RECONSTRUCCIÓN DINÁMICA EN V-SLAM BASADO EN SEGMENTACIÓN DE ZONAS DE PUNTOS DE ALTA VARIABILIDAD

Carlos Esteban Orbes Rosero

Juan Camilo Torres Muñoz

Trabajo de grado. Pregrado en Ingeniería

en Automática Industrial

Director

M.Sc. Elena Muñoz España

Co-director

M.Sc. Juan Fernando Flórez

Universidad del Cauca

Facultad de Ingeniería Electrónica y Telecomunicaciones

Programa de Ingeniería en Automática Industrial

Popayán, 2023

RECONSTRUCCIÓN DINÁMICA EN V-SLAM BASADO EN SEGMENTACIÓN DE ZONAS DE PUNTOS DE ALTA VARIABILIDAD

Trabajo de grado presentado a la Facultad de Ingeniería Electrónica y Telecomunicaciones de la Universidad del Cauca para la obtención del Título de

Ingeniero en Automática Industrial

Director

M.Sc. Elena Muñoz España

Co-director

M.Sc. Juan Fernando Flórez

Popayán, 2023

Querida madre, queridas tías, primas y hermano,

Hoy quiero expresar mi más sincero agradecimiento hacia cada uno de ustedes. Querida madre, tu amor incondicional y tu constante apoyo han sido un faro en mi vida. Queridas tías, su cariño y sabiduría han dejado una huella imborrable en mi corazón. Y a ti, mi querido hermano mayor, aunque no hemos convivido tanto, sé que siempre estás ahí dispuesto a apoyarme y darme un empujón adelante. Tu presencia en mi vida es invaluable y valoro profundamente tus gestos de apoyo y aliento. También quiero agradecer a mis primas, quienes con su constante apoyo han hecho posible este sueño, brindándome su cariño y guiándome en diferentes etapas de mi vida. A todos ustedes, gracias por ser pilares fundamentales en mi camino y por enriquecer mi vida de manera significativa.

Con amor y gratitud,

Juan Camilo Torres Muñoz

Queridos padres, hermanas, abuelos, novia y directores,

Hoy deseo expresar mi más profundo agradecimiento a cada uno de ustedes. Queridos padres, su apoyo incondicional, enseñanzas, valores y amor constante han sido la fuerza impulsora detrás de mi vida y el éxito de este proyecto. A mis queridas hermanas, les agradezco de corazón su cariño, apoyo y paciencia, elementos indispensables en cada etapa de este proceso. Queridos abuelos, su sabiduría y consejos han sido pilares fundamentales en este logro. A ti, querida novia, quiero expresarte mi gratitud sincera por estar a mi lado en los momentos difíciles, brindándome tu apoyo incondicional.

También quiero extender mi agradecimiento a mis directores por brindarme esta valiosa oportunidad, así como al SENA y la Universidad del Cauca por su inestimable respaldo en mi desarrollo personal y profesional.

Con mucho amor,

Carlos Esteban Orbes Rosero

Resumen

El desarrollo de vehículos autónomos es un tema crucial en este siglo, lo que ha impulsado la investigación en diversas áreas, incluyendo los sistemas de localización y mapeo simultáneo (SLAM). Dentro de los avances en esta área, se destaca el Visual SLAM, que utiliza técnicas de visión artificial para capturar información del entorno, reconstruir escenas y crear mapas 3D. Esta tecnología emergente tiene un gran potencial y está en constante evolución. No obstante, todavía existen desafíos por superar, como el diseño de sistemas de bajo peso computacional que permitan la ejecución en línea, y su inclusión en sistemas autónomos.

Esta investigación tiene como objetivo "Proponer un algoritmo soportado en un sensor RGB-D para la segmentación dinámica de objetos en movimiento para escenas en interiores", para lograr este objetivo se ha desarrollado una metodología eficaz qué logra reconstruir escenas en 3D. En primer lugar, se procesan los datos de una cámara RGB-D. A continuación, se aplica un algoritmo especializado para convertir estos datos en una representación simulada de LIDAR. Luego, se fusionan los datos de los sensores (RGB-D, LIDAR virtual) para detectar y eliminar objetos dinámicos, y se concatena las nubes de puntos para reconstruir la escena. Este proceso, llamado Rec-HV, es de baja complejidad computacional y se puede utilizar en tiempo real. Además, se ha implementado un método para medir la correlación entre dos escenas. La metodología se ha evaluado en diferentes escenas y ha demostrado ser eficiente incluso ante la presencia de objetos en movimiento.

Tabla de Contenido

Tabla de Figuras

Capítulo 1.
Aspectos generales del proyecto

La localización y mapeo simultáneos (SLAM) [1] es una técnica para la reconstrucción en tiempo real del entorno y la estimación de la posición del robot en él. Esta técnica es esencial para muchas aplicaciones que utilizan sensores para la navegación y estimación del movimiento [2]. Hay dos corrientes principales de SLAM: láser y Visual SLAM (V-SLAM) [3], donde el uso de cámaras y otros sensores visuales se ha convertido en una herramienta importante en el reconocimiento de escenas y entornos desconocidos [4].

Para mejorar la reconstrucción en tiempo real y lograr una representación más precisa de la escena [5], es necesario separar los elementos dinámicos de los estáticos, en V-SLAM [6][7], esto se logra mediante la segmentación de objetos dinámicos [8][9], lo que permite una mayor robustez ante dichos objetos en una escena desconocida [10][11]. La mayoría de los métodos actuales suponen entornos estáticos y tratan a los objetos dinámicos como ruido [12][13][14]. Sin embargo, esto puede generar errores acumulativos y costos computacionales innecesarios.

La técnica conocida como 'Pointcloud_to_laserscan' juega un papel fundamental en el procesamiento de datos LIDAR. Consiste en convertir nubes de puntos 3D en escaneos láser 2D, lo que facilita la visualización y análisis de los datos. Varios estudios han contribuido a mejorar la eficiencia y precisión de esta técnica en diversos entornos y aplicaciones.

Uno de los estudios relevantes, publicado en 2021 [15], presentó una técnica para el mapeo de entornos desconocidos utilizando LIDAR y 'Pointcloud_to_laserscan'. Esta técnica demostró una notable mejora en la eficiencia del proceso de mapeo y en la precisión de la reconstrucción del mapa, alcanzando una precisión cercana al 90% en la cartografía de entornos desconocidos.

Además, se han aplicado otras metodologías interesantes. Por ejemplo, [16] utilizó

'Pointcloud_to_laserscan' junto con LIDAR para la detección en tiempo real de objetos. Esta técnica logró una tasa de detección de objetos del 93.4%, lo que evidencia su eficacia en esta área. También se empleó en la detección de obstáculos [17], mejorando la precisión en comparación con métodos tradicionales y resultando altamente efectiva en aplicaciones de conducción autónoma [18], logrando una precisión del 98%.

Los estudios mencionados enfatizan la importancia de 'Pointcloud_to_laserscan' en el procesamiento de datos LIDAR. Su precisión y eficiencia han sido ampliamente probadas y mejoradas en diversos contextos, lo que la convierte en una herramienta valiosa para diversas aplicaciones industriales e investigativas.

En el presente estudio, se propone un método de bajo costo computacional para la eliminación de objetos dinámicos y la reconstrucción de ambientes en 3D, basado en la técnica SLAM. Para ello, se fusiona los datos de una cámara RGB-D con un emulador de LIDAR basado en los mismos datos ('Pointcloud_to_laserscan'), lo que permite generar una reconstrucción del ambiente con objetos en movimiento.

Los resultados obtenidos del procesamiento estadístico de los datos revelan la detección de las zonas donde se ubican los objetos dinámicos, la segmentación y eliminación en tiempo real de dichos objetos, y la consecuente reconstrucción del ambiente. Además, se implementa un método para establecer una métrica de comparación entre dos escenas 3D, que permite establecer la correlación entre escenas con la misma perspectiva desde una posición relativa idéntica. Todo esto se realiza con el objetivo de aplicarlo a sistemas autónomos de navegación en ambientes interiores.

1.1 Objetivos

1.1.1 General

Proponer un algoritmo soportado en un sensor RGB-D para la segmentación dinámica de objetos en movimiento para escenas en interiores.

1.1.2 Específicos

- Detectar un obj*e*to dinámico de tamaño no mayor al 30% de la escena total a tres velocidades diferentes a partir de la nube de puntos RGB-D.

- ❖ Reconstruir dinámicamente el componente estático de la escena con objetos en movimiento a partir de los datos arrojados por la nube de puntos RGB-D.
- ❖ Estimar la calidad de la reconstrucción obtenida a partir de la segmentación dinámica de una escena en interiores con objetos en movimiento.

