aprox. 10 metros
Tecnología de profundidad	IR estéreo activa
Campo de profundidad	86° x 57° ( $\pm 3^\circ$ )
Dist. Min. de profundidad	0.105 metros
Resolución de profundidad	1280 x 720
Resolución RGB	1920 x 1080
Frame Rate profundidad	90 fps
Frame Rate RGB	30 fps

La cámara fue montada sobre una base con ayuda de un tripié como se muestra en la Fig. 3.3. Para la captura de las nubes de puntos, la base se fue rotando cuidando que entre cada captura no se superara un ángulo de rotación mayor a  $10^\circ$  ya que de ser mayor el registro se hace ineficiente debido a los puntos de correspondencia entre ambas nubes de puntos.

La resolución del sensor de profundidad para la adquisición de las nubes de puntos es un factor importante a considerar ya que de ello depende un mejor muestreo del entorno, pero de igual manera también se ve afectado el tiempo de procesamiento. Debido a esto se debe tener cuidado al seleccionar el parámetro de resolución. Para este trabajo se determinó una resolución de 640x480 pixeles de entrada para el sensor de profundidad.

Una vez capturadas las nubes de puntos se realiza el preprocesamiento para su entrada al algoritmo de alineación. En esta parte se reduce el número de puntos contenido dentro de cada conjunto para optimizar el proceso, después se normalizan los datos y por medio de (Rusu, Blodow & Beetz, 2009) se genera el conjunto de correspondencias iniciales para el algoritmo de pre-alineación.



**Fig. 3.3.** Montaje de la cámara de profundidad para el muestreo de las nubes de puntos.

### 3.1.2 Registro de las nubes de puntos.

El proceso de registro está constituido por tres partes: en las primeras dos se trata el problema del registro de las nubes de puntos dividido en la pre-alineación y la transformación afín y en la tercera parte se detalla la representación del gráfico de pose.

#### 3.1.2.1 Pre-alineación (Fast Global Registration).

Una vez procesados los datos de las nubes de puntos  $P = \{p_1, p_2, \dots, p_N\} \subset \mathbb{R}^3$ ,  $Q = \{q_1, q_2, \dots, q_N\} \subset \mathbb{R}^3$  y obtenido el conjunto de correspondencias iniciales  $K = \{(p_i, q_i), \dots, (p_m, q_m)\}$ , se realiza la pre-alineación de las nubes de puntos por medio del algoritmo *Fast Global Registration* (Zhou, Park & Koltun, 2016). El fin de este algoritmo es optimizar la transformación rígida  $M$  con la forma mostrada en la ecuación 3.1, sin embargo, en esta parte el objetivo es ofrecer una buena estimación de  $M$  para la transformación afín.

$$E(M) = \sum_{(p_i, q_i) \in K} \vartheta(\|p_i - Mq_i\|) \quad (3.1)$$

Donde  $\vartheta$  es el estimador escalado de German-McClure para diferentes valores del residuo  $\mu$  con la forma mostrada en la ecuación 3.2.

$$\vartheta(\|p_i - Mq_i\|) = \frac{\mu(\|p_i - Mq_i\|)^2}{\mu + (\|p_i - Mq_i\|)^2} \quad (3.2)$$

### 3.1.2.2 Transformación Afín (ICP punto a plano).

Una vez que se tiene la pre-alineación de (Zhou, Park & Koltun, 2016) se pasa el resultado obtenido de la transformación inicial  $M$  y su información al algoritmo ICP punto a plano para realizar la transformación afín.

Cuando se utiliza ICP punto a plano se busca minimizar la distancia de la suma de los cuadrados entre cada punto perteneciente a la fuente y el plano de la nube de puntos objetivo en cada uno de sus puntos de correspondencia (Low, 2004). Para cada punto perteneciente a  $P$ , el punto más cercano en  $Q$  es tomado como correspondencia. Este proceso se repite con la finalidad de encontrar una transformación 3D de cuerpo rígido  $M = [R_{PQ}(\alpha, \beta, \gamma), t_{PQ}(t_x, t_y, t_z)]$ , donde  $R_{PQ}(\alpha, \beta, \gamma) \in SO(3)$  y  $t_{PQ}(t_x, t_y, t_z) \in \mathbb{R}^3$ , que transforme a la fuente  $Q$  tal que el error total de las correspondencias entre ambas nubes de puntos sea el mínimo (Liu et al, 2018).

De manera formal, si  $q_i = (q_{ix}, q_{iy}, q_{iz}, 1)^T$  es un punto perteneciente a la fuente  $Q$ ,  $p_i = (p_{ix}, p_{iy}, p_{iz}, 1)^T$  es el punto de correspondencia perteneciente al objetivo  $P$  y  $n_i = (n_{ix}, n_{iy}, n_{iz}, 0)^T$  es el vector normal en  $p_i$ , como se muestra en la ecuación 3.3, se desea obtener la transformación  $M_F$ .

$$M_F = \underset{(p_i, q_i) \in K}{\operatorname{argmin}_M} \sum ((Mq_i - p_i) \cdot n_i)^2 \quad (3.3)$$

Donde  $M$  y  $M_F$  representan las matrices 4x4 de la transformación rígida.

La matriz de transformación de cuerpo rígido  $M_F$  está compuesta por una matriz de rotación  $R_{PQ}(\alpha, \beta, \gamma)$  y una matriz de traslación  $t_{PQ}(t_x, t_y, t_z)$  como se muestra en las ecuaciones 3.4 y 3.5 respectivamente.

$$R_{PQ}(\alpha, \beta, \gamma) = R_{PQ_z}(\gamma) \cdot R_{PQ_y}(\beta) \cdot R_{PQ_x}(\alpha) = \begin{pmatrix} R_{PQ_{11}} & R_{PQ_{12}} & R_{PQ_{13}} & 0 \\ R_{PQ_{21}} & R_{PQ_{22}} & R_{PQ_{23}} & 0 \\ R_{PQ_{31}} & R_{PQ_{32}} & R_{PQ_{33}} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (3.4)$$

$$t_{PQ}(t_x, t_y, t_z) = \begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (3.5)$$

Donde

$$R_{PQ_{11}} = \cos(\gamma)\cos(\beta),$$

$$R_{PQ_{12}} = -\sin(\gamma)\cos(\alpha) + \cos(\gamma)\sin(\beta)\sin(\alpha),$$

$$R_{PQ_{13}} = \sin(\gamma)\sin(\alpha) + \cos(\gamma)\sin(\beta)\cos(\alpha),$$

$$R_{PQ_{21}} = \sin(\gamma)\cos(\beta),$$

$$R_{PQ_{22}} = \cos(\gamma)\cos(\alpha) + \sin(\gamma)\sin(\beta)\sin(\alpha),$$

$$R_{PQ_{23}} = -\cos(\gamma)\sin(\alpha) + \sin(\gamma)\sin(\beta)\cos(\alpha).$$

$$R_{PQ_{31}} = -\sin(\beta),$$

$$R_{PQ_{32}} = \cos(\beta)\sin(\alpha),$$

$$R_{PQ_{33}} = \cos(\beta)\cos(\alpha).$$

La salida del algoritmo de alineación es la matriz de transformación  $M_F$  que al ser aplicada a la fuente alinea cada uno de sus puntos con los puntos de correspondencia en la nube de puntos objetivo. Debido a que  $\alpha$ ,  $\beta$  y  $\gamma$  son argumentos no lineales en la matriz de rotación, se hace uso de técnicas de solución para mínimos cuadrados no lineales en la parte de optimización del gráfico de pose.

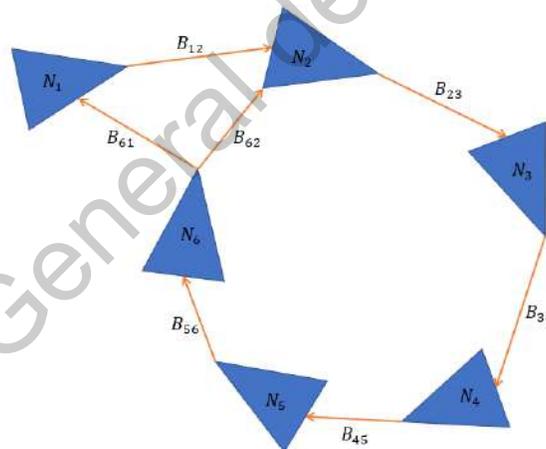
### 3.1.2.3 Optimización del gráfico de pose.

Un gráfico está compuesto principalmente por dos estructuras: los nodos, que representan una pose del sistema en determinado tiempo y los bordes, que representan la relación en

términos de odometría entre cada nodo, la cual puede ser obtenida por medio de la revolución de las llantas del sistema, sistemas con IMU o por medio de observaciones si se utiliza un sistema LiDAR o un sensor de profundidad (Stachniss, 2020).

Se puede denotar la estructura de un gráfico de pose como  $G = \{N, B\}$ . Aquí  $N = \{N_1, \dots, N_m\}$  donde cada uno de los  $N_i$  nodos representan una pose del sistema en el instante  $t_i$  y  $B = \{B_{N_i N_j}, \dots, B_{N_{m-1} N_m}\}$  en donde cada término representa la medición de odometría entre cada nodo. En la Fig. 3.4 se muestra un ejemplo de la estructura de un gráfico de pose.

En la optimización del gráfico, los nodos representan las variables que se desea optimizar y los bordes representan las restricciones entre cada una de las variables. El objetivo de la optimización de un gráfico es construir un gráfico con cada uno de los nodos y bordes que existe entre ellos y cambiar la localización de los nodos mediante su rotación y traslación con tal de encontrar una configuración entre cada uno de los  $n$  nodos tal que minimice el error total o el error cuadrático introducido por cada uno de los bordes en el gráfico (Stachniss, 2020).



**Fig. 3.4** Representación de un gráfico de pose.

Cada uno de los nodos es representado como  $N_i = [\tau_i^T, S_i^T]^T$  y corresponde a un punto en la trayectoria de escaneo, contiene información como su traslación en un sistema global de coordenadas el cual es representado por medio de  $\tau_i = [x_i, y_i, z_i]^T$ . La información como su posición y rotación también es representada en el sistema global de coordenadas por medio del cuaternión  $S_i = [sw_i, sx_i, sy_i, sz_i]^T$ .

La optimización del gráfico de pose es un método global. Esto quiere decir que cada estimación realizada para cada uno de los nodos se verá afectada por cada borde dentro del gráfico, por lo que en este tipo de sistemas una mala estimación de los bordes podría llevar a una mala optimización del gráfico entero (Stachniss, 2020).

Dicho en otras palabras, si el registro de puntos realizado por el algoritmo ICP punto a plano da como resultado una mala alineación, al realizar la optimización del gráfico en ese punto, la reducción del error podría ser deficiente, esto pasa cuando las nubes de puntos procesadas contienen demasiado ruido o existen muchos puntos fuera de línea, también por un mal emparejamiento en los puntos de correspondencia debido a que las nubes de puntos son muy distintas una de la otra, por este motivo se tiene que tener cuidado al momento de capturar los datos de entrada de tal manera que pueda existir cierto número de correspondencia entre ambas nubes (Stachniss, 2020).

De acuerdo a (Yan et al., 2019) la función objetivo que obtiene la pose óptima para cada una de las variables de pose  $\tilde{N} = \{\tilde{N}_1, \dots, \tilde{N}_m\}$  esta dada por la ecuación 3.6.

$$\tilde{N} = \underset{N}{\operatorname{argmin}} \sum_{i=1}^m \sum_{j=1}^m r(N_i, N_j, B_{ij}) \quad i < j \quad (3.6)$$

Donde  $m$  es el número total de nodos en la trayectoria y  $r$  el residuo entre cada borde  $B_{ij}$  entre los nodos  $N_i$  y  $N_j$ .

De esta manera, de acuerdo a la literatura en (Low, 2004), la función objetivo propuesta en la ecuación 3.3 por el algoritmo ICP punto a plano se modifica tomando la forma de la ecuación 3.7.

$$M_F(R, t) = \min \left\{ \sum_{i=1}^m \|((R \cdot q_i + t - p_i) \cdot n_i)\|_2^2 \right\} \quad (3.7)$$

Donde  $q_i = (q_{ix}, q_{iy}, q_{iz}, 1)^T$  es un punto perteneciente a la fuente  $Q$ ,  $p_i = (p_{ix}, p_{iy}, p_{iz}, 1)^T$  es el punto de correspondencia perteneciente al objetivo  $P$ ,  $n_i = (n_{ix}, n_{iy}, n_{iz}, 0)^T$  es el vector normal en  $p_i$ ,  $R$  es el parámetro de rotación y  $t$  el parámetro de traslación en la transformada  $M_F$ .

Cuando se realiza el registro en las nubes de puntos se obtienen la transformada  $M_{Fij}$  con los parámetros de rotación  $R_{ij}$  y traslación  $t_{ij}$  para cada uno de los nodos en el gráfico de pose, al transformar la parte de rotación en un cuaternión, esto se puede expresar como  $[\tau_{ij}^T, S_{ij}^T]^T$ . Para cada borde  $B_{ij} = [\tau_{Bij}^T, S_{Bij}^T]^T$  entre los nodos  $N_i$  y  $N_j$  el residual se divide en dos, el residual para la parte de rotación (ecuación 3.8), y el residual para la parte de traslación (ecuación 3.9). En cada borde  $B_{ij}$  su parte de rotación y traslación corresponden a  $S_{Bij}$  y  $\tau_{Bij}$  respectivamente y la función  $V(S)$  retorna la parte del vector del cuaternión  $S$ ,  $[sx, sy, sz]$ .

$$r_R(N_i, N_j, B_{ij}) = R(S_i)^T(\tau_i - \tau_j) - \tau_{Bij} \quad (3.8)$$

$$r_t(N_i, N_j, B_{ij}) = 2 \cdot V(S_i^{-1} S_j S_{Bij}^{-1}) \quad (3.9)$$

Donde  $r_R(N_i, N_j, B_{ij})$  corresponde a la parte residual de la rotación y  $r_t(N_i, N_j, B_{ij})$  a la parte residual de la traslación.