1.2. Estructura del documento

Este trabajo se estructura en cinco capítulos interconectados que guían al lector a través de un recorrido lógico y coherente, abarcando desde los fundamentos teóricos hasta la implementación y validación de un algoritmo innovador. Cada capítulo desempeña un papel fundamental en la construcción de un panorama completo y profundo de la investigación realizada.

En el Capítulo 1, se introduce el contexto y el propósito de este trabajo, estableciendo las bases que justifican la necesidad de explorar y desarrollar un nuevo enfoque en el campo en cuestión. Aquí se presenta una visión general de la estructura del trabajo y se anticipan las etapas cruciales que serán abordadas en los capítulos subsiguientes.

El Capítulo 2, se centra en el establecimiento de los cimientos teóricos que permiten al lector adentrarse en los aspectos esenciales que fundamentan la construcción del algoritmo propuesto. Se abordan conceptos clave y teorías fundamentales que son indispensables para comprender la estructura y la lógica subyacente detrás del algoritmo en cuestión.

En el Capítulo 3, se presenta la propuesta con mayor detalle, comenzando con un análisis del algoritmo Rec-HV y su fundamentación matemática. También se establece la base de datos y condiciones que se utilizaran para desarrollar la fase de pruebas.

El Capítulo 4, se sumerge en la metodología utilizada para evaluar los algoritmos, desglosando la técnica de evaluación adoptada y presentando los resultados obtenidos de manera visual mediante gráficos y representaciones ilustrativas. Esta sección proporciona una perspectiva cuantitativa y cualitativa de la eficacia y el rendimiento del algoritmo propuesto.

El Capítulo 5, ofrece las conclusiones derivadas del estudio y la implementación del

algoritmo. Aquí se recapitulan los logros, se discuten los hallazgos significativos y se contextualizan dentro del panorama general del campo. Además, se plantean sugerencias valiosas para futuras investigaciones y desarrollos, abriendo la puerta a nuevas posibilidades y expandiendo el horizonte del conocimiento en este dominio.

En conjunto, estos cinco capítulos tejen una narrativa cohesiva y progresiva que guía al lector desde la base conceptual hasta la culminación de un algoritmo novedoso y su posterior evaluación crítica.

1.3. Planteamiento del problema

El presente trabajo surge como una extensión del proyecto de maestría "Detección de instancias de objetos en un sistema de localización y mapeo simultáneos basado en visión de máquina para entornos dinámicos", en este trabajo se preprocesan los datos de un LiDAR y una cámara RGB-D, para posteriormente realizar la detección del objeto dinámico.

A partir del algoritmo desarrollado en el proyecto de maestría, en esta investigación se propone trabajar un algoritmo que permita prescindir del LIDAR real, y realizar la segmentación a partir de detección de puntos de alta variabilidad presentes en la nube de puntos y genere la reconstrucción de la escena estática usando únicamente la cámara RGB-D.

De acuerdo con esto se plantea la siguiente pregunta de investigación: ¿Cuáles son las características que debe tener un algoritmo para la segmentación dinámica de objetos en movimiento para reconstruir la escena estática a partir de datos RGB-D?

Una de las diferencias más notorias entre el trabajo citado y el utilizado en este estudio es la sustitución del sensor LIDAR físico por un LIDAR virtual implementado a partir de la cámara RGB-D (Kinect). Esta transición ofrece beneficios en términos de optimización y peso computacional. Es relevante subrayar que, aunque no se realizó la traslación en el sistema, la idea de integrar una traslación siempre estuvo activa, lo que permitió mantener en consideración esta posibilidad para futuras investigaciones.

Del trabajo previamente mencionado, se adopta el procesamiento de las imágenes de profundidad y el algoritmo para calcular objetos dinámicos a partir de puntos de alta

variabilidad. También se incorpora el algoritmo de segmentación y reconstrucción. Sin embargo, la base móvil no es considerada, ya que en este trabajo no se implementa la sección de ORB-SLAM y sus cálculos. Cabe mencionar que ORB-SLAM es un sistema externo que permite obtener el posicionamiento con precisión, y aunque no se haya integrado en este estudio, se reconoce como una herramienta valiosa para futuras investigaciones. Además, se modifica la apertura angular debido a la operación exclusiva con la cámara RGB-D (Kinect).

En la etapa de reconstrucción, se ajusta el margen de preprocesamiento. Dado que el Kinect no maneja la apertura de un LIDAR convencional, esta adaptación es esencial parael proceso de detección. El reemplazo del LIDAR convencional con el nuevo nodo llamado "pointcloud_to_laserscan" implica cambios adicionales. Se introduce un nodo de procesamiento adicional para reducir la nube de puntos y evitar una mayor carga computacional. También se ajustan los ángulos en los nodos de detección, segmentación y reconstrucción para adaptarse al rango del Kinect, donde se implementa el LIDAR virtual.

Se incorporan nuevos conjuntos de datos preparados y grabados para su utilización. En la sección de correlación, se mejoran y modifican aspectos, incluyendo una interfaz de usuario que facilita la comprensión y visualización de resultados. Respecto a los resultados finales, se optimiza el hardware necesario. Mientras que el algoritmo original funcionaba en un equipo de alto rendimiento, en este trabajo se utiliza un equipo más básico (**¡Error! No se encuentra el origen de la referencia.**). Los tiempos de ejecución también mejoran, acercando la ejecución al tiempo real.

Tabla comparativa computadores	Procesador	Memoria RAM	GPU
PC anterior (DELL G15)	Razen 7 5800H 8 núcleos y 16 hilos de 4.4GHz	32 GB	4GB
PC actual (ASUS VIVOBOOK X512F)	Core i5 de 3.9 GHz, 4 núcleos reales y 4 virtuales	12 GB	MX230 Nvidia.

Tabla 1 Tabla comparativa rendimiento y optimización de los computadores utilizados

En el Capítulo 3 de este estudio, se proporciona una explicación detallada del algoritmo de referencia, con nuevas pruebas y datasets preparados para futuras investigaciones, como se mencionó previamente. Estos nuevos recursos contribuyen al desarrollo y progresión en el campo.

Capítulo 2.
Revisión de la Literatura

En este capítulo introductorio, se emprende un exhaustivo recorrido por la literatura que abarca desde las problemáticas inherentes al campo de investigación, como es el caso del SLAM, caracterizado por su alta demanda computacional, hasta la definición de diversas técnicas que prometen brindar soluciones y mejoras sustanciales en esta área. La Sección 2.1 ofrece una panorámica general de los enfoques que se desglosarán en las próximas secciones. A continuación, en la Sección 2.2, se explorarán los conceptos relacionados con la segmentación y la reconstrucción, dos elementos ampliamente empleados en el desarrollo de este trabajo con el objetivo de lograr resultados satisfactorios. En la Sección 2.3 se abordará el complejo ámbito del SLAM, así como las consideraciones cruciales en torno a las reconstrucciones de entornos estáticos y dinámicos en el contexto del SLAM. También se examinará en detalle el nodo "*pointcloud_to_laserscan*", un nuevo método empleado para obtener un LIDAR virtual. Adicionalmente, se discutirán métricas relevantes. Por último, en la Sección 2.4, se extraerán conclusiones fundamentales derivadas de esta labor de revisión literaria.

2.1 Introducción

El campo de SLAM ha cobrado un protagonismo significativo en el ámbito de la robótica en los últimos años, principalmente debido a su papel crucial en el desarrollo de vehículos autónomos. SLAM se enfoca en abordar el desafío de que los robots puedan mapear y comprender entornos desconocidos en tiempo real, lo que resulta fundamental para lograr la autonomía en la navegación robótica. Esto implica el uso y la integración de diversos sensores, como cámaras de profundidad, cámaras RGB y sistemas de posicionamiento, con el fin de obtener una visión completa y precisa del entorno. Asimismo, la navegación autónoma en entornos con presencia de objetos dinámicos constituye un área de especial interés, lo que ha impulsado investigaciones en visión por

computadora y técnicas de detección y seguimiento de objetos en movimiento. A través de un esfuerzo conjunto de la comunidad científica, se han logrado avances significativos que permiten a los vehículos autónomos afrontar desafíos del mundo real y avanzar hacia una navegación autónoma y segura.

2.2 Segmentación y reconstrucción

La segmentación y la reconstrucción son dos técnicas clave en el procesamiento de imágenes y señales. La segmentación de imágenes es una tarea fundamental en la visión artificial, ya que nos ayuda a comprender y analizar las imágenes dividiéndolas en regiones con características visuales similares. Esto permite identificar y separar objetos, fondos, y otros elementos importantes en la escena. La segmentación es esencial en muchas aplicaciones, como reconocimiento de objetos, seguimiento de objetos en movimiento y detección de anomalías.

Por otro lado, la reconstrucción se enfoca en crear una representación tridimensional de una escena o un objeto a partir de datos bidimensionales, como imágenes o videos. Esta técnica permite obtener una vista más completa y detallada del entorno, lo cual es muy útil en aplicaciones como realidad virtual, simulación de entornos, navegación de robots y creación de modelos 3D para ingeniería y diseño.

En las últimas décadas, se han desarrollado muchos métodos de segmentación de imágenes que se pueden clasificar en dos categorías: supervisados y no supervisados. Los métodos supervisados, como *Support Vector Machine* (*SVM*) y *Convolutional Neural Networks* (*CNN*), requieren un conjunto de datos para entrenar los algoritmos. Por otro lado, los métodos no supervisados se enfocan en clasificar automáticamente los píxeles de una imagen en diferentes etiquetas sin necesidad de un proceso de aprendizaje previo. Esta última opción es muy beneficiosa para aplicaciones de procesamiento de imágenes en tiempo real.

Anteriormente, la mayoría de las investigaciones se centraban en la segmentación de imágenes en escala de grises debido a las limitaciones de hardware y tecnología. Sin embargo, con el rápido desarrollo de equipos de hardware, la segmentación de imágenes a color ha ganado mucha atención y se ha vuelto cada vez más relevante en el campo del procesamiento de imágenes y señales [19].

Entre los métodos de segmentación no supervisada, se tienen los algoritmos de agrupamiento que se han utilizado con éxito en segmentación de imágenes y clasificación de datos. Aun así, es un tema desafiante, ya que es difícil lograr agrupamiento automático y proporcionar buenos resultados para la segmentación de imágenes. Los algoritmos de segmentación de imágenes basados en agrupamiento tienen tres ventajas: En primer lugar, pueden lograr una segmentación de imágenes no supervisada. En segundo lugar, son más robustos que otros algoritmos de segmentación de imágenes, como modelos de contorno activos, cortes de gráficos, caminantes aleatorios y fusión de regiones ya que requieren menos parámetros. Finalmente, el agrupamiento tiene una clara ventaja en la segmentación de imágenes multicanal porque es fácil aplicar algoritmos de agrupamiento en la clasificación de gran cantidad de datos [20],[21].

La segmentación semántica de imágenes es un método ampliamente estudiado que consiste en asignar una etiqueta semántica a cada píxel de la imagen [22], [23]. Esta técnica se distingue de la segmentación de instancia, que se enfoca en producir máscaras y clases para cada instancia [24]. Recientemente, ha ganado popularidad la segmentación panóptica, que combina la segmentación semántica a nivel de píxel y a nivel de instancia. Si bien existen algoritmos tradicionales de aprendizaje automático para enfrentar estos desafíos, las técnicas de *deep learning* han obtenido un éxito sin precedentes y superan a otros enfoques de manera significativa. Por ejemplo, las *CNN* se han convertido en uno de los algoritmos más impresionantes para tareas de reconocimiento de patrones en la gestión de imágenes. Además, *las Recurrent Neural Networks* (*RNNs*) se utilizan comúnmente para recuperar características contextuales y recordar información a lo largo del tiempo. Estos avances en el aprendizaje profundo han impulsado significativamente la investigación en segmentación semántica. Además, la disponibilidad de múltiples modalidades de detección ha fomentado el desarrollo de la fusión multimodal. Al aplicar la fusión multimodal profunda para la segmentación semántica de imágenes, se logra una mejora significativa en el rendimiento, aprovechando los beneficios de múltiples fuentes de información y generando automáticamente una predicción conjunta óptima [25].

El desarrollo de algoritmos de enfoque de región ha emergido rápidamente como una de las áreas de investigación más críticas en los últimos años [26]. La precisión de las técnicas de reconocimiento basadas en regiones ha impulsado el uso de estos algoritmos como una parte esencial en varios problemas de reconocimiento. El objetivo principal de estos algoritmos es extraer regiones relevantes de una imagen, manteniendo un número

adecuado de ellas para reducir el espacio de búsqueda y aumentar la precisión de detección.

El procesamiento de imágenes a partir de la nube de puntos es otro campo de investigación de alto interés. La segmentación de la nube de puntos [27] es un problema clave en la comprensión del contenido 3D. Los métodos existentes basados en redes neuronales profundas para la segmentación de nubes de puntos principalmente entrenan la red de manera supervisada, que depende en gran medida de una gran cantidad de nubes de puntos de entrenamiento de alta calidad etiquetadas manualmente. Sin embargo, es muy tedioso y lleva mucho tiempo asignar manualmente etiquetas de partes para cada punto en las nubes de puntos. Sin embargo, se pueden obtener fácilmente muchas nubes de puntos sin etiquetar a partir de escáneres 3D, Internet o reconstrucción.

El método de segmentación de nubes de puntos semi supervisado propuesto en [28] es capaz de utilizar tanto nubes de puntos etiquetadas como no etiquetadas durante el entrenamiento. Con el objetivo de aprovechar mejor las nubes de puntos no etiquetadas, se propone en [28] una arquitectura de segmentación contradictoria que incorpora la discriminación de confianza en las predicciones de etiquetas para estas nubes de puntos. Esta técnica permite seleccionar pseudo etiquetas en las nubes de puntos no etiquetadas con mayor confiabilidad, lo que mejora significativamente el rendimiento de la segmentación. En consecuencia, el método propuesto logra un uso más eficiente de las nubes de puntos no etiquetadas durante el entrenamiento, y el rendimiento de la segmentación se ve mejorado mediante el auto entrenamiento con predicciones de etiquetas más confiables.

En resumen, a pesar de los notables avances e investigaciones [29], todavía se identifican brechas desafiantes y aspectos por mejorar en el campo de estudio. Algunos de estos desafíos incluyen la complejidad de lidiar con oclusiones y variaciones impredecibles de la iluminación en la escena. La segmentación con información de profundidad también representa un reto adicional debido a su naturaleza más exigente.

Asimismo, desarrollar modelos de aprendizaje para abordar escenas como fondos abarrotados, patrones de movimiento complicados y diversidad de funciones espacio-temporales plantean desafíos significativos en la detección de secuencias de video.

Con datos de entrenamiento limitados, el aprendizaje de características que contengan información intra-frame, inter-frame y de movimiento a través de una red profunda para

la detección de secuencias de video se convierte en un desafío adicional.

Estas áreas representan temas de investigación desafiantes y aún no resueltos que requieren futuros esfuerzos para avanzar en el campo de manera significativa y seguir mejorando las técnicas y metodologías para abordar estas problemáticas.

Durante las últimas décadas, se han introducido una serie de métodos para identificar, emparejar y analizar los puntos clave de las imágenes y su posterior uso en SLAM, actualmente se utilizan métodos relativamente simples basados en gradientes de imagen y métodos más complejos de análisis de características que utilizan redes neuronales convolucionales profundas y multitarea [30]. El mapeo de puntos característicos usando redes neuronales permite lograr una solución más confiable, porque los puntos característicos, calculados usando métodos estándar, se agrupan principalmente en objetos de alto contraste, como césped, edificios y árboles. Los algoritmos basados en CNN están entrenados para detectar puntos característicos [31] de manera más inteligente y, como resultado, se obtiene una distribución más uniforme de los puntos característicos en la imagen.

Existe una arquitectura llamada *SuperGlue* modificada la cual toma puntos de interés y descriptores de dos imágenes como entrada. Esta arquitectura se basa en una red neuronal de grafos con unidades de atención que aumentan el campo receptivo de los descriptores y aseguran su interacción cruzada. La reconstrucción 3D dispersa se puede lograr durante el proceso SLAM mediante el cálculo y el seguimiento de puntos característicos con predicción de sensor inercial [32].

La reconstrucción 3D es una técnica clave en gráficos por computadora con diversas aplicaciones en realidad virtual, aumentada, animación, como también en el campo de la ingeniería civil [33] y demás aplicaciones [34]. En los últimos años, se han realizado muchos avances tanto en la calidad como en la velocidad de la reconstrucción. Desde el temprano éxito de KinectFusion, escanear con una cámara RGB-D básica y reconstruir la geometría capturada en línea se ha convertido en algo común. El trabajo posterior mejoró la escalabilidad del sistema para admitir escenas más grandes y detalles más finos mediante la introducción de nuevas estructuras de datos persistentes. Si bien la investigación sobre la reconstrucción y el modelado de escenas interiores estáticas ha madurado en los últimos años, la reconstrucción de objetos dinámicos [35] (por ejemplo, humanos, animales y otros objetos que se mueven libremente) sigue siendo un problema abierto tanto en las comunidades gráficas como robóticas (referido como SLAM

dinámico), así como también la reconstrucción en tiempo real [36],[37],[38].