De esta manera, el residuo total entre la parte de rotación y traslación está dado por la ecuación 3.10.

$$r(N_i, N_j, B_{ij}) = w_R \|r_R(N_i, N_j, B_{ij})\|_2 + w_t \|r_t(N_i, N_j, B_{ij})\|_2 \quad (3.10)$$

Donde  $w_R$  y  $w_t$  son los pesos de las partes de rotación y traslación respectivamente.

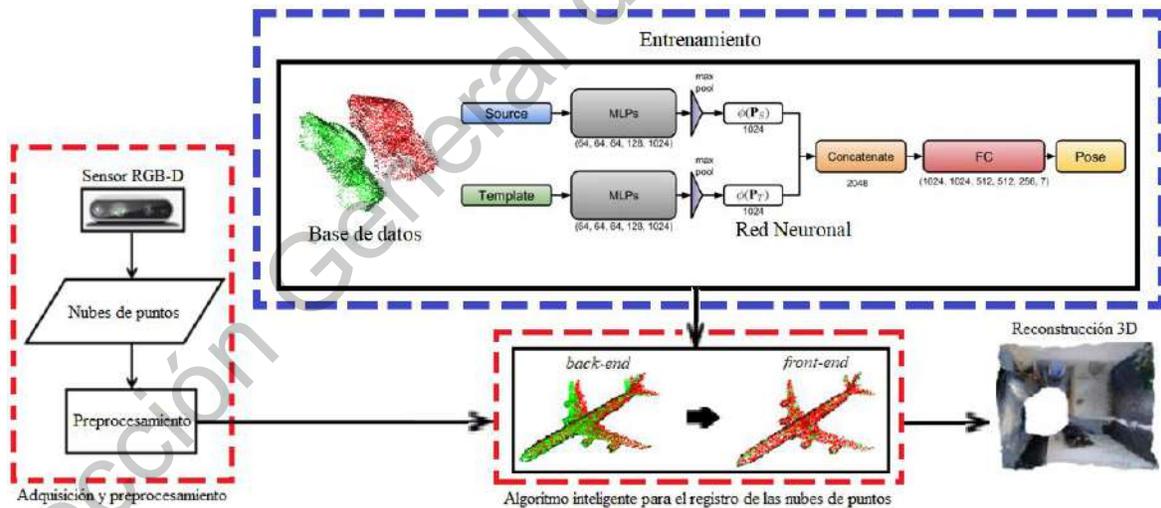
### 3.2 SISTEMA DE RECONSTRUCCIÓN 3D BASADO EN APRENDIZAJE PROFUNDO.

El modelo del sistema de reconstrucción 3D basado en algoritmos de aprendizaje profundo se muestra en la Fig. 3.5. El sistema está dividido en 2 fases. La primera fase está dedicada al entrenamiento de la arquitectura de la red neuronal utilizada para el registro de las nubes de puntos. En esta sección se detallan las partes que conforman la fase de entrenamiento: base de datos y la arquitectura de la red neuronal junto con el método que utiliza para el registro de las nubes de puntos.

En la segunda fase se describe la implementación del módulo de aprendizaje en el sistema de reconstrucción 3D. El sistema de reconstrucción 3D está constituido por 2 bloques. En el primero se realiza la adquisición y el preprocesamiento de las nubes de puntos, al igual que en el modelo basado en métodos tradicionales, en esta parte se utiliza un sensor RGB-D para capturar los datos de entrada.

En el segundo bloque se describe el algoritmo inteligente implementado para realizar el proceso del registro de las nubes de puntos. Un fenómeno común dentro de los métodos para el registro de nubes de puntos basados en aprendizaje profundo es que, en la mayoría de los casos, no son arquitecturas end-to-end, por lo que pueden llegar a confundirse los módulos tradicionales con los módulos de aprendizaje (Zhang et al., 2020).

Este bloque se divide en dos secciones: en el *back-end* se aplica el módulo de aprendizaje previamente entrenado, el objetivo principal de esta sección es la búsqueda inteligente de la transformación de inicialización o pre-alineación, en el *front-end* se realiza la transformación afín por medio del algoritmo ICP punto a plano, tomando como punto de partida la transformación entregada por el módulo de aprendizaje.



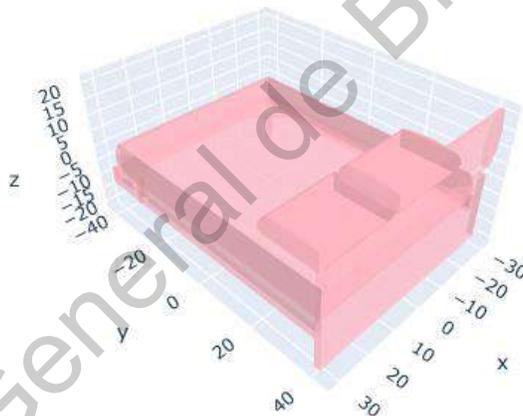
**Fig. 3.5.** Modelo del sistema de reconstrucción 3D basado en aprendizaje profundo.

### 3.2.1 Fase de entrenamiento.

La fase de entrenamiento está integrada por dos partes: una base de datos, que incluye los objetos para realizar el entrenamiento, y la arquitectura de la red neuronal que se desea entrenar.

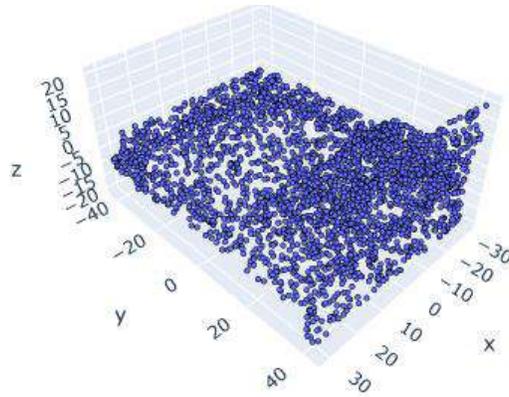
#### 3.2.1.1 Base de datos.

En esta tesis, para el entrenamiento de la arquitectura de la red neuronal implementada, se utilizó la base de datos de PRINCETON ModelNet40 (<https://modelnet.cs.princeton.edu/>). Esta base de datos cuenta con 40 diferentes categorías de objetos, 9843 para entrenamiento y 2468 para pruebas y validación. Todos estos archivos tienen extensión *.off* (*Object File Format*) que contienen mallas representadas por vértices y caras triangulares. En la Fig. 3.6 se muestra un objeto contenido en la base de datos.



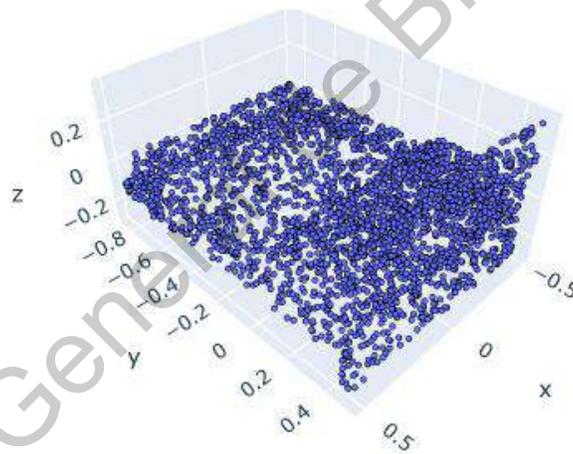
**Fig. 3.6.** Objeto extraído de la base de datos ModelNet40.

Para el registro de las nubes de puntos fue necesario realizar un muestreo de la superficie de todos estos objetos de manera uniforme, en la Fig. 3.7 se muestra la nube de puntos obtenida.



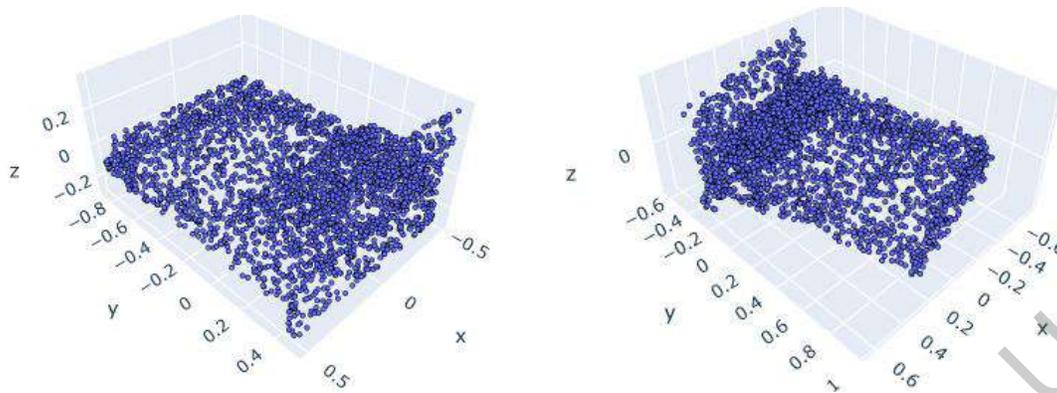
**Fig. 3.7.** Nube de puntos obtenida de ModelNet40.

Una vez obtenidas las nubes de puntos de cada uno de los objetos contenidos en la base de datos, se procedió a normalizarlos. Todos los datos fueron normalizados dentro de una caja unitaria. En la Fig. 3.8 se muestra el resultado de la normalización para un objeto.



**Fig. 3.8.** Nube de puntos normalizada.

En el registro de las nubes de puntos se requiere una nube de puntos objetivo y una nube de puntos fuente, esta última es a la que se le desea aplicar una transformación de cuerpo rígido para alinearla con la nube de puntos objetivo. Para generar esta nube de puntos fuente se le aplicó una rotación aleatoria a cada uno de los objetos de la base de datos en un rango de  $-45^\circ$  a  $45^\circ$ . En la Fig. 3.9 se muestran ambas nubes de puntos.



**Fig. 3.9.** Nube de puntos objetivo (Izq.), fuente (Der.).

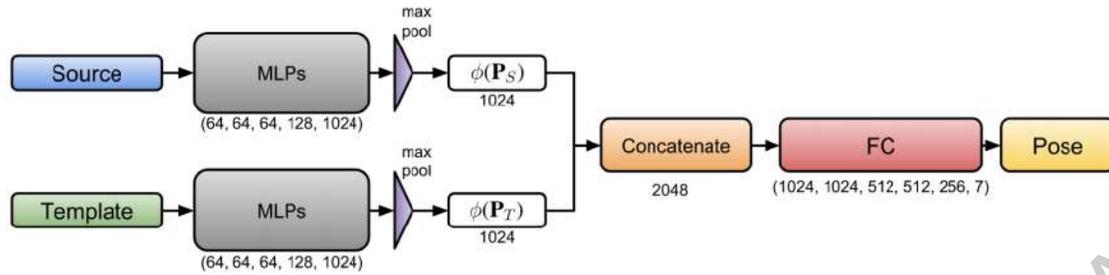
A todas las nubes de puntos se les aplica un muestreo para fijar el número de puntos contenido en cada conjunto en 1024. Por último, se realiza la conversión a tensores para poder ser ingresados a la red neuronal.

### 3.2.1.2 Arquitectura de la red neuronal.

En la Fig. 3.10 se muestra la arquitectura implementada en este trabajo. La arquitectura consta de dos ramificaciones en una estructura siamés basada en PointNet. Pertenece al grupo de los métodos libres de correspondencias.

Ambas entradas pasan por 5 multicapas de perceptrones con medidas de 64 en las tres primeras, 128 en la cuarta y 1024 en la quinta multicapa, mediante *max-pooling* se obtienen las características globales de ambas entradas, ambas características son concatenadas y se les aplican 6 capas *fully connected*, las primeras dos de dimensión 1024, la tercera y cuarta de 512, una quinta de 256 y por último una salida de dimensión 7 donde los primeros 3 valores representan el vector de traslación y los últimos 4 valores el cuaternión de rotación.

Por medio de estos parámetros se obtiene la transformación de cuerpo rígido.



**Fig. 3.10.** Arquitectura de la red neuronal PCRNet (Sarode et al., 2019).

Para el entrenamiento de la red neuronal se utilizó Google Colaboratory. Los hiperparámetros del entrenamiento de la red neuronal se muestran a continuación:

- 200 - 300 épocas de entrenamiento. (Sarode et al., 2019 / Wang et al., 2019).
- Optimizador Adam (Sarode et al., 2019 / Wang et al., 2019).
- Batch 20 / 32.
- Ratio de aprendizaje 0.001 con tasa de caída exponencial de 0.7 cada  $3 \times 10^{-6}$  pasos. (Sarode et al., 2019, Wang et al., 2019).

### 3.2.2 Fase de reconstrucción.

La fase de reconstrucción consta de dos partes: la adquisición y el preprocesamiento de las nubes de puntos, y el algoritmo basado en aprendizaje profundo para el registro de las nubes de puntos.

#### 3.2.2.1 Adquisición y preprocesamiento de los datos.

Para la adquisición de las nubes de puntos se siguió el mismo procedimiento que en el sistema de reconstrucción 3D basado en métodos tradicionales. Una vez que se tienen capturadas las nubes de puntos se realiza el preprocesamiento para la entrada a la red neuronal.

En el preprocesamiento se realiza un muestreo para ajustar el número de puntos de acuerdo a la entrada de la red neuronal, estos datos son convertidos a arreglos para posteriormente ser pasados a tensores para poder ser ingresados al algoritmo inteligente.

### 3.2.2.2 Algoritmo inteligente.

El algoritmo desarrollado para el registro de nubes de puntos basado en aprendizaje profundo se compone de dos módulos: el módulo de aprendizaje y el módulo de refinamiento, esto siguiendo la literatura descrita en (Zhang et al., 2020) y (Wang & Solomon, 2019) para los métodos basados en aprendizaje profundo.

La arquitectura utilizada pertenece al grupo de los métodos libres de correspondencias, también conocidos como métodos globales, por este motivo se utiliza en la parte de inicialización en el proceso del registro de las nubes de puntos.