Con el desarrollo de las tecnologías de Realidad Virtual, Realidad Aumentada y 5G, se ha ido elevando la demanda de técnicas de reconstrucción 4D (espacio + tiempo). Especialmente con los últimos avances de las cámaras RGB-D de nivel de consumidor, el interés ha ido en aumento en el desarrollo de tales técnicas de reconstrucción 4D para capturar varias escenas dinámicas [39].

Para lograr una reconstrucción de escena 4D (3D + tiempo), algunos investigadores estimaron la trayectoria de movimiento de cada objeto que se movía en primer plano alrededor del robot y reconstruyeron su modelo 3D basándose en la segmentación semántica obteniendo información sobre objetos en movimiento. Sin embargo, su ámbito de aplicación es limitado porque muchos objetos en movimiento son desconocidos y no pueden segmentarse semánticamente en entornos prácticos donde este tipo de método no es válido. Por el contrario, también se han desarrollado métodos basados en la segmentación de múltiples movimientos, que agrupan puntos del mismo movimiento en una instancia de parámetro de modelo de movimiento, segmentando así los múltiples modelos de movimiento correspondientes a objetos en movimiento uno por uno en el escenario dinámico. Estos métodos estiman la trayectoria de la cámara y los objetos en movimiento sin depender de las señales semánticas iniciales y son más sólidos en las escenas del mundo real [40].

Sin embargo, todavía existen dos problemas importantes relacionados con la reconstrucción dinámica de escenas en 4D. En primer lugar, todos los métodos de salida siguen siendo vulnerables al movimiento rápido y ocluido de la escena dinámica. Los movimientos rápidos introducen desenfoque de movimiento y pueden degradar gravemente la precisión de seguimiento de las correspondencias entre fotogramas, lo que afecta a la fusión de la geometría [41].

Otros métodos propuestos son los algoritmos estéreo multivista (MVS) basados en *PatchMatch* que han logrado un gran éxito en tareas de reconstrucción de escenas a gran escala. Sin embargo, la reconstrucción de planos sin textura a menudo falla, ya que los métodos de medición de similitud pueden volverse ineficaces en estas regiones. Por lo tanto, se opta por implementar una estrategia de inferencia de hipótesis de profundidad potencial para hacer que el resultado de la reconstrucción sea más completo. La estrategia consta de dos pasos. En primer lugar, para los píxeles que no se reconstruyen con éxito, se generan múltiples hipótesis de profundidad potencial utilizando valores de píxeles

próximos reconstruidos con profundidad precisa. En segundo lugar, las hipótesis de profundidad se seleccionan utilizando el campo aleatorio de Markov (MRF). La estrategia puede aumentar significativamente la integridad de la reconstrucción [42].

2.3 SLAM

SLAM es un área de investigación en creciente interés. En [43], se examinan varios sistemas SLAM, incluido el V-SLAM, que utiliza cámaras RGB, cámaras estéreo, cámaras omnidireccionales y cámaras RGB-D para lograr la localización, mapeo y orientación de sistemas autónomos, mediante sistemas monoculares, estereoscópicos y mixtos. Los investigadores han estructurado estos sistemas en diversas arquitecturas para diseñar algoritmos efectivos. Además, se presenta una visión general de las diferentes métricas en SLAM y sus líneas de investigación. Aunque el error relativo de posicionamiento se usa comúnmente para medir la precisión de la trayectoria, se sugiere la necesidad de nuevas métricas para evaluar los algoritmos de reconstrucción con mayor precisión. Otro estudio [44] destaca la importancia de determinar el posicionamiento y la reconstrucción de ambientes en el desarrollo del SLAM en vehículos autónomos robóticos.

2.3.1 Reconstrucciones ambientes estáticos en SLAM

El desarrollo de sensores para la percepción del entorno ha permitido la reconstrucción de escenas mediante sistemas de visión artificial y V-SLAM. Esto ha impulsado diferentes desarrollos en varias áreas de aplicación.

Uno de los enfoques más comunes es la aplicación de redes neuronales para la reconstrucción de escenas. En [45], se utilizó una *Detail-preserving network* (DPNet) para extraer profundidad monocular a partir de imágenes RGB de una sola cámara. En otro estudio [46], se combinó la predicción por redes neuronales convolucionales con la reducción de nubes de puntos con *voxelgrid* y la generación de normales para crear bocetos 3D de múltiples vistas.

Otro enfoque es la utilización de sistemas de extracción de características, como el método de red estéreo [47] de vista múltiple que utiliza la extracción de volumen piramidal para la reconstrucción 3D de escenarios exteriores.

También se han utilizado algoritmos matemáticos y estadísticos para la reconstrucción de escenas. En un método propuesto [48], se utilizó una cámara RGB-D para capturar los frames sucesivos, que se procesaron con el algoritmo *Ransac* y el Iterative *Closest Point* (ICP) para reconstruir escenas interiores.

Otro enfoque es la reconstrucción de escenas en línea mientras se recorre la escena. Por ejemplo, en un método [49] de reconstrucción de escenas estáticas con múltiples robots, varios robots van aportando segmentos del mapa captado, que se fusionan en tiempo real para generar un mapa del entorno en 2D.

Finalmente, existe la posibilidad de utilizar la fusión sensorial para la reconstrucción de escenas, como se ha demostrado en [50] que fusiona los datos de un LIDAR 2D y RGB-D para mejorar la detección de características del entorno.

Aunque estos métodos son muy prometedores, algunos de ellos pueden ser costosos en términos de computación [51],[52]. Sin embargo, cada vez se están desarrollando más soluciones que permiten una reconstrucción de escenas en tiempo real y con un menor costo computacional.

2.3.2 SLAM en ambientes dinámicos

Los sistemas autónomos enfrentan dificultades al operar en entornos dinámicos (conocido como SLAM dinámico) debido a la constante movilidad de los objetos, lo que genera registros cambiantes en los sensores. La detección de objetos dinámicos en entornos reales implica considerar factores como la velocidad, la dirección y la distancia a los sensores, lo que aumenta la complejidad de esta tarea.

El SLAM dinámico ha sido objeto de pocos avances debido a su complejidad. En [53] se ha evaluado la capacidad de detección, adaptabilidad, novedad y rapidez de 119 algoritmos de detección de objetos dinámicos en una secuencia de imágenes 2D. Sin embargo, esta área de estudio aún tiene muchas limitaciones que deben ser exploradas para ofrecer soluciones.

En cuanto a la detección de objetos dinámicos, se han planteado dos categorías [54] de detección: detección por cajas delimitadoras en 2D o 3D y estimadores de postura de 6D, en los que se aplican clasificadores por sistemas de inteligencia artificial. En algunos

experimentos, se han utilizado CNN [55] y detección por bolsa de palabras sobre nubes detectadas por LIDAR 3D. Estos métodos se han evaluado mediante la variación de la densidad de puntos en las nubes y se han aplicado sobre distintos conjuntos de datos.

Se han realizado varios experimentos de detección de objetos en movimiento mediante cubos de ocupación sobre nube de puntos [56], técnicas de visión en imágenes de profundidad y RGB, y el método *onboard diagnostics* (OBD) [57]. En estos experimentos se han utilizado nubes de puntos reducidas de entornos dinámicos exteriores con múltiples objetos dinámicos, y se ha evaluado la precisión del sistema de clasificación y la segmentación del objeto dinámico.

Se ha propuesto un método para la eliminación de objetos dinámicos mediante la fusión de imágenes, aplicado fuera de línea sobre un conjunto de datos [58]. Sin embargo, la detección de objetos dinámicos sigue siendo un área de estudio compleja y se necesita seguir investigando, comenzando por la detección a bajas velocidades y con objetos de grandes dimensiones, con el objetivo de avanzar en la detección y eliminación de objetos dinámicos y la reconstrucción de la escena.

2.3.3 Pointcloud_to_Laserscan

Pointcloud_to_laserscan es un algoritmo utilizado en robótica para convertir datos de nubes de puntos en datos de escaneo láser [59][60]. Esta herramienta es muy útil en la navegación autónoma de robots, ya que los sensores láser son muy precisos y proporcionan información detallada sobre la posición y la orientación del robot [61].

La conversión de datos de nubes de puntos a datos de escaneo láser es importante porque los sensores de láser son muy comunes en la robótica y se utilizan para crear mapas de entornos y evitar obstáculos [62]. Sin embargo, no todos los robots tienen sensores láser integrados. Algunos robots tienen sensores de nubes de puntos, que son más económicos, pero no son tan precisos como los sensores de láser [59].

La herramienta pointcloud_to_laserscan ayuda a resolver este problema, ya que permite que los robots con sensores de nubes de puntos utilicen algoritmos y herramientas diseñados para robots con sensores de láser [60].

Para comprender mejor pointcloud_to_laserscan, es importante comprender cómo

funciona la detección de obstáculos en la robótica. La detección de obstáculos se realiza a través de la creación de un mapa del entorno del robot y la identificación de obstáculos en ese mapa. Los sensores de láser y de nubes de puntos se utilizan para crear este mapa [62] [61].

Los sensores de láser emiten un haz de luz y miden el tiempo que tarda en regresar al sensor después de rebotar en un objeto. Con esta información, el sensor puede determinar la distancia del objeto y su posición en relación con el robot [62]. Así mismo, los sensores de nubes de puntos tienen un funcionamiento similar, pero en lugar de emitir un haz de luz, capturan múltiples puntos de luz que se reflejan en los objetos. A partir de estos puntos de luz, se puede construir una imagen tridimensional del entorno del robot [61].

Para convertir los datos de nubes de puntos en datos de escaneo láser, pointcloud_to_laserscan utiliza un algoritmo que simula los resultados que se obtendrían si se estuvieran utilizando sensores de láser [60]. Este algoritmo mapea los puntos de la nube de puntos en un plano bidimensional y calcula la distancia de cada punto al robot. Luego, el algoritmo asigna un valor de intensidad a cada punto, lo que permite distinguir entre objetos sólidos y espacios abiertos [59].

Supongamos que tenemos una nube de puntos en tres dimensiones dada por P(x, y, z), donde (x, y, z) representan las coordenadas de cada punto en el espacio tridimensional. Queremos convertir estos puntos en un escaneo láser, que generalmente se representa como un conjunto de ángulos y distancias (θ, r), donde θ es el ángulo horizontal y r es la distancia medida desde el origen en esa dirección. La conversión se puede realizar de la siguiente manera:

Para cada punto (x, y, z) en la nube de puntos:

- Calcule el ángulo θ usando la fórmula atan2(y, x), que es la función arco tangente de la relación entre las coordenadas x e y del punto.
- Calcule la distancia r usando la distancia euclidiana desde el origen: $r = \sqrt{x^2 + y^2}$.

El resultado será un conjunto de pares (θ, r) que representan el escaneo láser. En términos matemáticos, podemos definir la transformación de la nube de puntos (P) a datos de escaneo láser (L) como una función T:

T: P (x, y, z) -> L (θ, r)

Donde:

θ = atan2(y, x)

$r = \sqrt{x^2 + y^2}$.

En resumen, pointcloud_to_laserscan es una herramienta útil para la navegación autónoma de robots, ya que permite que los robots con sensores de nubes de puntos utilicen herramientas y algoritmos diseñados para robots con sensores de láser.

2.3.4 Métricas

El desarrollo de sistemas SLAM ha requerido la evaluación de diferentes métricas para medir la precisión y el rendimiento de los algoritmos. El error cuadrático medio, el error absoluto y la desviación estándar son algunas de las métricas comúnmente utilizadas en la evaluación de sistemas de seguimiento de la ruta de los instrumentos y del recorrido de los objetos dinámicos en la escena [63]. Estas métricas se aplican ampliamente en SLAM [43] y permiten comparar diferentes sistemas y métodos.

En sistemas SLAM 2D, se ha desarrollado un método de mapeo de escenas de bajo costo computacional [64]. En la evaluación de este sistema se utiliza la superposición del mapa de objetos en la escena real para realizar una comparativa visual y se establecen los porcentajes de error en la medida de los fragmentos de la escena. Otros sistemas SLAM 2D han sido evaluados mediante la comparativa del error absoluto de posicionamiento en escenarios interiores [65]. En este tipo de evaluaciones, se han obtenido resultados significativamente mejores que otros algoritmos evaluados.

Los sistemas SLAM 3D de reconstrucción requieren métricas diferentes para evaluar su precisión y rendimiento. Algunas de estas métricas incluyen la necesidad de la escala absoluta, la generación de mapas de alta resolución y mapas 3D, y la capacidad de generar mapeo RGB de puntos [66]. La precisión de registro entre cada escaneo estático y la comparación cualitativa de las reconstrucciones obtenidas también son utilizadas para evaluar estos sistemas. En algunos casos, se evalúa el crecimiento de la nube de puntos y

se realiza un análisis de la tasa de recuperación [67].

Los sistemas SLAM 3D de reconstrucción con objetos dinámicos requieren evaluaciones específicas para medir su rendimiento y precisión en la detección y seguimiento de objetos en movimiento. En algunos estudios se evalúa el consumo de recursos computacionales, el porcentaje de uso y la detección del objeto dinámico de forma cualitativa [68]. Sin embargo, es importante destacar que estas medidas no son suficientes para establecer la calidad de reconstrucción. La evaluación de la trayectoria del objeto dinámico [63] se realiza a través del error absoluto, el error cuadrático medio y la desviación estándar, y se utiliza el error absoluto de trayectoria y error relativo de posicionamiento para evaluar los puntos suministrados. Además, se realiza la segmentación sobre la RGB y se compara de forma cualitativa [69].

2.4 Conclusiones

En SLAM, el factor común es el costo computacional asociado con la ejecución de los algoritmos de procesamiento de pointcloud. Este costo debe ser limitado para permitir la ejecución en línea.

Aunque los sistemas de reconstrucción han tenido gran éxito en la reconstrucción de ambientes estáticos, no son aplicables directamente a escenas con objetos dinámicos debido a la oclusión que estos generan.

Las métricas existentes para evaluar el comportamiento de los algoritmos son insuficientes ya que no proporcionan un valor que relacione la similitud entre la escena reconstruida y la escena de referencia.

Actualmente, no existe un método de bajo costo computacional que permita generar una reconstrucción 3D de escenas con presencia de objetos dinámicos.

Capítulo 3.
Segmentación de Zonas de Puntos de Alta Variabilidad

En este capítulo se despliega con detalle el algoritmo paso a paso destinado a la segmentación por zonas de puntos de alta variabilidad. Iniciando en la Sección 3.1, se exponen las fases fundamentales del método de reconstrucción dinámica. Este proceso es acompañado por una imagen ilustrativa que amplifica la comprensión del lector respecto a cada etapa. Cada una de estas fases, como el preprocesamiento, interpolación y extrapolación, promedio móvil y desviación, ángulo de detección, etc. Son etapas que se entrelazan para componer el algoritmo de segmentación de zonas de puntos de alta variabilidad.

En la continuación, en la Sección 3.2, se aborda la implementación, donde se retoman algunos pasos de la Sección 3.1, pero se amplía el contenido con la inclusión de los procesos relativos a la fusión de datos. La Sección 3.3 introduce los elementos que conforman los materiales utilizados, como los sensores empleados y el proceso de calibración de la cámara RGB-D. Luego, en la Sección 3.4, se sumergen en los datos generados, describiendo los datasets, los objetos dinámicos incorporados, las escenas seleccionadas con sus características específicas y los recorridos realizados. En conjunto, este capítulo proporciona una guía exhaustiva a través del algoritmo de segmentación y su aplicación, con una mirada detallada a los componentes clave y los elementos prácticos involucrados en el proceso.

3.1 Etapas del método de reconstrucción dinámica

El método propuesto en [70] para la Reconstrucción Dinámica basada en la Segmentación

de Zonas de Puntos de Alta Variabilidad (Rec-HV) utiliza la desviación estándar de los datos del LIDAR 2D virtual en cada ciclo de ejecución. Este cálculo se utiliza para determinar el ángulo de detección de la zona dinámica y aplicarlo en la segmentación de la *PCL* del RGB-D. De esta manera, se eliminan los puntos del objeto dinámico en cada ciclo, lo que permite la reconstrucción de la escena estática mediante la concatenación de *PCL* sucesivas. Para lograr este proceso, se llevaron a cabo diferentes procedimientos estadísticos y matemáticos en los datos del LIDAR virtual y del RGB-D.

Las etapas del método Rec-HV anterior y actual se presentan en la Figura 1 y en las secciones siguientes se explica en detalle cada uno de los pasos del método actual, esta información es adaptada de [70].