Una vez que se realiza el entrenamiento de la red neuronal y que se obtiene el mejor modelo, se implementa en el sistema de reconstrucción como la parte del *back-end*. Esta sección tiene como objetivo entregar una inicialización inteligente para la parte del *front-end*, donde por medio del algoritmo ICP punto a plano se realiza la transformación afín o refinamiento de la transformada de cuerpo rígido.

En la Fig. 3.11 se muestra el diagrama del algoritmo inteligente. En el back-end se implementa el modelo entrenado de la red neuronal y en el front-end el algoritmo ICP punto a plano. Este modelo fue inspirado por arquitecturas como (Wang & Solomon, 2019) que implementan ICP como refinamiento para la arquitectura que proponen en DCP.

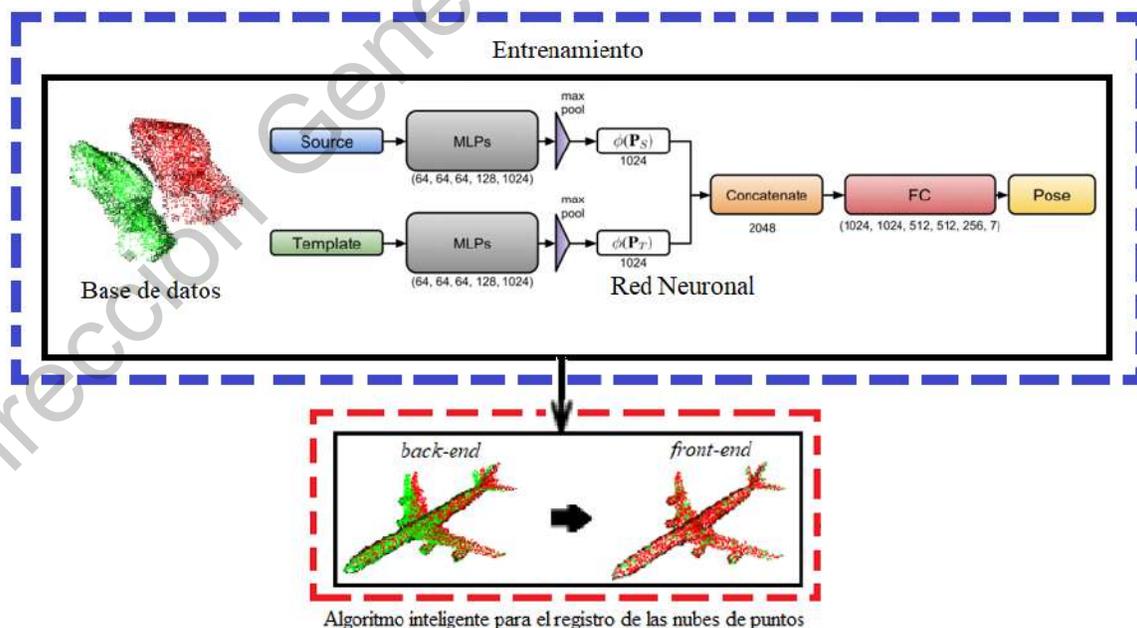


Fig. 3.11. Diagrama del algoritmo inteligente.

## 4. RESULTADOS

Los resultados presentados en esta investigación están divididos en dos partes. En la primera parte se describen los resultados del sistema de reconstrucción 3D basado en métodos tradicionales, en la segunda parte los resultados obtenidos por el sistema de reconstrucción 3D basado en métodos de aprendizaje profundo. Finalmente, se realiza la comparación entre el modelo propuesto utilizando algoritmos de aprendizaje profundo y el modelo basado en métodos tradicionales.

Ambos sistemas de reconstrucción 3D fueron implementados en una computadora personal hp Envy TS con procesador AMD A10-5745M a 2.10 GHz y 6 GB de RAM.

Los requerimientos principales del sistema de entrenamiento son: Python == 3.6, Cuda == 10, tensorflow == 1.14, transforms3d == 0.3.1, h5py == 2.9.0, PyTorch == 1.8.1.

Con el sensor de profundidad Intel RealSense D435 se capturaron distintas instancias de una escena interior. En la Fig. 4.1 y 4.2 se muestra un ejemplo de un par de nubes de puntos capturadas, ambas muestras son fragmentos de la misma instancia tomadas de diferentes ángulos, la muestra de la izquierda representa la nube de puntos objetivo y la de la derecha la fuente.



**Fig. 4.1.** Nubes de puntos obtenidas con el sensor de profundidad. Objetivo (a), fuente (b).



**Fig. 4.2.** Nubes de puntos capturadas con el sensor de profundidad con diferente ángulo y distancia.

Para su visualización durante la experimentación, ambas nubes de puntos son desplegadas con su textura normal ya que al ser distinguidas mediante colores los detalles eran menos visibles, como se muestra en la Fig. 4.3.



**Fig. 4.3.** Nubes de puntos de entrada. Textura normal (a), distinción de color (b).

#### **4.1 SISTEMA DE RECONSTRUCCIÓN BASADO EN MÉTODOS TRADICIONALES.**

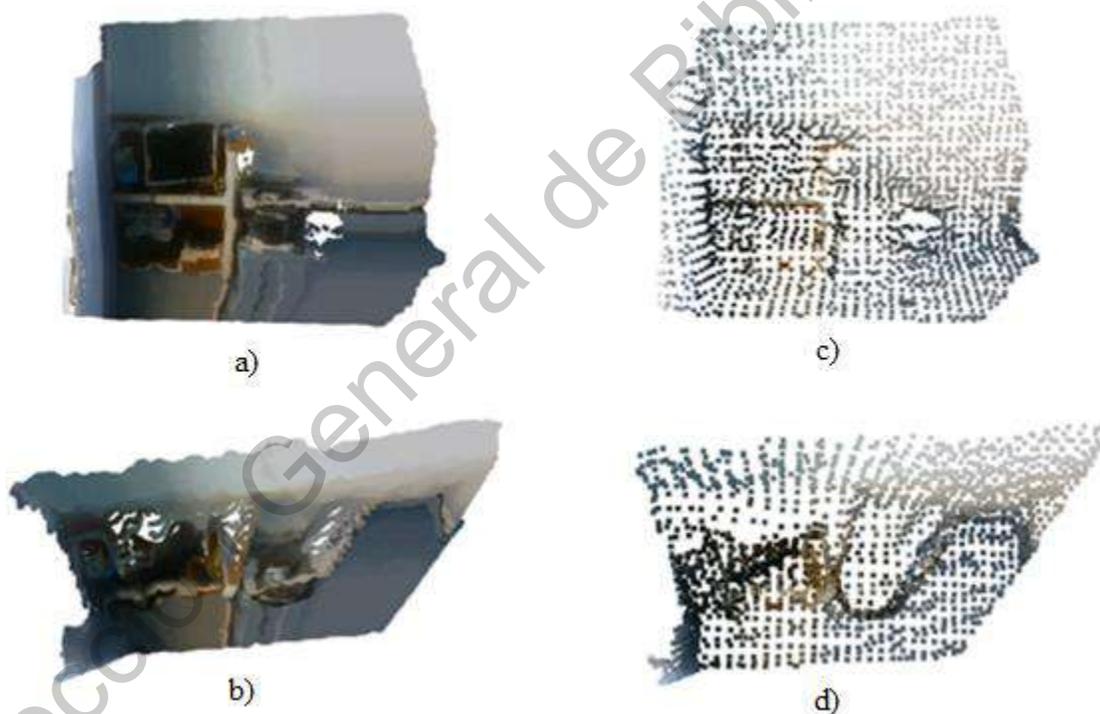
En esta sección se presentan detalladamente los resultados obtenidos para la reconstrucción de una escena interior utilizando algoritmos basados en métodos tradicionales. Primero se describe la fase del preprocesamiento, después la comparativa entre distintos métodos tradicionales y el modelo de comparación de esta tesis, finalmente se muestran los resultados de la reconstrucción obtenida con el modelo presentado en esta sección.

#### 4.1.1 Adquisición y preprocesamiento.

Una vez que se tienen capturadas las nubes de puntos se procede a realizar el preprocesamiento para su entrada al algoritmo del registro de las nubes de puntos.

Lo primero que se realiza es un *down sampling* para reducir el número de puntos en cada uno de los conjuntos. Las nubes de puntos obtenidas por el sensor de profundidad Intel RealSense D435 contienen alrededor de 63,000 puntos. Trabajar con esta cantidad de puntos haría que el proceso fuera muy lento y poco eficiente. Por este motivo se realizó la reducción de puntos para optimizar el proceso de registro.

El número de puntos establecido fue de 2048 puntos. En la Fig. 4.4 se muestra un ejemplo de un par de nubes de puntos capturadas por el sensor de profundidad y las mismas nubes de puntos una vez realizado el *down sampling*.



**Fig. 4.4.** Muestreo de nubes de puntos. (63,074 puntos a), b). 2048 puntos c), d) respectivamente).

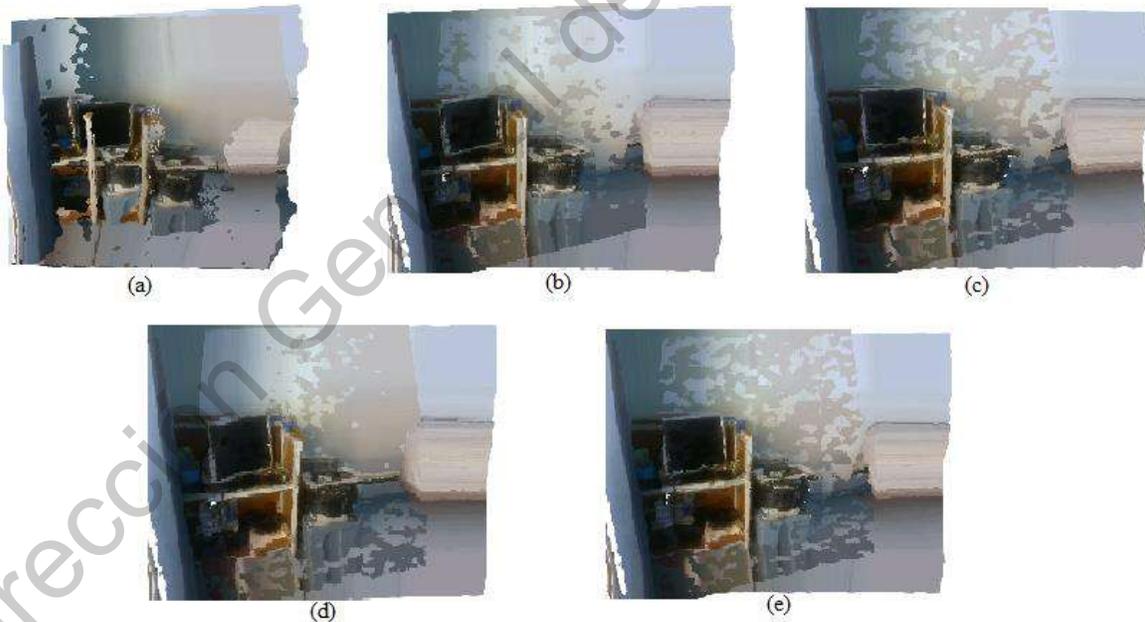
Una vez realizado el muestreo para reducir la cantidad de puntos, se realiza la normalización de cada nube de puntos dentro de una caja unitaria.

#### 4.1.2 Métodos tradicionales.

Como parte de las pruebas dentro del proceso de registro se compararon los algoritmos RANSAC, ICP punto a plano, ICP punto a punto y el algoritmo FGR en la alineación de un par de nubes de puntos para determinar su desempeño.

Para evaluar el desempeño de los algoritmos se tomaron en cuenta tres métricas: *fitness score* que representa la superficie de superposición entre nubes de puntos, el error cuadrático medio (RMSE) y el tiempo de procesamiento. En la Fig. 4.5 se puede observar la alineación obtenida por cada algoritmo.

El algoritmo de registro global basado en RANSAC obtiene un valor RMSE de 0.04466 con un *fitness score* de 0.891. Al aplicar el algoritmo ICP punto a punto se obtiene un valor RMSE de 0.011 con *fitness score* de 0.573. por su parte, el algoritmo FGR obtiene un valor RMSE de 0.01513 con un *fitness score* de 0.184. Por último, el algoritmo ICP punto a plano obtiene un valor RMSE de 0.01496 con un *fitness score* de 0.328. En la Tabla 4.1 se muestran los resultados de las métricas aplicadas para cada algoritmo.



**Fig. 4.5.** Alineación obtenida para dos nubes de puntos. (a) Entrada, (b)RANSAC, (c)ICP punto a punto, (d)FGR, (e)ICP punto a plano.

**Tabla 4.1.** Resultados obtenidos en el registro de nubes de puntos.

<b>Algoritmo</b>	<b>RMSE</b>	<b><i>fitness score</i></b>	<b>Procesamiento</b>
RANSAC	0.05366414	<b>0.8955023</b>	<b>0.471seg</b>
ICP punto a punto	0.04651124	0.5731295	1.159seg
FGR	0.01513006	0.1847563	0.692seg
ICP punto a plano	<b>0.01496852</b>	0.3283060	0.868seg

Como se muestra en la Tabla 4.1, el mejor rendimiento en cuanto al valor RMSE lo presentan los algoritmos FGR e ICP punto a plano con un valor de 0.01513006 y 0.01496852 respectivamente. Aunque el menor tiempo de procesamiento y mayor área de superposición lo tiene el algoritmo basado en RANSAC, su desempeño en cuanto a precisión es bajo pues su valor RMSE es el más elevado (0.05366414). En términos de precisión se puede observar que ICP punto a plano tiene el mejor desempeño seguido del algoritmo FGR, siendo el tiempo de procesamiento el único punto donde éste lo supera.

#### **4.1.3 Inicialización y gráfico de pose.**

Con base en los resultados obtenidos por la sección anterior, se procedió a realizar el proceso del registro de las nubes de puntos y la representación de los nodos y bordes que conforman el gráfico de pose. Como parte de las pruebas, se realizó la implementación del algoritmo ICP punto a plano en ambas partes del proceso de registro, esto es, tanto en la pre-alineación como en la transformación afín.