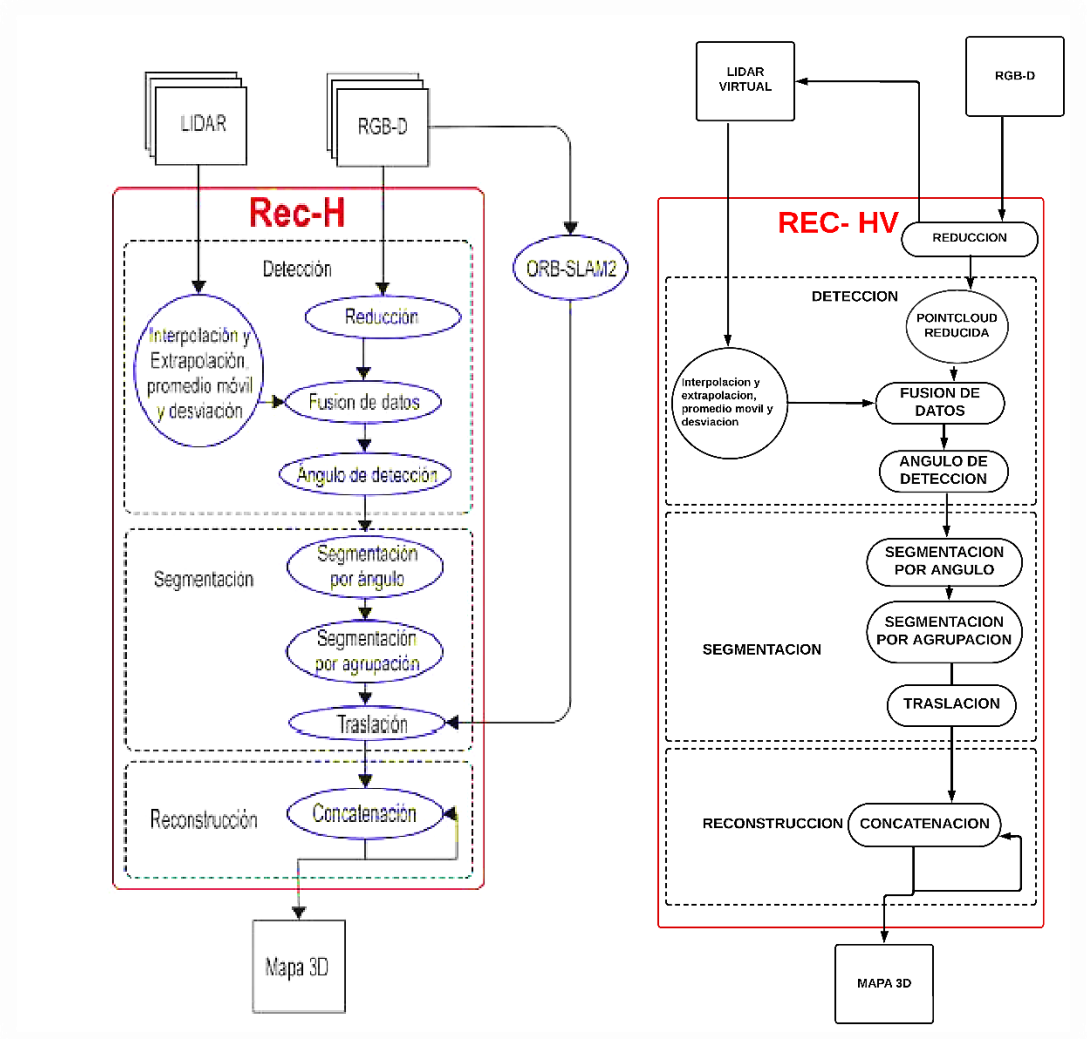


Figura 1. Diagramas: anterior/actual de las etapas del método Rec-HV

3.1.1 Preprocesamiento

Los datos del LIDAR virtual se reciben cada ciclo de medición. Estos están constituidos por: el vector de la profundidad ($\boldsymbol{r}$) de longitud (l), el ángulo de inicio de muestreo (θmin) y el ángulo de incremento ($\Delta\theta$). A partir de estos datos se construye un vector de ángulos (θ) de longitud l, ver (1).

$$\begin{aligned} \boldsymbol{r} &= [r_1, r_2, r_3 \dots r_l] \\ \theta_i &= \theta \min + \Delta\theta * i \\ \boldsymbol{\theta} &= [\theta_1, \theta_2, \theta_3 \dots \theta_l] \end{aligned} \tag{1}$$

Los datos del LIDAR virtual están constituido por una medición de 57°, lo que establece l igual a 57°, a partir de este se construye el vector de ángulos θ y se recorta el vector profundidad r.

3.1.2 Interpolación y extrapolación

El vector r puede contener valores de profundidad infinitos debido a que en el LIDAR virtual se establece un rango máximo y se representa la infinitud como verdadera cuando los valores exceden ese límite. Para corregir esta situación y reemplazar los valores infinitos por valores reales, se emplea la técnica de interpolación y extrapolación, en función de la posición dentro del vector r. En específico, se utiliza la interpolación para ajustar los valores entre dos puntos reales, incluso si estos no son adyacentes. Por otro lado, la extrapolación entra en juego para sustituir los datos que presentan valores infinitos en los extremos del vector r.

La interpolación y extrapolación se realizan encontrando una línea de tendencia con los vecinos más cercanos y proyectándola sobre los puntos con valores atípicos. De esta manera, se obtiene un conjunto completo de datos del LIDAR virtual, compuesto por parejas ordenadas que corresponden a la posición dentro del vector θ y r.

3.1.3 Promedio móvil y desviación

El promedio móvil se calcula aplicando una ventana de n ciclos a cada punto del vector de profundidad $\boldsymbol{r}$ del ciclo actual, junto con los $n-1$ ciclos anteriores inmediatos al ciclo actual, ver ecuación (2). Luego, se determina un vector de desviación estándar ($\boldsymbol{dr}$) para estos mismos datos, que consta de la desviación estándar de cada punto en los n ciclos ver ecuación (3).

$$\bar{r_k} = \frac{1}{n}\sum_{k-n+1}^{k} \boldsymbol{r}_{k-n+1} \tag{2}$$

$$\mathbf{d}\boldsymbol{r_k} = \sqrt{\frac{1}{n}\sum_{k-n+1}^{k} (\boldsymbol{r}_{k-n+1} - \bar{r}_k)^2} \tag{3}$$

Se tienen dos vectores de datos provenientes del LIDAR virtual: $\boldsymbol{r_k}$, que es el vector de profundidad para el k-ésimo ciclo de lectura, y $\overline{\boldsymbol{r_k}}$, que es su promedio. Además, $\boldsymbol{dr}$ es la desviación estándar para cada valor de r_k en el vector de profundidad del k-ésimo ciclo de lectura. Estos tres vectores de datos se obtienen en cada ciclo de datos LIDAR virtual y también se incluye el vector de ángulos $\boldsymbol{\theta_k}$. Esto se muestra en la ecuación (4) para el k-ésimo ciclo.

$$\begin{aligned} \bar{\boldsymbol{r}}_k &= [\bar{r_1}, \bar{r}_2, \bar{r}_3 \dots r_l] \\ \boldsymbol{\theta_k} &= [\theta_1, \theta_2, \theta_3 \dots \theta_l] \\ \mathbf{d}\boldsymbol{r_k} &= [\mathrm{d}r_1, \mathrm{d}r_2, dr_3 \dots \mathrm{d}r_l] \end{aligned} \tag{4}$$

3.1.4 Ángulo de detección

En cada ciclo, se determinan los máximos de variabilidad a partir de los valores de $\overline{\boldsymbol{r_k}}$, $\boldsymbol{\theta_k}$ y $\boldsymbol{dr_k}$. Estos máximos se obtienen de $\overline{\boldsymbol{r_k}}$ cuyos valores de dr son los máximos (dr_{max1} y dr_{max2}) y no son consecutivos en el vector dr. Estos máximos tienen posiciones $i_1 1$ e i_2 en el vector dr, a partir de las cuales se obtienen los ángulos de detección (α1 y α2), como se muestra en la ecuación (5).

$$\alpha 1 = \theta \min + i_1 * \Delta\theta$$
$$\alpha 2 = \theta \min + i_2 * \Delta\theta \quad (5)$$

3.1.5 Reducción de la PCL (Point Cloud)

La nube de puntos RGB-D (PCL_{RGB-D}) puede contener hasta un máximo de 307.200 puntos (para el Kinect v1.0), aunque este tamaño puede variar dependiendo del entorno en el que se encuentre. Aunque no se capturen todos los puntos, sigue siendo un gran volumen de datos. Para procesarlos, primero se selecciona solo la información de profundidad de cada punto en la PCL_{RGB-D}, y luego se aplica un voxelgrid que promedia los puntos en segmentos de una grilla con un tamaño definido (t_{vg}). Esto resulta en una versión reducida de la nube de puntos llamada$PCL_{Reducida}$, que contiene un menor número de puntos.

3.1.6 Segmentación por ángulo

En el proceso de segmentación por ángulo detectado, se realiza la segmentación de la nube de puntos $PCL_{Reducida}$, extrayendo la región que está delimitada por los ángulos de detección (α1 y α2). Para lograr esto, es necesario calcular el ángulo en el plano XY, con respecto al eje y, para cada punto de la PCL, utilizando la ecuación

$$r_{pt} = \sqrt{x_{pt}^2 + y_{pt}^2 + z_{pt}^2}$$
$$\alpha_{pt} = \cos\left({}^{y_{pt}}/_{r_{pt}}\right) \quad (6)$$

Se calcula el ángulo α_{pt} para cada punto de la $PCL_{Reducida}$ en función de su radio r_{pt}y su posición con respecto al eje y. Luego, se lleva a cabo la segmentación de los puntos en cada ciclo, donde aquellos que caen dentro del rango de ángulos α1 y α2 se consideran como puntos pertenecientes a la zona dinámica del ciclo k actual (PCL_{ZD}), mientras que los puntos que se encuentran fuera de este rango angular se consideran como puntos pertenecientes a las zonas estáticas del ciclo k actual (PCL_{Zek}).