Por otra parte, se realizó la implementación del algoritmo FGR como inicialización del algoritmo ICP punto a plano debido a que obtiene resultados casi tan buenos como éste, pero con un menor tiempo de convergencia. En la pre-alineación sólo se requiere una buena estimación por lo que el número de iteraciones del algoritmo FGR es reducido a 10 para acelerar el proceso. Este proceso fue aplicado sobre el mismo par de nubes de puntos analizadas previamente. En la Fig. 4.6 se muestra el registro de nubes de puntos por ambos algoritmos.



**Fig. 4.6.** Proceso de registro de las nubes de puntos. ICP punto-plano (a), FGR-ICP punto-plano (b).

Una de las mejoras a simple vista fue la precisión en la alineación de los contornos ya que en el resultado obtenido por el algoritmo conformado solamente por ICP punto a plano se puede observar un pequeño desalineamiento mientras que en el algoritmo inicializado con FGR se disminuye notoriamente, esto se ve reflejado en los resultados de las métricas mostrados en la Tabla 4.2 incluyendo el valor residual obtenido al realizar la optimización global del gráfico de pose por medio del algoritmo Levenberg Marquardt.

**Tabla 4.2.** Resultados obtenidos registro de puntos con optimización global.

Algoritmo	RMSE	<i>fitness score</i>	Procesamiento	Residual
ICP punto a plano	0.0323	0.8287	0.595999seg	$1.2272 e^{-30}$
FGR-ICP punto a plano	<b>0.0322</b>	0.8287	<b>0.539982seg</b>	<b><math>3.9004e^{-31}</math></b>

#### 4.1.4 Reconstrucción 3D.

Para validar el desempeño del sistema de reconstrucción 3D basado en métodos tradicionales se realizó el escaneo de dos escenas interiores. La primera escena consta de un total de 19 instancias y la segunda de 34 instancias adquiridas por el sensor de profundidad utilizado.

En la Fig. 4.7 se muestran los resultados finales obtenidos para la primera escena teniendo como resultado un valor RMSE de 0.0310 y un *fitness score* de 0.9422 con un residual final de 0.000343. y en la Fig. 4.8 se muestran los resultados para la segunda escena obteniendo un RMSE de 0.0275 y un *fitness score* de 0.9215 con un residual final de 0.0000985.

En la Tabla 4.3 se muestran los resultados finales obtenidos por el sistema de reconstrucción 3D utilizando algoritmos basados en métodos tradicionales. Se muestran los resultados obtenidos por el modelo presentado en (Yan et al., 2019) y los resultados del modelo propuesto con inicialización por el algoritmo FGR.



**Fig. 4.7.** Reconstrucción 3D con métodos tradicionales, vista superior escena 1.



**Fig. 4.8.** Reconstrucción 3D con métodos tradicionales, vista superior escena 2.

Algo importante de mencionar sobre el modelo de reconstrucción realizado es que su aplicación podría ser en sistemas que requieran poca resolución de mapeo como sistemas de reconocimiento y exploración o en aplicaciones donde los detalles no sean tan importantes a gran escala.

Como se puede observar en la Fig. 4.9 (a) los detalles más grandes se mantienen un poco asemejados a la realidad en objetos de gran tamaño. Sin embargo, al realizar un acercamiento como se observa en la Fig. 4.9 (b) se ve claramente que estos detalles se van perdiendo a medida que se hace un acercamiento a los objetos más pequeños.



Fig. 4.9. Nivel de detalles en el sistema de reconstrucción 3D.

Esto se puede modificar cambiando la configuración de la cámara de profundidad. El trabajo realizado en esta tesis se centra en la reconstrucción de interiores por lo cual se trabajó con un rango de visión activo de 0.15 a 2.3 metros de distancia.

Tabla 4.3. Resultados finales obtenidos por el sistema de reconstrucción 3D.

Reconstrucción	Modelo	RMSE	<i>fitness score</i>	Procesamiento	Residual
Escena 1	Propuesto	<b>0.0310</b>	0.9422	118.55655seg	<b><math>3.43 e^{-04}</math></b>
Escena 1	Yan et al.	0.0324	0.9422	<b>117.91059seg</b>	$3.74 e^{-04}$
Escena 2	Propuesto	<b>0.0275</b>	<b>0.9215</b>	<b>286.54123seg</b>	<b><math>9.85e^{-05}</math></b>
Escena 2	Yan et al.	0.0277	0.9045	287.21839seg	$1.13e^{-04}$

## 4.2 SISTEMA DE RECONSTRUCCIÓN BASADO EN APRENDIZAJE PROFUNDO.

En esta sección se presentan detalladamente los resultados obtenidos para la reconstrucción de una escena interior utilizando algoritmos basados en aprendizaje profundo. La sección se divide en cuatro partes. Primero se describe la fase de entrenamiento, después el refinamiento de la red neuronal y la comparativa entre los algoritmos basados en métodos tradicionales, en la tercera parte se describe el preprocesamiento de los datos reales para la entrada a la red neuronal y finalmente, en la cuarta parte se realiza la implementación del algoritmo inteligente en el sistema de reconstrucción 3D.

### 4.2.1 Fase de entrenamiento.

Los autores de la arquitectura implementada en este trabajo (PCRNet) tienen disponible un modelo pre entrenado de la red neuronal, como primer paso se implementó la red con este modelo para familiarizarse con hiperparámetros y observar su funcionamiento.

La función de pérdida utilizada en el entrenamiento de la red para el registro de las nubes de puntos minimiza la distancia entre los puntos de correspondencia de la nube de puntos fuente y objetivo. Esta función es denominada *Earth Mover Distance* (EMD) y tiene la forma mostrada en la ecuación 4.1.

$$EMD(P_S^{est} P_T) = \min_{\varphi: P_S^{est} \rightarrow P_T} \frac{1}{|P_S^{est}|} \sum_{x \in P_S^{est}} \|x - \varphi(x)\|_2 \quad (4.1)$$

Donde  $P_T$  es la nube de puntos objetivo,  $P_S^{est}$  la nube de puntos fuente. Esta función encuentra una biyección  $\varphi$  y minimiza la distancia entre los puntos de correspondencia  $x$  basados en  $\varphi$ .

Al implementar el modelo pre entrenado se obtuvieron como resultados un *Test Loss* promedio de 0.036081979712, con un error promedio en la rotación de  $14.8360364521^\circ$  y un error promedio en la traslación de 0.00592406694 unidades, como se muestra en la Fig. 4.10. Este modelo, según la literatura, fue entrenado durante 300 épocas, utilizando un flujo de aprendizaje de  $10^{-3}$  con una tasa de caída exponencial de 0.7 después de cada  $3 \times 10^6$  pasos

y un *batch size* de 32. El optimizador utilizado fue Adam. Esta arquitectura fue implementada en una GPU NVIDIA GeForce GTX 1070 GPU con un procesador Intel Core i7 a 4.0 GHz.

```
[ ] ! python test_pcrnet.py

/usr/local/lib/python3.7/dist-packages/torch/utils/data/dataloader.py:477: UserWarning: This DataLoader will
  cpuset_checked))
running on the GPU
cuda:0
100% 124/124 [00:10<00:00, 11.34it/s]
Test Loss: 0.03608197971217094, Rotation Error: 14.836036452154577 & Translation Error: 0.005924066942445218
```

**Fig. 4.10.** Resultados obtenidos por el modelo pre entrenado en (Sarode et al., 2019).

Para realizar el entrenamiento de la arquitectura de la red neuronal implementada se utilizó Google Colaboratory. La red neuronal fue entrenada por 250 épocas con un *batch size* de 20, al igual que en los trabajos de (Sarode et al., 2019) y (Wang & Solomon, 2019) se utilizó un flujo de aprendizaje de  $10^{-3}$  con una tasa de caída exponencial de 0.7 después de cada  $3 \times 10^6$  pasos y el optimizador Adam.

El modelo entrenado obtuvo un *Test loss* de 0.0342955613, un error promedio de rotación de  $11.8934892723^\circ$  y un error promedio de traslación de 0.0337615849 unidades. En la Fig. 4.11 se muestran los resultados obtenidos por el modelo entrenado y en la Tabla 4.4 la comparación de ambos modelos. En la Fig. 4.12 se muestra un ejemplo del resultado del registro de puntos obtenido por ambos modelos.

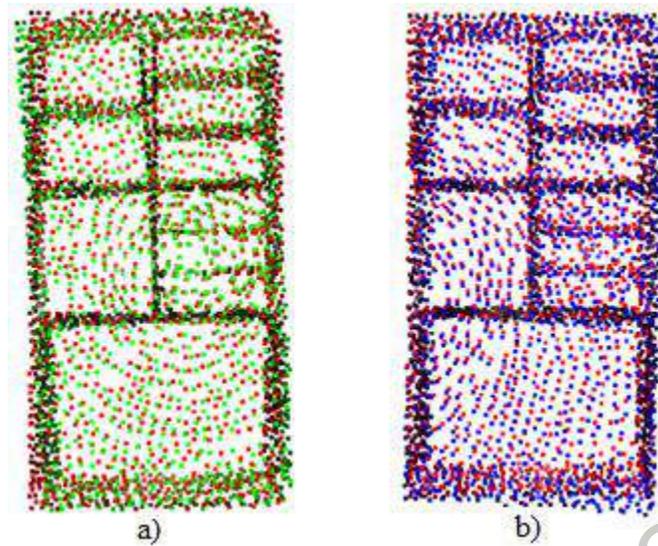
```
[ ] ! python test_PCR.py

/usr/local/lib/python3.7/dist-packages/torch/utils/data/dataloader.py:477: UserWarning: This DataLoader will
  cpuset_checked))
running on the GPU
cuda:0
100% 124/124 [00:12<00:00, 9.59it/s]
Test Loss: 0.034295561362899116, Rotation Error: 11.893489272355698 & Translation Error: 0.03376158492422789
```

**Fig. 4.11.** Resultados obtenidos por el modelo entrenado.

**Tabla 4.4.** Comparación de resultados del modelo pre entrenado y el modelo entrenado.

Parámetro	Modelo pre entrenado	Modelo entrenado
<i>EMD</i>	0.036081979712	<b>0.0342955613</b>
Error Rotación	14.8360364521°	<b>11.8934892723°</b>
Error Traslación	<b>0.00592406694 u</b>	0.0337615849 u



**Fig. 4.12.** Resultado del registro de puntos. Entrenado (a), pre entrenado (b).

Como se puede observar en la Tabla 4.4, el modelo entrenado obtiene mejores resultados en dos parámetros, el error de rotación y el valor de la función de pérdida *EMD*. Con base en estos resultados se tomó este nuevo modelo para la implementación al algoritmo inteligente de registro de nubes de puntos.

#### 4.2.2 Refinamiento de la red neuronal.

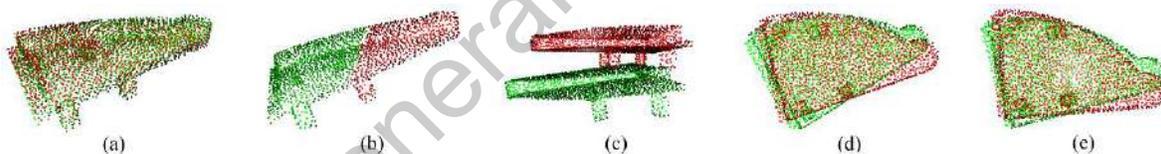
Como se menciona en la literatura revisada en (Zhang et al., 2020), los métodos de aprendizaje para el registro de puntos generalmente no son arquitecturas *end-to-end*, siendo que se tiene un módulo de aprendizaje y un módulo de registro tradicional como en el trabajo de DCP (Wang & Solomon, 2019). Esto sirvió de inspiración para el desarrollo del algoritmo inteligente integrado por el módulo de aprendizaje; conformado por el modelo entrenado de la arquitectura neuronal implementada, y el módulo de refinamiento; integrado por el algoritmo ICP punto a plano.

La red neuronal entrega cinco parámetros como salida de los cuales solamente nos interesan tres: la matriz de rotación; con dimensión  $3 \times 3$ , el vector de rotación; con dimensión  $1 \times 3$  y principalmente la transformación de cuerpo rígido; compuesta por los dos parámetros anteriores teniendo una dimensión de  $4 \times 4$ . Esta transformación de cuerpo rígido es utilizada

como inicialización inteligente para la parte de la transformación afín, conformada por el algoritmo ICP punto a plano en el proceso de registro, similar a lo realizado en DCP (Wang & Solomon, 2019).

Como parte de las pruebas se implementaron los algoritmos ICP punto a plano, FGR, el modelo del sistema de reconstrucción basados en métodos tradicionales FGR-ICP, la arquitectura PCRNet y el algoritmo inteligente integrado por el módulo de aprendizaje y el módulo de refinamiento PCRNet-ICP punto a plano. Se realizaron 2468 experimentos para cada algoritmo. Los objetos de prueba fueron seleccionados aleatoriamente del conjunto de prueba de la base de datos ModelNet40. Se compararon los valores RMSE, FITNESS SCORE y el número de correspondencias en la alineación obtenida.

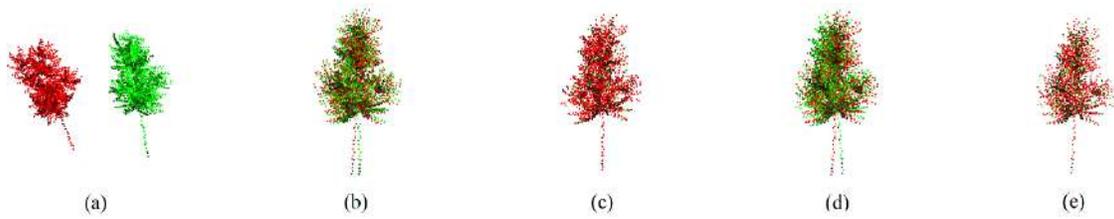
En las Fig. 4.13, 4.14, 4.15, 4.16 y 4.17 se muestran los resultados cualitativos del registro de las nubes de puntos en cinco objetos diferentes para los algoritmos FGR, ICP punto a plano, el modelo de reconstrucción integrado por FGR-ICP punto a plano, la arquitectura PCRNet y el algoritmo inteligente integrado por el módulo de aprendizaje y el módulo de refinamiento PCRNet-ICP punto a plano y en las Tablas 4.5, 4.6, 4.7, 4.8 y 4.9 se muestran los resultados cuantitativos del registro para cada par de nubes de puntos.