3.1.7 Segmentación por agrupación de profundidad

En el proceso de segmentación por agrupación por profundidad, se realiza la segmentación de la PCL_{ZD} que contiene los puntos del objeto dinámico y los alrededores de la zona estática ocluida. Esto se logra seleccionando los grupos de puntos que tienen un valor de coordenada x menor que el punto más lejano (x_{max}) del grupo cuyo centroide está más cerca de los sensores. Estos puntos se consideran como los puntos pertenecientes al objeto dinámico (PCL_{Din}). Además, se identifican los grupos de puntos cuyos centroides se encuentran a mayor profundidad que x_{max}, los cuales constituyen la zona estática que rodea a la zona ocluida por la PCL_{Din}, y se denominan como la nube de puntos PCL_{Oek}.

Posteriormente, se realiza la concatenación de la PCL_{Zek} y la PCL_{Oek} para obtener la nube de puntos de la zona estática (PCL_{est-k}) del ciclo k-ésimo, según la ecuación (7).

$$
\begin{aligned}
PCL_{zek} &= \begin{bmatrix} x_1 & y_1 & z_1 \\ \vdots & \vdots & \vdots \\ x_s & y_s & z_s \end{bmatrix} \\
PCL_{Oek} &= \begin{bmatrix} x_1 & y_1 & z_1 \\ \vdots & \vdots & \vdots \\ x_w & y_w & z_w \end{bmatrix} \\
PCL_{est-k} &= PCL_{zek} \parallel PCL_k \\
p &= s + w \\
PCL_{est-k} &= \begin{bmatrix} x_1 & y_1 & z_1 \\ \vdots & \vdots & \vdots \\ x_p & y_p & z_p \end{bmatrix}
\end{aligned}
\tag{7}
$$

La PCL_{est-k} se refiere a la escena estática de la cual se han eliminado los puntos correspondientes al objeto dinámico.

3.1.8 Traslación

La PCL_{est-k} se posiciona con respecto al centro óptico de los instrumentos como marco de referencia, sin embargo, para llevar a cabo la reconstrucción, es necesario que todas las nubes de puntos estén en el mismo marco de referencia. Para lograr esto, se debe tener la posición de los instrumentos en cada ciclo en referencia a una posición inicial, y luego se

realiza la traslación de la PCL a la base de coordenadas del punto de origen del movimiento. Esto permite determinar la posición absoluta de cada punto en la PCL_{est-k}. El cambio de base de coordenadas se aplica multiplicando cada punto por la matriz de transformación homogénea correspondiente a cada ciclo k, como se muestra en la (Figura 2).

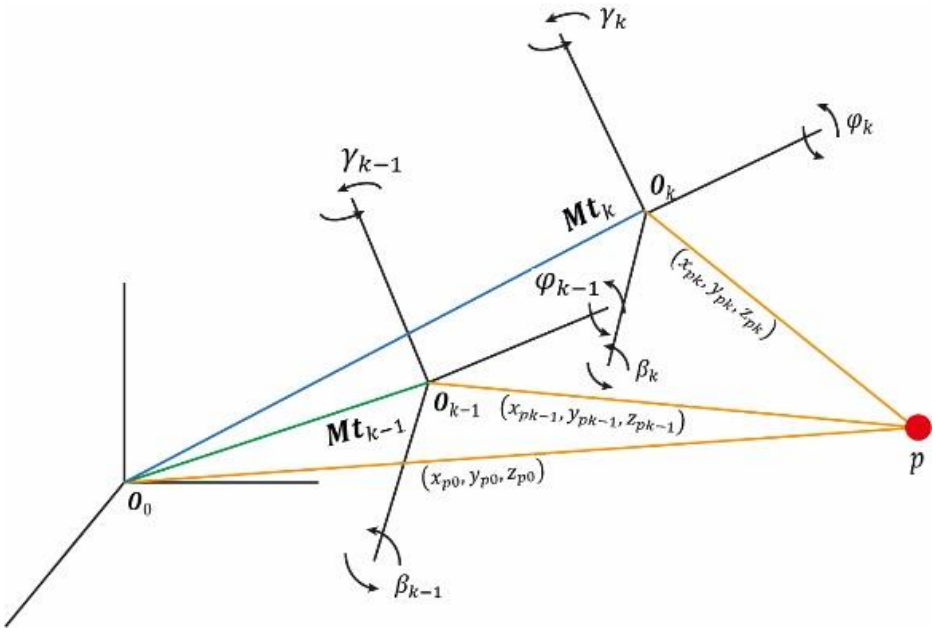


Figura 2. Vector traslación para cada marco de referencia con respecto al origen.

En la Figura 2, p es un punto en la escena estática. Mt_k es la matriz de transformación homogénea para el k-ésimo ciclo, mientras que Mt_{k-1} es la matriz de transformación homogénea para el ciclo $k-1$. x_{pk}, y_{pk} y z_{pk}, son las coordenadas de un punto p que pertenece a la PCL_{est-k}. del ciclo k. Por otro lado, x_{pk-1}, y_{pk-1} y z_{pk-1} son las coordenadas de un punto que pertenece a la PCL del ciclo $k-1$. O_0 es el punto de origen del marco de referencia cuando k es igual a cero, mientras que O_k es el punto de referencia para el ciclo k, y O_{k-1} es el punto de referencia para los puntos registrados en el ciclo $k-1$.

Cada ciclo k tiene una matriz de transformación homogénea M_{tk} correspondiente, la cual está determinada por los parámetros de traslación (Vt_k). Vt_k contiene las coordenadas de traslación P_x, P_y yP_z, así como los ángulos de rotación φ, β y γ para cada eje de coordenadas x, y, y z, obtenidos a través de la odometría visual. La traslación se realiza a partir de los puntos que se encuentran en la PCL_{est-k}, los cuales están referidos al marco de coordenadas del ciclo k , O_k. Estos puntos deben ser trasladados al marco de referencia O_0 mediante la aplicación de la matriz M_{tk}, de acuerdo con la ecuación (8), para cada punto

de laPCL_{est-k}.

$$PCL_{est-k} = \begin{bmatrix} x_{r1} & y_{r1} & z_{r1} \\ \vdots & \vdots & \vdots \\ x_{rp} & y_{rp} & z_{rp} \end{bmatrix}$$

$$v_{rp} = \begin{bmatrix} x_{rp} \\ y_{rp} \\ z_{rp} \\ 1 \end{bmatrix}$$

$$\boldsymbol{Mt}_k = \begin{bmatrix} \cos\varphi\cos\beta & \cos\varphi\sin\beta\sin\gamma + \sin\varphi\cos\gamma & \cos\varphi\sin\beta\cos\gamma + \sin\varphi\sin\gamma & P_x \\ \sin\varphi\cos\beta & \sin\varphi\sin\beta\sin\gamma + \cos\varphi\cos\gamma & \sin\varphi\sin\beta\cos\gamma - \cos\varphi\sin\gamma & P_y \\ -\sin\beta & \cos\beta\sin\gamma & \cos\beta\cos\gamma & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (8)$$

$$v_{rtp} = \boldsymbol{Mt}_k \,.\, v_{rp}$$

$$v_{rtp} = \begin{bmatrix} x_{rtp} \\ y_{rtp} \\ z_{rtp} \\ 1 \end{bmatrix}$$

$$PCL_k = \begin{bmatrix} x_{rt1} & y_{rt1} & z_{rt1} \\ \vdots & \vdots & \vdots \\ x_{rtp} & y_{rtp} & z_{rtp} \end{bmatrix}$$

Durante este proceso, cada vector v_{rp}, compuesto por las coordenadas X_{rp}, Y_{rp} y Z, se multiplica por la matriz M_{tk} para obtener v_{rtp}, que contiene las coordenadas X_{rtp}, Y_{rtp} y Z_{rtp}. Estas coordenadas conforman laPCL_k, que es la versión reducida de la nube de puntos del k-ésimo ciclo con respecto al marco de referencia O_0.