**Fig. 4.13.** Resultados del registro de nubes de puntos en objeto 1. (a) FGR, (b) FGR-ICP, (c) ICP punto a plano, (d) PCRNet, (e) PCRNet-ICP.

**Tabla 4.5.** Comparación de resultados objeto 1.

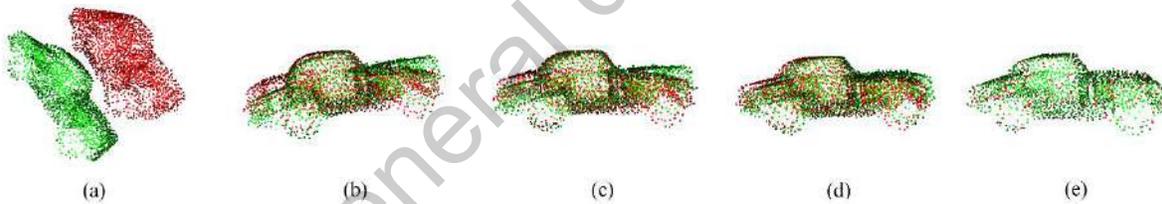
Algoritmo	RMSE	FITNESS	Correspondencias
FGR	0.0140256	0.4038086	827
FGR-ICP	0.0003623	1	2048
ICP punto a plano	0	0	0
PCRNet	0.015145	0.1821	373
PCRNet-ICP	0.013744	0.4995	1023



**Fig. 4.14.** Resultados del registro de nubes de puntos en objeto 2. (a) FGR, (b) FGR-ICP, (c) ICP punto a plano, (d) PCRNet, (e) PCRNet-ICP.

**Tabla 4.6.** Comparación de resultados objeto 2.

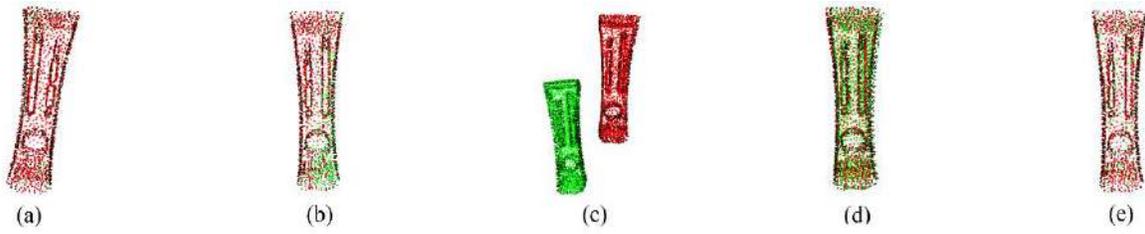
Algoritmo	RMSE	FITNESS	Ncorrespondencias
FGR	0	0	0
FGR-ICP	0.0072776	0.1425	292
ICP punto a plano	0.00000003649	1	2048
PCRNet	0.014635	0.35058	718
PCRNet-ICP	0.00000016216	1	2048



**Fig. 4.15.** Resultados del registro de nubes de puntos en objeto 3. (a) FGR, (b) FGR-ICP, (c) ICP punto a plano, (d) PCRNet, (e) PCRNet-ICP.

**Tabla 4.7.** Comparación de resultados objeto 3.

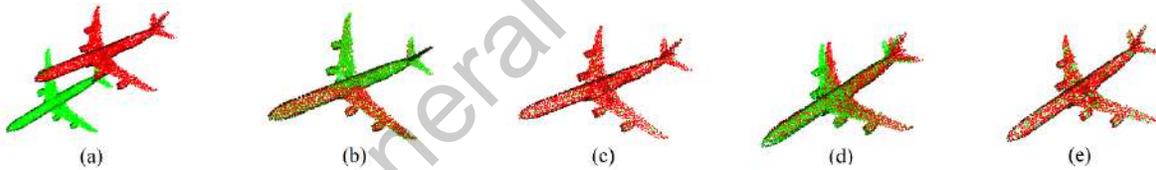
Algoritmo	RMSE	FITNESS	Ncorrespondencias
FGR	0	0	0
FGR-ICP	0.008296	0.0239	49
ICP punto a plano	0.014857	0.2045	419
PCRNet	0.0158483	0.15576	319
PCRNet-ICP	0.00000023135	1	2048



**Fig. 4.16.** Resultados del registro de nubes de puntos en objeto 4. (a) FGR, (b) FGR-ICP, (c) ICP punto a plano, (d) PCRNet, (e) PCRNet-ICP.

**Tabla 4.8.** Comparación de resultados objeto 4.

Algoritmo	RMSE	FITNESS	Ncorrespondencias
FGR	0.00000005341	1	2048
FGR-ICP	0.000029017	1	2048
ICP punto a plano	0	0	0
PCRNet	0.015596	0.274414	562
PCRNet-ICP	0.00000008847	1	2048



**Fig. 4.17.** Resultados del registro de nubes de puntos en objeto 5. (a) FGR, (b) FGR-ICP, (c) ICP punto a plano, (d) PCRNet, (e) PCRNet-ICP.

**Tabla 4.9.** Comparación de resultados objeto 5.

Algoritmo	RMSE	FITNESS	Ncorrespondencias
FGR	0	0	0
FGR-ICP	0.0068405	0.39697	813
ICP punto a plano	0.00000003784	1	2048
PCRNet	0.507812	0.507812	1040
PCRNet-ICP	0.00000021175	1	2048

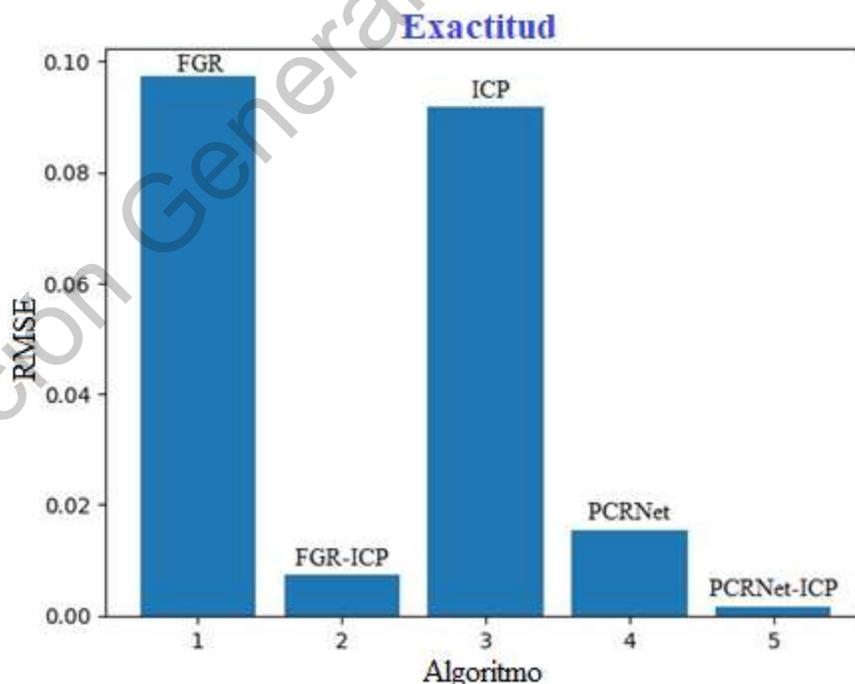
El número total de objetos tomados para estas pruebas fue de 2468, en la Fig. 4.18 se muestra la media de los resultados cuantitativos totales obtenidos por cada algoritmo, en la Fig. 4.19 se muestra la gráfica del valor RMSE total obtenido por cada algoritmo y en la Fig. 4.20 el valor FITNESS SCORE. Para mostrar los datos de manera cualitativa se tomaron aleatoriamente 100 de los objetos y se graficaron los parámetros RMSE y FITNESS SCORE. En la Fig. 4.21 y 4.22 se muestra los resultados de cada parámetro respectivamente.

```

100% 2468/2468 [11:20<00:00, 3.63it/s]
FGR
0.09554325160561043
0.7457300824980282
FGR+ICP
0.007410888621170296
0.11437909143284036
ICP
0.09041233583593705
0.8036725555865073
PCRNET
0.015816343619084676
0.33109010173976905
PCRNET+ICP
0.0012743197830950867
0.9582153518157415

```

**Fig. 4.18.** Media de los resultados cuantitativos totales obtenidos por cada algoritmo (2468 experimentos).



**Fig. 4.19.** Media del valor RMSE obtenido por cada algoritmo (2468 experimentos).

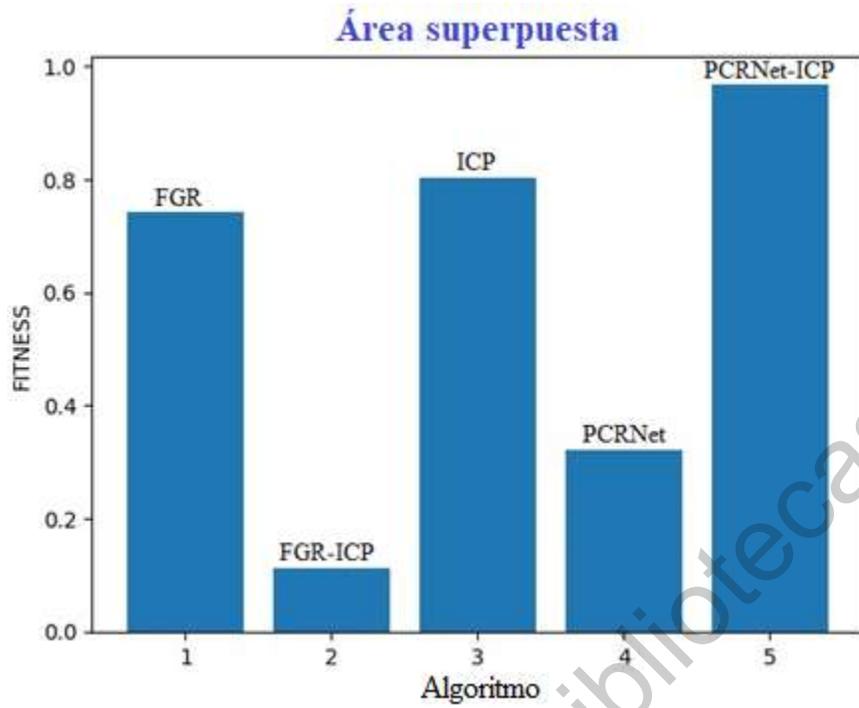


Fig. 4.20. Media del valor FITNESS SCORE obtenido por cada algoritmo (2468 experimentos).

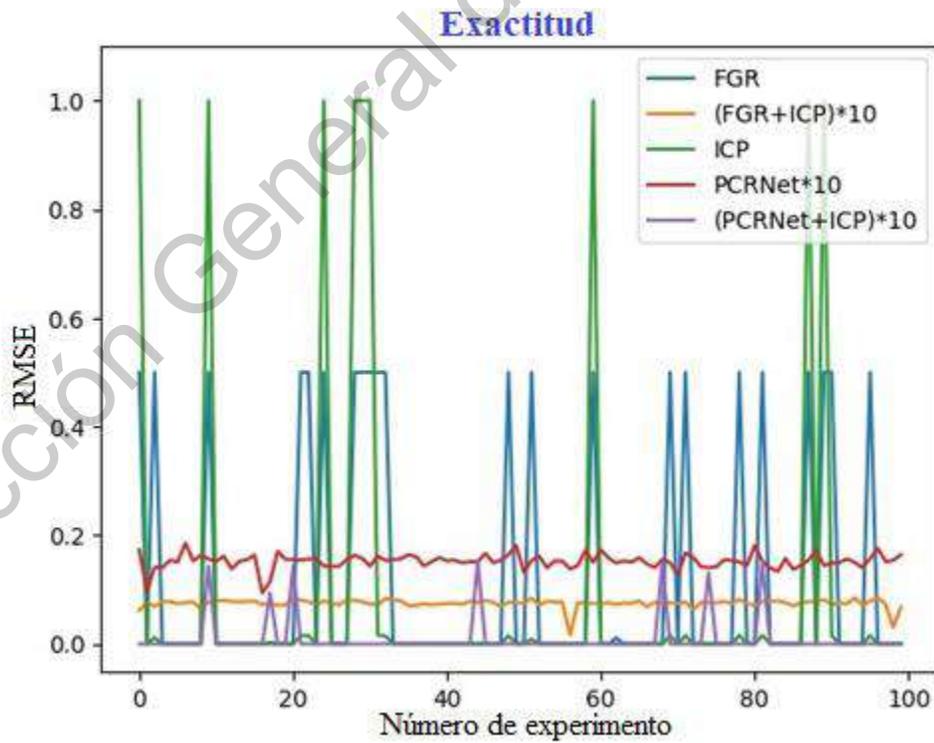
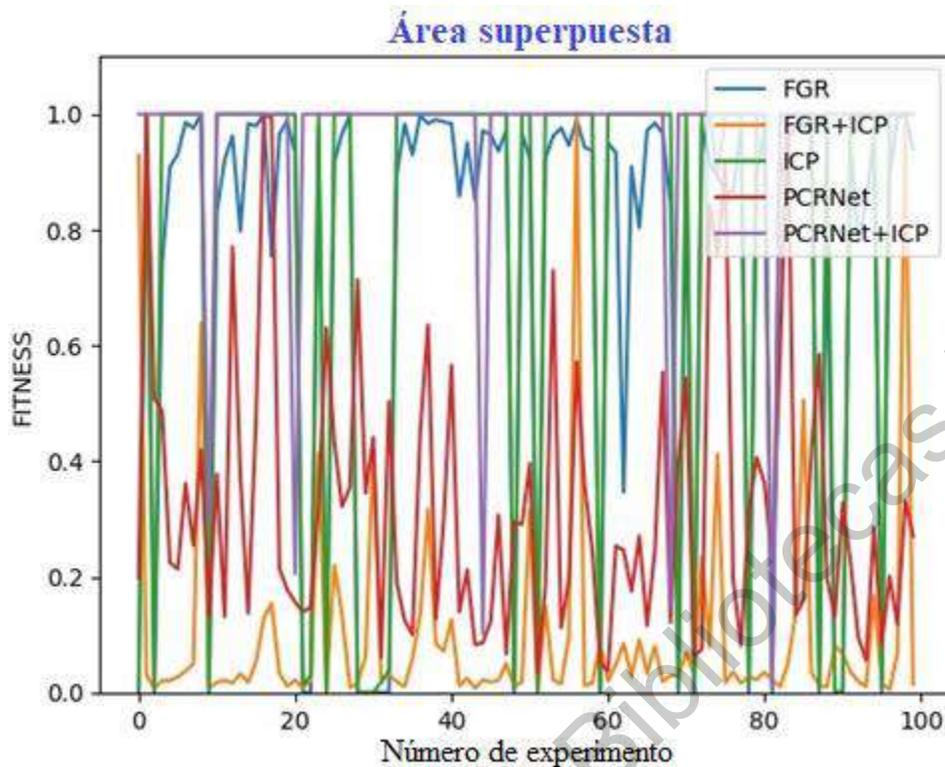


Fig. 4.21. Resultados del valor RMSE obtenido por cada algoritmo en 100 objetos.



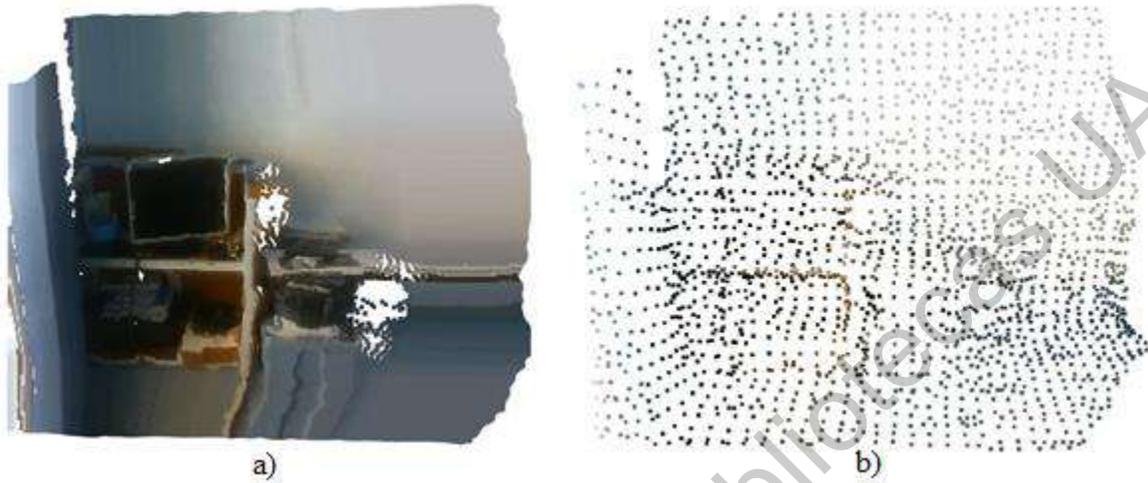
**Fig. 4.22.** Resultados del valor FITNESS SCORE obtenido por cada algoritmo en 100 objetos.

Como se puede observar, tanto en los resultados cuantitativos como en los cualitativos, el algoritmo conformado por el módulo de aprendizaje y el módulo de refinamiento obtiene los mejores resultados teniendo en promedio un valor RMSE de 0.00127 y un área de superposición del 95.82%, mejorando al algoritmo basado en métodos tradicionales propuesto en la sección anterior.

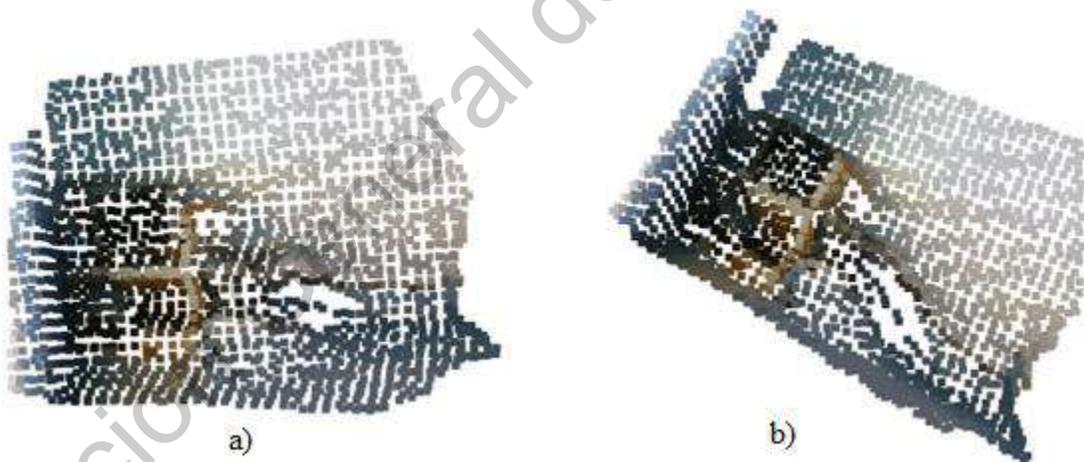
### 4.2.3 Preprocesamiento de los datos reales.

Una vez que se tiene el algoritmo inteligente para el registro de las nubes de puntos, se procede a realizar el preprocesamiento de las nubes de puntos obtenidas con el sensor de profundidad. El primer paso fue aplicar un *down sampling* en cada conjunto. La dimensión de entrada de la arquitectura de la red neuronal es de 2048 puntos, por lo que el número de puntos contenido dentro de cada nube de puntos fue establecido en esta cantidad. En la Fig. 4.23 (a) se muestra un ejemplo de una nube de puntos original y en la Fig. 4.23 (b) la misma

nube de puntos una vez hecho el *down sampling*. Debido a que puede ser un poco difícil visualizar la nube de puntos con *down sampling* en la Fig. 4.24 se realiza un zoom para tener una mejor perspectiva desde una vista frontal y una vista angular de la nube de puntos.



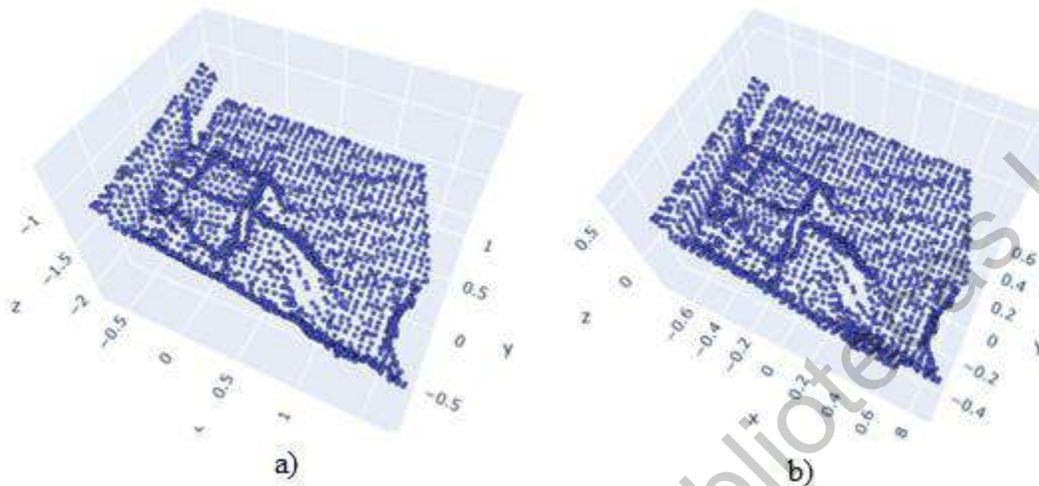
**Fig. 4.23.** Nube de puntos original (a), nube de puntos con *down sampling* (b).



**Fig. 4.24.** Vista con zoom de la parte frontal (a) y angular (b) de la nube de puntos con *down sampling*.

El siguiente paso fue convertir las nubes de puntos en arreglos mediante la biblioteca Numpy y la normalización de cada arreglo. Los datos fueron normalizados dentro de una caja unitaria. En la Fig. 4.25 se muestra la nube de puntos una vez convertida en arreglo (a) y

después de ser normalizada (b). Estos arreglos tienen la forma  $(n, d)$  donde  $n$  es el número de puntos y  $d$  la dimensión del arreglo, en la Fig. 4.26 se muestra el resultado de la conversión de la nube de puntos.



**Fig. 4.25.** Nube de puntos convertida en arreglo (a) y nube de puntos normalizada (b).

```
Target as np
[[-0.35825908  0.14318457 -1.4105898 ]
 [-0.4557273  0.79579306 -1.3664376 ]
 [ 1.8812275  0.92145395 -2.2810001 ]
 ...
 [ 0.93407625 -0.76303023 -1.2607083 ]
 [ 0.4643882  -0.74615645 -1.2253335 ]
 [ 0.5968354  -0.7572412  -1.24      ]]
(2048, 3)
```

**Fig. 4.26.** Forma del arreglo de la nube de puntos  $(n, d)$ .

Una vez que se tienen los datos normalizados, son convertidos en tensores, estos tensores tendrán la forma  $([n, d])$ . En la Fig. 4.27 se muestra el resultado de la conversión.

```
Template Tensor 1
torch.Size([2048, 3])
```

**Fig. 4.27.** Forma del tensor de la nube de puntos  $([n, d])$ .

Teniendo todos los datos convertidos en tensores, el siguiente paso es crear el conjunto de pruebas. La entrada de la arquitectura debe de tener la forma  $([b, n, d])$ , donde  $b$  es el tamaño de *batch*. Cada conjunto de prueba contiene dos tensores con la forma  $[T([n, d]), S([n, d])]$ , donde  $T$  corresponde al tensor de la nube de puntos objetivo y  $S$  al tensor de la nube de puntos fuente.

Para darle la forma que requiere la red neuronal se utilizó la subclase para datos primitivos *DataLoader* de PyTorch para crear el conjunto de pruebas y dar la forma  $[T([b, n, d]), S([b, n, d])]$ .

```
[tensor([[[[-0.3883,  0.4394, -0.7104],
           [ 0.6357,  0.8979, -1.4659],
           [-0.4210, -0.3662, -1.2958],
           ...,
           [ 0.9757, -0.8186, -1.3430],
           [ 0.8265, -0.8185, -1.3403],
           [ 0.8956, -0.8180, -1.3395]]]), tensor([[[[-0.4101,  0.4577, -0.7564],
           [ 0.9176,  0.7271, -1.7547],
           [-0.0722,  0.9243, -1.4604],
           ...,
           [-0.4607, -0.7637, -1.2560],
           [ 1.1822, -0.9033, -1.4833],
           [ 0.2187, -0.8556, -1.4010]]]])]
Template
tensor([[[[-0.3883,  0.4394, -0.7104],
           [ 0.6357,  0.8979, -1.4659],
           [-0.4210, -0.3662, -1.2958],
           ...,
           [ 0.9757, -0.8186, -1.3430],
           [ 0.8265, -0.8185, -1.3403],
           [ 0.8956, -0.8180, -1.3395]]]])
torch.Size([1, 2048, 3])
Source
tensor([[[[-0.4101,  0.4577, -0.7564],
           [ 0.9176,  0.7271, -1.7547],
           [-0.0722,  0.9243, -1.4604],
           ...,
           [-0.4607, -0.7637, -1.2560],
           [ 1.1822, -0.9033, -1.4833],
           [ 0.2187, -0.8556, -1.4010]]]])
torch.Size([1, 2048, 3])
```

**Fig. 4.28.** Creación del conjunto de pruebas con forma  $[T([b, n, d]), S([b, n, d])]$ .

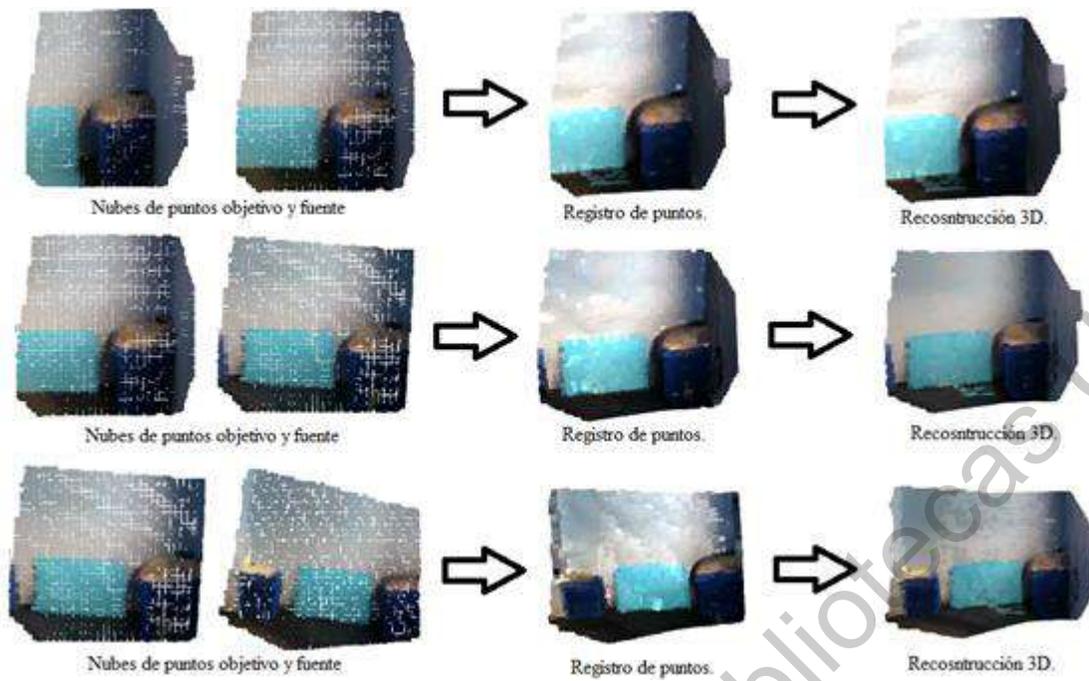
#### 4.2.4 Reconstrucción 3D.

El sistema de reconstrucción 3D basado en aprendizaje profundo opera bajo los siguientes pasos:

1. Toma las nubes de puntos dentro del conjunto de pruebas correspondiendo a la nube de puntos objetivo y fuente.
2. Este par de nubes de puntos entran al algoritmo inteligente para el registro de puntos.
3. Cuando se tiene la transformada de cuerpo rígido, es aplicada a la nube de puntos fuente original dentro del conjunto de pruebas.
4. Ambas nubes de puntos, objetivo y fuente, son desplegadas en la gráfica de la reconstrucción 3D.
5. La nube de puntos fuente pasa a ser seleccionada como la nueva nube de puntos objetivo dentro del conjunto de pruebas.
6. Si la nube de puntos actual es la última dentro del conjunto de datos de prueba, el proceso finaliza. En caso contrario vuelve al paso 1.

En la Fig. 4.29 se muestra un diagrama del proceso del sistema de reconstrucción 3D basado en algoritmos de aprendizaje profundo. Este proceso se realiza hasta que se hayan alineado todas las nubes de puntos correspondientes a la escena. En la Fig. 4.30 se muestra el resultado final de la reconstrucción 3D obtenida por el modelo basado en aprendizaje propuesto para la escena 1 desde una vista superior y en la Fig. 4.31 se muestra la comparación de los resultados cualitativos entre ambos sistemas de reconstrucción 3D.

En la Tabla 4.10 se muestra la comparativa de los resultados cuantitativos obtenidos por ambos sistemas de reconstrucción 3D.



**Fig. 4.29.** Proceso del sistema de reconstrucción 3D basado en algoritmos de aprendizaje profundo (4 nubes de puntos).



**Fig. 4.30.** Reconstrucción 3D con métodos de aprendizaje, escena 1, vista superior.



**Fig. 4.31.** Comparación de los resultados cualitativos entre ambos sistemas de reconstrucción 3D (vista superior). Modelo basado en métodos tradicionales (a), modelo basado en aprendizaje profundo (b).

Como se puede observar en la Fig. 4.31, algunas de las características de la reconstrucción 3D basada en métodos tradicionales de la escena 1 se mejora con el método propuesto basado en aprendizaje profundo. En el óvalo amarillo se observa una mejor definición en los contornos de la esquina inferior derecha. Sin embargo, también sucede lo contrario con algunas características como en el área dentro del óvalo verde .

**Tabla 4.10.** Comparación de los resultados finales obtenidos.

Modelo	RMSE	<i>fitness score</i>	Procesamiento (CPU)	Procesamiento (GPU)
Aprendizaje	<b>0.030258461</b>	<b>0.95381547</b>	612.755395seg	<b>6.1004614seg</b>
Tradicional	0.0310	0.9422	<b>118.55655seg</b>	69.5858261seg

Tal como se muestra en la Tabla 4.10, el sistema de reconstrucción 3D basado en algoritmos de aprendizaje profundo obtiene mejores resultados que el sistema de reconstrucción 3D basado en métodos tradicionales. Sin embargo, el tiempo de procesamiento fue exageradamente elevado cuando se implementó en la computadora personal, la cual carece de una GPU, siendo de 10 minutos y medio aproximadamente, mientras que el sistema basado en métodos tradicionales tardó 2 minutos aproximadamente.

Por tal motivo se decidió implementar ambos sistemas de reconstrucción en Google Colaboratory para comparar el tiempo que tardan ambos algoritmos en una GPU.

Los resultados obtenidos demostraron un gran cambio en cuanto al tiempo de procesamiento por parte del sistema basado en métodos de aprendizaje profundo reduciéndose hasta los 6.1 segundos, mientras que el sistema basado en métodos tradicionales se reduce en un 45% aproximadamente obteniendo un tiempo de 69.5 segundos. Dados estos resultados es ampliamente recomendable el uso de una GPU para este tipo de sistemas basados en aprendizaje profundo o una CPU con buena calidad computacional.

#### **4.3 REQUERIMIENTOS PARA LA IMPLEMENTACIÓN EN UN SISTEMA SLAM.**

*Simultaneous Localization and Mapping* por sus siglas en inglés SLAM, o en español: Localización y Mapeo Simultáneos, es una técnica usada por robots móviles y vehículos autónomos para estimar simultáneamente la posición de un robot dentro de un entorno desconocido y la construcción de un modelo de dicho entorno con ayuda de sensores integrados. Los sistemas SLAM se componen principalmente de los siguientes elementos:

- Hardware: Mecanismo móvil y sensores.
- Sistema de Odometría.
- Sistema de medición de distancia y profundidad.

La odometría junto con la medición de distancia y profundidad pueden realizarse de diferentes maneras dependiendo del tipo de SLAM que se desee implementar; SLAM basado en técnicas probabilísticas o SLAM basado en técnicas visuales.

El trabajo realizado en esta tesis está dedicado únicamente a la reconstrucción 3D y podría ser aplicado la parte de la medición de distancia y profundidad. Para poder realizar un sistema SLAM se requeriría integrar la parte de odometría. En la Fig. 4.32 se muestra el esquema de un sistema SLAM.

En este modelo el sistema de SLAM es del tipo visual. Se implementa la cámara Intel RealSense D435 para realizar la medición de distancia y profundidad, y poder hacer la reconstrucción 3D mediante el algoritmo inteligente presentado en este trabajo. Para realizar

la parte de localización se propone implementar la cámara de Intel RealSense T265 integrada con una unidad de medición inercial (IMU por sus siglas en inglés *inertial measurement unit*) para la parte de odometría.

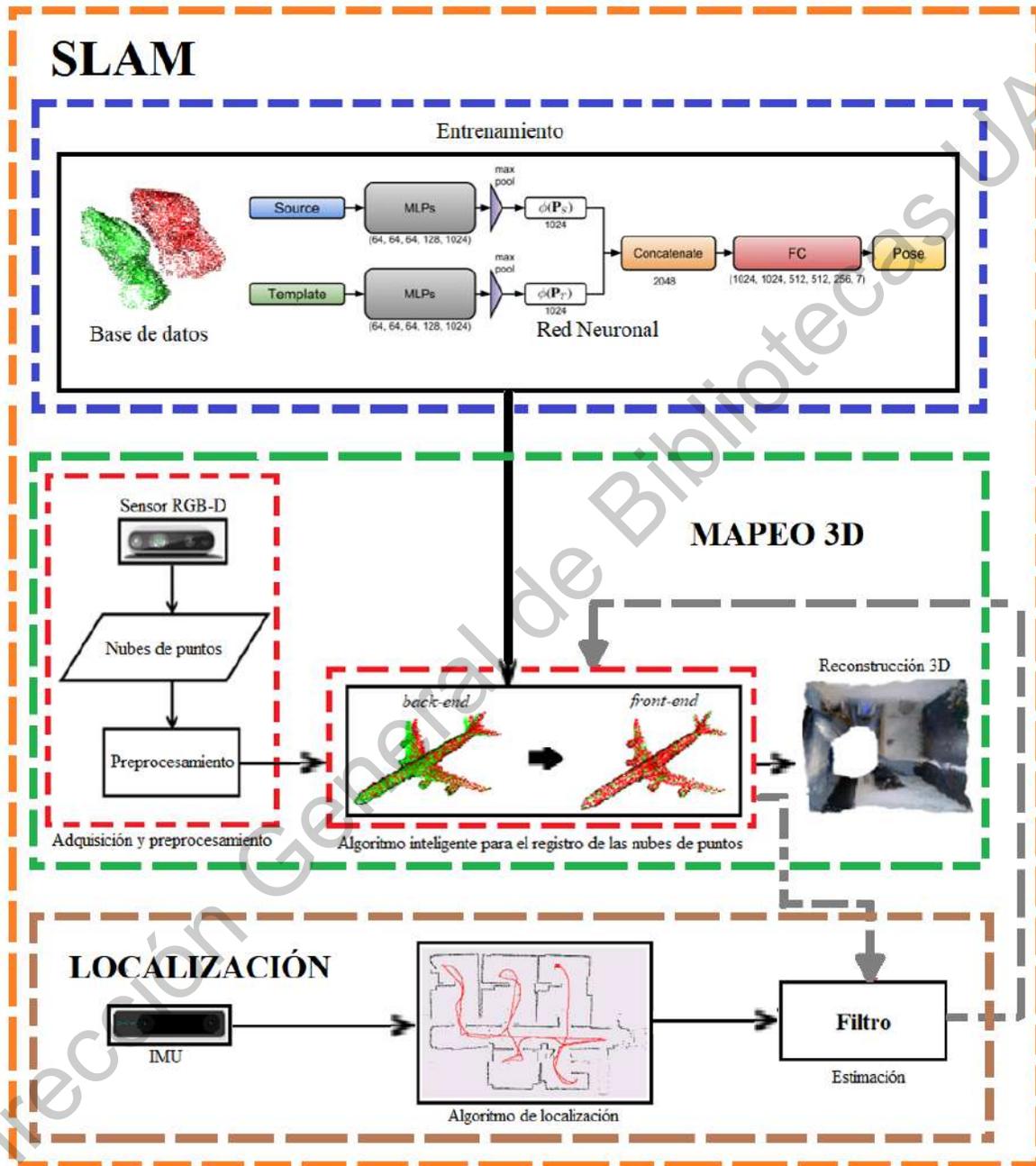


Fig. 4.32. Modelo del sistema SLAM visual.

El algoritmo de localización consta de la adquisición y preprocesamiento de los datos adquiridos por el sensor Intel RealSense T265 para la parte de odometría, para realizar una localización más exacta y precisa se requiere añadir un filtro no lineal para linealizar el sistema, el más utilizado en estos casos es el filtro de kalman extendido.

Este algoritmo interactúa con el algoritmo de mapeo en el módulo del registro de nubes de puntos brindando y recibiendo información para mejorar las características de la reconstrucción 3D y la localización del sistema dentro del entorno. Algunos métodos basados en algoritmos genéticos o redes bayesianas pueden ser implementados en este módulo para realizar un algoritmo de localización más robusto al momento de seguir una trayectoria.

En general se puede resumir la implementación del módulo de localización en los siguientes pasos:

- Adquisición y preprocesamiento de los datos. Por medio del sensor Intel RealSense T265 integrada con una IMU se realiza la odometría.
- Algoritmo de localización. Basado en métodos de inteligencia artificial como algoritmos evolutivos, algoritmos genéticos, redes bayesianas, entre otros.
- Filtro. Como refinamiento al algoritmo de localización y linealización del sistema para interactuar con el módulo de mapeo y mejorar el rendimiento de ambos módulos y obtener un sistema más robusto.

## 5. CONCLUSIONES Y TRABAJO FUTURO

La reconstrucción 3D es una de las principales áreas de investigación dentro de las ciencias de la computación, visión por computadora y la robótica. El principal caso de estudio dentro de la reconstrucción 3D es el registro de nubes de puntos. En la actualidad, con el gran auge de la inteligencia artificial, el campo del Deep learning, o aprendizaje profundo, ha llamado bastante la atención en el campo de la investigación y esto ha llevado al desarrollo de diversos algoritmos para realizar tareas como la detección y clasificación de objetos, segmentación y el registro de nubes de puntos. El presente trabajo realiza la implementación de dos sistemas de reconstrucción 3D, el primero de ellos basado en métodos tradicionales y el segundo basado en algoritmos de aprendizaje profundo reduciendo tiempo de procesamiento y mejorando la exactitud del sistema.

En esta tesis también se describe detalladamente cada paso tomado para la implementación de ambos sistemas y poder obtener la reconstrucción 3D de un entorno cerrado. También se presenta la implementación y comparación de diversos algoritmos de la literatura utilizados para realizar el registro de las nubes de puntos. Las métricas utilizadas fueron el valor RMSE, el área de superposición y el tiempo de procesamiento. Con base en los resultados obtenidos se comprueba la hipótesis planteada para esta investigación.

Hay puntos muy importantes que comentar sobre el trabajo realizado en esta tesis. A continuación, se mencionan de manera detallada.

### **RECONSTRUCCIÓN 3D TRADICIONAL.**

El sistema desarrollado para la reconstrucción 3D basado en métodos tradicionales realiza la inicialización a nivel global por medio del algoritmo FGR y la transformación afín con el método de registro a nivel local ICP punto a plano y por medio del método de gráfico de pose se realiza un refinamiento y mejor representación de la reconstrucción 3D.

Establecer una buena pre-alineación para los algoritmos locales es crucial en su desempeño, por esta razón se utilizó un método global sencillo y capaz de obtener buenos resultados como inicialización con pocas iteraciones. Si bien el uso del método de gráfico de

pose incrementa considerablemente el tiempo de procesamiento, también hace posible obtener una mejor estimación de la transformación rígida y con esto lograr un mejor mapeo tridimensional de la escena demostrando la reducción del error cuadrático medio de manera considerable entre cada una de las nubes de puntos alineadas.

## **RECONSTRUCCIÓN 3D BASADA EN APRENDIZAJE PROFUNDO.**

El sistema desarrollado para la reconstrucción 3D basado en métodos de aprendizaje profundo consiste principalmente en la implementación de un módulo de aprendizaje, el cual realiza una inicialización inteligente, y un módulo de refinamiento encargado de realizar la transformación afín.

Como ya se ha mencionado, dar una buena inicialización a los algoritmos locales, como es el caso de ICP punto a plano, es crucial para su desempeño. El uso de algoritmos de aprendizaje profundo para realizar esta función asegura tener una inicialización adecuada para el buen desempeño del algoritmo ICP además de acelerar el tiempo de convergencia del proceso. Con esto se evita también utilizar otra clase de métodos como el gráfico de pose ya que no son realmente necesarios para este sistema.

## **METODOS TRADICIONALES VS APRENDIZAJE PROFUNDO.**

Los algoritmos basados en métodos de aprendizaje profundo demuestran tener mejores resultados que los algoritmos basados en métodos tradicionales. Cuando se quiere realizar alguna tarea sencilla, como la simple alineación de dos nubes de puntos, el uso de algoritmos de aprendizaje profundo ofrece mayores ventajas y mejor rendimiento en el tiempo de procesamiento y la exactitud. Sin embargo, el presente trabajo demostró que al realizar una tarea compleja como la reconstrucción 3D de una escena la simple utilización de las redes neuronales no basta, esto también se observó en los trabajos de la literatura.

Como se pudo observar en las gráficas de las métricas, la arquitectura PCRNet que fue entrenada obtiene mejores resultados que los algoritmos tradicionales ICP punto a plano y FGR. Sin embargo, al combinar ambos algoritmos tradicionales se obtienen mejores resultados que con la red neuronal por sí sola. De la misma manera, al combinar el módulo

de aprendizaje con el módulo de refinamiento se obtuvieron los mejores resultados para la reconstrucción 3D.

Cada algoritmo presenta ventajas y desventajas, por ejemplo; el problema más común en ICP es su susceptibilidad a caer en mínimos locales y su necesidad de una buena inicialización para obtener resultados óptimos, por su parte al ser PCRNet un método global obtiene buenos resultados de manera rápida basándose en la geometría de las nubes de puntos, sin embargo, no alcanza la exactitud y precisión del algoritmo ICP cuando este cuenta con una buena inicialización. Es por esto que la ejecución en conjunto de ambos métodos ofrece una herramienta mucho más sofisticada y capaz de tener un mejor desempeño.

### **COSTO COMPUTACIONAL.**

Algo importante de mencionar es que cuando se trabaja con algoritmos de aprendizaje profundo, dependiendo de la complejidad de la arquitectura de la red neuronal, es altamente recomendable utilizar una computadora con GPU ya que, como sucedió en este trabajo, al implementar el modelo en la computadora portátil con CPU el tiempo de procesamiento fue muy elevado. Este fue un caso especial debido al bajo rendimiento computacional de la computadora utilizada.

Al implementar ambos sistemas en la GPU de Google Colaboratory el tiempo de procesamiento bajó radicalmente en el sistema de aprendizaje profundo, demostrando una mayor eficiencia que el sistema de reconstrucción tradicional. Sin embargo, al utilizar esta clase de algoritmos, teniendo arquitecturas complejas, es indispensable un mayor rendimiento computacional y por ende un mayor costo económico.

### **TRABAJO FUTURO.**

Una de las principales áreas de oportunidad dentro de este trabajo es la introducción de un filtro robusto para eliminar puntos fuera de línea o reducir el ruido en las nubes de puntos ya que el sistema de reconstrucción 3D se ve afectado cuando los datos de entrada presentan estas características.

Como se describió en los resultados finales, el sistema de reconstrucción 3D puede ser implementado junto con un sistema de localización para realizar un sistema de localización y mapeo simultaneo (SLAM). Lo ideal sería que el módulo de localización implemente algoritmos basados en inteligencia artificial, esto hace que este tema sea bastante extenso por lo que se puede realizar otro trabajo de tesis.

Dirección General de Bibliotecas UAQ

## 6. REFERENCIAS

- Wu, Y., Chan, L. & Lin, W. (2019). Tangible and Visible 3D Object Reconstruction in Augmented Reality IEEE International Symposium on Mixed and Augmented Reality (ISMAR), Beijing, China, pp. 26-36
- Durrant-Whyte, H. & Bailey, T. (2006). Simultaneous Localization and Mapping (SLAM) Part 1 The Essential Algorithms. University of Sydney
- Chen, W., Ihara, S. & Hasegawa, M. (2020). Proposal of a rescue operation support system based on 3D Reconstruction, GPS, and digital pen. International Workshop on Advanced Imaging Technology (IWAIT), Vol 11515
- Lee, H., Song, S. & Jo, S. (2016). 3D Reconstruction using a Sparse Laser Scanner and a Single Camera for Outdoor Autonomous Vehicle. 19<sup>th</sup> International Conference on Intelligent Transportation Systems (ITSC), pp. 629-634
- Furukawa, Y. & Hernández, C. (2013). Multi-View Stereo: A Tutorial. Foundations and Trends in Computer Graphics and Vision. vol. 9, no. 1-2, pp. 1-148
- Zhu, H., Guo, B., Zou, k., Li, Y., Yuen, K., Mihaylova, L. & Leung, H. (2019). A Review of Point Set Registration: From Pairwise Registration to Groupwise Registration. Sensors Journal, 19(5):1191
- Besl, P. & McKay, N. (1992). A method for registration of 3-d shapes. In Robotics-DL tentative, pp. 586-606
- Rusu, R., Blodow, N., Marton, Z. & Lipschutz, M. (2008). Aligning Point Cloud Views using Persistent Feature Histograms. 21<sup>st</sup> IEEE International Conference on Intelligent Robots and Systems (IROS), pp. 22-26
- Tao, N. & Hasegawa, H. (2015). Global iterative closest point using nested annealing for initialization. 19<sup>th</sup> International Conference on Knowledge Based and Intelligent Information and Engineering Systems, Procedia Computer Science, pp. 381-390
- Zhou, Q., Park, J. & Koltun, V. (2016). Fast Global Registration. European Conference on Computer Vision, pp. 766-782

- Yan, L., Dai, J., Tan, J., Hua, L. & Chen, C. (2019). Global fine registration of point cloud in LiDar SLAM based on pose graph. *Journal of geodesy and Geoinformation science*. Vol 48. 3(2)
- Chen, Y. & Medioni, G. (1991). Object modeling by registration of multiple range images. *Proceeding of the 1991 IEEE International Conference on Robotics and Automation*, 10(3)
- Wu, Y., Wang, W., Lu, K., Wei, Y. & Chen, Z. (2015). A new method for registration of 3D point sets with low overlapping ratios. *13<sup>th</sup> CIRP conference on Computer Aided Tolerancing*. 27, pp. 202-206
- Agamennoni, R., Fontana, S. & Sorrenti, D. (2016). Point clouds registration with probabilistic data association. *Proceeding of the International Conference on Intelligent Robots and Systems (IROS)*
- Liu, S., Gao, D., Wang, P., Guo, X., Xu, J. & Liu, D. (2018). A Depth-Based Weighted Point Cloud Registration for Indoor Scene. *Sensors Journal*, 18(11):3608
- Seal, A. & Bhowmick, A. (2017). Performance Analysis of Iterative Closest Point (ICP) Algorithm using modified Hausdorff Distance. *International Research Journal of Engineering and Technology (IRJET)*. Vol. 04, 07
- Rusu, R., Blodow, N. & Beetz, M. (2009). Fast Point Feature Histograms (FPFH) for 3D Registration. *International Conference on Robotics and Automation*, pp. 3212-3217
- Chen, C., Hung, Y. & Cheng, J. (1999). Ransac-based darces: A new approach to fast automatic registration of partially overlapping range images. *IEEE Trans. Pattern Anal. Mach. Intell.*, 21: 1229-1234
- Koguciuk, D. (2017). Parallel RANSAC for Point Cloud Registration. *Foundations of Computing and Decision Sciences*. Vol. 42, 203-217
- Yang, J., Li, H., Dylan, C. & Jia, Y. (2016). Go-ICP: A Globally Optimal Solution to 3D ICP Point-Set Registration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*

- Low, K. (2004). Linear Least-Squares Optimization for Point-to-Plane ICP Surface Registration. Department of Computer Science, University of North Carolina at Chapel Hill. Technical report
- Stachniss, C. (2020). Graph-based SLAM using Pose Graphs. Recuperado de <https://www.youtube.com/watch?v=uHbRKvD8TWg>  
<https://www.intelrealsense.com/depth-camera-d435>
- Miller, F., Vandome, A. & Mcbrewhster, J. (2009). KD-Tree. Alpha Press
- W. Lu, G. Wan, Y. Zhou, X. Fu, P. Yuan & S. Song. (2019). "DeepVCP: An End-to-End Deep Neural Network for Point Cloud Registration," 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 12-21, doi: 10.1109/ICCV.2019.00010.
- R. Q. Charles, H. Su, M. Kaichun & L. J. Guibas. (2017). "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 77-85, doi: 10.1109/CVPR.2017.16.
- R. Q. Charles, H. Su, Li, Y. & L. J. Guibas. (2017). "PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 77-85, doi: 10.1109/CVPR.2017.16.
- Zhang, Z., Dai, Y. & Sun, J. (2020). Deep Learning based point cloud registration: an overview. Virtual Reality & Intelligent Hardware. Vol. 2, pp. 222-246.
- Wang, Y. & Solomon, J. (2019). Deep Closest Point: Learning Representations for Point Cloud Registration, pp.3522-3531.
- Deng, H., Birdal, T. & Ilic, S. (2018) PPFNet: global context aware local features for robust 3D point matching. In: 2018 IEEE/ CVF Conference on Computer Vision and Pattern Recognition. Salt Lake City, UT, USA, IEEE, pp. 195–205.
- Wang, Y., Sun, Y., Liu, Z., Sarma, S., Bronstein, M. & Solomon, J. (2019). Dynamic graph CNN for learning on point clouds. ACM Transactions on Graphics, 38(5): 1–12.
- Atzmon, M., Maron, H. & Lipman, Y. (2018). Point convolutional neural networks by extension operators. ACM Transactions on Graphics, 37(4): 1–12.

- Li, Y., Bu, R., Sun, M., Wu, W., Di, X. & Chen, B. (2018). Pointcnn: Convolution on x-transformed points. In: *Advances in neural information processing systems*, pp. 820–830.
- Aoki, Y., Goforth, H., Srivatsan, R. & Lucey, S. (2019). PointNetLK: robust & efficient point cloud registration using PointNet. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7156–7165.
- Baker, S. & Matthews, I. (2004). Lucas-kanade 20 years on: a unifying framework. *International Journal of Computer Vision*, 56(3): 221–255.
- Huang, X., Mei, G. & Zhang, J. (2020). Feature-metric registration: a fast semi-supervised approach for robust point cloud registration without correspondences.
- Sarode, V., Li, X., Goforth, H., Aoki, Y., Srivatsan, R.A., Lucey, S., & Choset, H. (2019). PCNet: Point Cloud Registration Network using PointNet Encoding. *ArXiv*, abs/1908.07906.
- Russell, S. J. & Norvig, P. (2009). *Artificial intelligence: a modern approach*, Upper Saddle River. p.2.
- Goodfellow, I., Bengio, Y. & Courville, A. (2016). *Deep Learning*. MIT Press.
- Basogain, X. (2015). *Redes Neuronales Artificiales y sus Aplicaciones*. Escuela superior de ingeniería de Bilbao.
- Larrañaga, P., Inza, I. & Moujahid, A. (2020). *Redes Neuronales*. Departamento de ciencias de la computación e Inteligencia artificial.
- Izaurieta, F. & Saavedra, C. (2015). *Redes Neuronales Artificiales*. Departamento de Física, Universidad de Concepción, Chile.
- García, F. (2017). *Tools for 3D Point Cloud Registration*. Doctoral Thesis.
- Han, X., Laga, H. & Bennamoun, M. (2019). *Image-based 3D Object Reconstruction: State-of-the-Art and Trends in the Deep Learning Era*.
- Intwala, A. & Magikar, A. (2016). *A Review on Process of 3D Model Reconstruction*. 10.1109/ICEEOT.2016.7755218.

PRINCETON ModelNet40 (<https://modelnet.cs.princeton.edu/>).

Dirección General de Bibliotecas UAQ

## 7. ANEXOS

### 7.1 ARTÍCULO PUBLICADO.

**COMIA 2021**

**UNISON**

LA SOCIEDAD MEXICANA DE INTELIGENCIA ARTIFICIAL  
Y LA UNIVERSIDAD DE SONORA

**OTORGAN ESTE CERTIFICADO A:**

**Víctor Beltrán Barrera, Jesús Carlos Pedraza Ortega, Juan Manuel Ramos Arreguín, Marco Antonio Aceves Fernández, Saúl Tovar Arriaga y Efrén Gorrostieta Hurtado**

por la presentación del artículo titulado:

**Registro de Nubes de Puntos Por Pares con Optimización Global basado en Gráfico de Pose para un Sistema de Reconstrucción 3D**

**en el XIII Congreso Mexicano de Inteligencia Artificial, COMIA 2021**  
Hermosillo, Sonora, México, del 18 al 21 de mayo de 2021

Dr. Félix Castro Espinoza  
Presidente SMIA

Dr. Oscar Herrera Alcántara  
Presidente del Comité de Programa

Dr. Noé A. Castro Sánchez  
Presidente del Comité de Programa

Dr. María Elena Robles Baldenegro  
Comité Local COMIA

## 7.2 CONSTANCIA DE LENGUA EXTRANJERA.



UNIVERSIDAD AUTÓNOMA DE QUERÉTARO  
FACULTAD DE LENGUAS Y LETRAS

**A QUIEN CORRESPONDA:**

La que suscribe, Directora de la Facultad de Lenguas y Letras, hace **C O N S T A R** que

**BELTRAN BARRERA VICTOR**

Presentó el **Examen de Manejo de la Lengua** efectuado el día tres de junio de dos mil veintiuno, en el cual obtuvo la siguiente calificación:

**8**

Se extiende la presente a petición de la parte interesada, para los fines escolares y legales que le convengan, en el Campus Aeropuerto de la Universidad Autónoma de Querétaro, el día catorce de junio de dos mil veintiuno.

Atentamente,  
"Enlazar Culturas por la Palabra"



**LIC. LAURA PÉREZ TÉLLEZ**

LPT/japa\*CL\*FLL-C.-951