3.1.9 Reconstrucción

Durante el proceso de reconstrucción se lleva a cabo la concatenación de las escenas de cada ciclo, con el objetivo de obtener una escena estática reconstruida sin obstrucciones. Para lograr esto, es necesario completar la información faltante en la PCL_k del ciclo actual, la cual pudo haber sido ocultada por un objeto dinámico. Para ello, se concatena la PCL_k

actual con la PCL final del ciclo previo k-1 (PCL_{kt-1}), tal como se muestra en la ecuación (9)

$$\mathrm{PCL_{kT-1}} = \begin{bmatrix} x_1 & y_1 & z_1 \\ \vdots & \vdots & \vdots \\ x_q & y_q & z_q \end{bmatrix}$$

$$\mathrm{PCL_{Trans.}} = \mathrm{PCL_{kT-1}} \parallel \mathrm{PCL_k} \tag{9}$$

$$\mathrm{PCL_{Trans.}} = \begin{bmatrix} x_1 & y_1 & z_1 \\ \vdots & \vdots & \vdots \\ x_{p+q} & y_{p+q} & z_{p+q} \end{bmatrix}$$

Luego del proceso mencionado, se genera una PCL transitoria (PCL_{Trans}) completa del ciclo actual. Esta PCL se caracteriza por presentar zonas superpuestas, donde PCL_{kt-1} y PCL_k coinciden y comparten puntos que corresponden a la misma zona de la escena estática. Esto se puede visualizar en la siguiente (Figura 3).

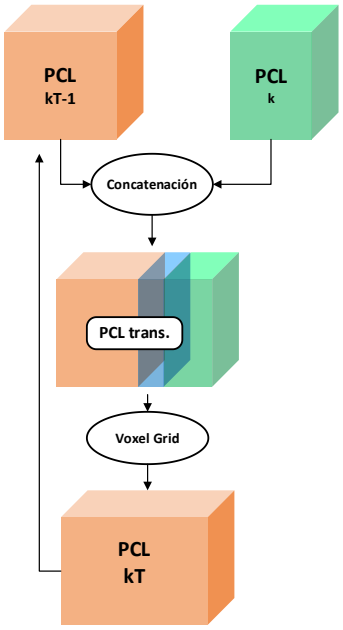


Figura 3. Concatenación y reducción de PCL para la reconstrucción de escenas estáticas

La concatenación sucesiva de los ciclos anteriores permite obtener la PCL_{kt}, que es la PCL reconstruida del ciclo actual.

3.1.10 Correlación por vistas localizadas

En esta investigación se propone una medida de correlación entre la PCL_{kt} obtenida mediante el método Rec-HV y una PCL de referencia (PCL_{ref}) capturada en un fotograma de la escena estática sin objetos dinámicos. Para lograr una correlación precisa, ambas PCL deben tener el mismo marco de referencia. Las PCL contienen 3 columnas: X, Y y Z, donde se registran las coordenadas de cada punto. Primero se organiza la PCL en orden descendente según la columna X, y luego se obtienen el valor máximo ($X_{max-cor}$) y mínimo ($X_{min-cor}$) de la columna X de profundidad de la escena para realizar una transformación a escala de grises. Posteriormente, se escala las columnas Y y Z para transformarlas y obtener valores que corresponden a píxeles, tal como se muestra en la ecuación (10)

$$Color = 255 * (X\text{-}x_{min-cor})/x_{max-cor}$$

$$Y_{px} = 100 * Y$$

$$Z_{px} = -100 * Z \tag{10}$$

$$\text{Pt} = \begin{bmatrix} y_{px(1)} & z_{px(1)} & color_{(1)} \\ \vdots & \vdots & \vdots \\ y_{px(m)} & z_{px(m)} & \text{color}_{(m)} \end{bmatrix}$$

Se aplica el proceso de transformación a ambas PCL , PCL_{ref} y PCL_{kt}, lo que genera las matricesPT_{kt} y PT_{ref}. Estas matrices son una transformación de la matriz de tres dimensiones de la PCL a una matriz de dos dimensiones y escala de grises. Una vez obtenidas las matrices en dos dimensiones, se procede a comparar píxel por píxel para establecer la similitud entre ambas matrices. La comparación arroja el valor de la correlación lineal de las dos escenas.

3.2 Implementación

El sistema Rec-HV se desarrolla en Linux con Python 3.8 y utiliza diversas bibliotecas para el procesamiento matemático, estadístico y de ciencia de datos, así como el *framework* ROS v1.0 (*Robotic Operation System*), que utiliza una arquitectura de nodos que intercambian mensajes mediante líneas de publicación y suscripción. La arquitectura de software de Rec-HV está diseñada para conectarse en paralelo con los instrumentos y

ejecutar los nodos correspondientes en cascada, tal como se muestra en la Figura 4.

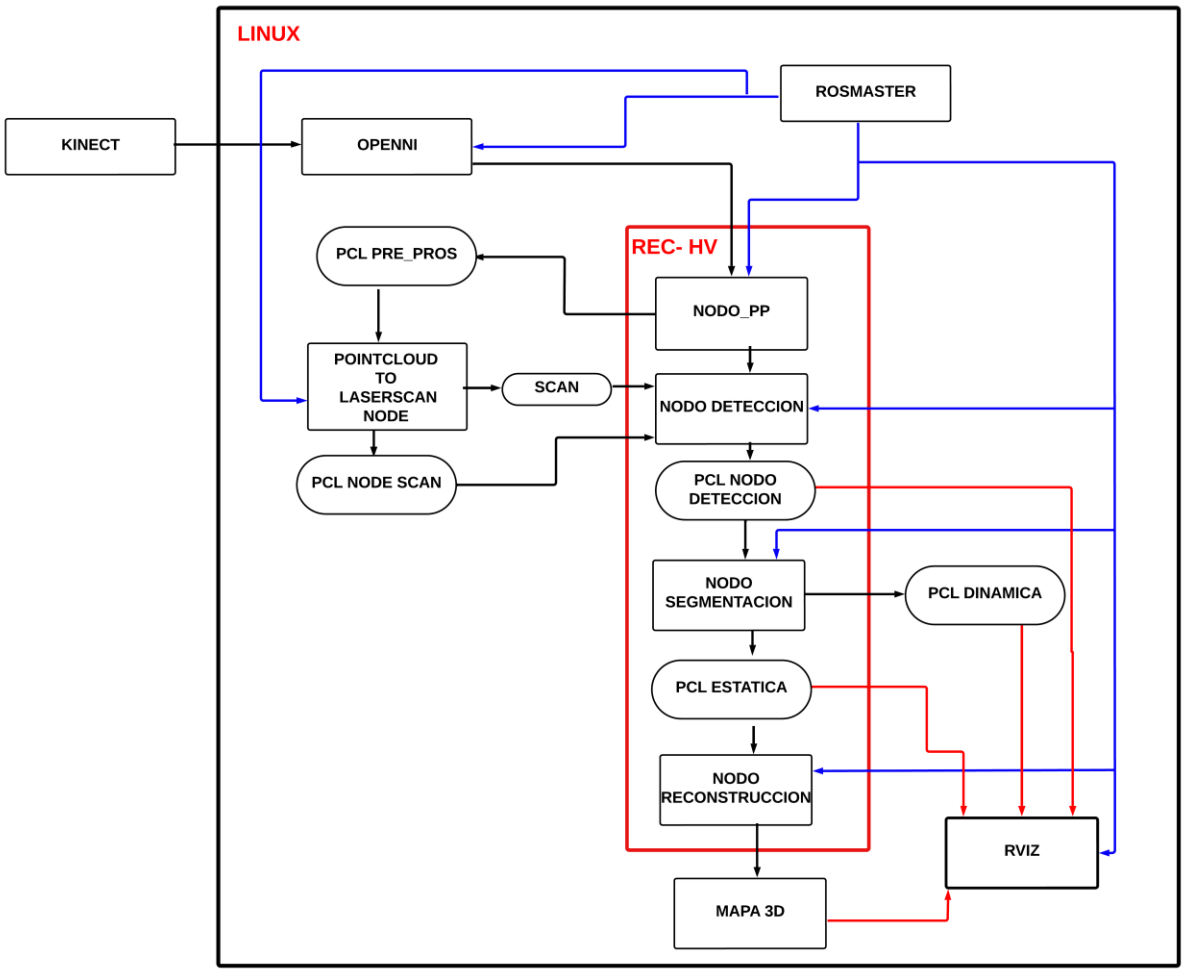


Figura 4. Arquitectura del sistema Rec-HV.

En la configuración de la Figura 4, se destaca la presencia del nodo principal ROSMASTER, cuya función es habilitar los canales necesarios para el funcionamiento del sistema. Una vez activado, se ejecuta el nodo OpenNi para establecer la conexión con el sensor RGB-D y permitir diversas publicaciones, como la PointCloud2 y la imagen RGB. A continuación, entra en acción el nodo PP, que hace referencia al nodo preprocesamiento, encargado de llevar a cabo un preprocesamiento de los datos y proporcionar una PointCloud2 procesada al nodo "pointcloud to laser scan", la cual simula el funcionamiento del LIDAR virtual.

Estos mensajes conforman el conjunto de datos que el sistema Rec-HV recibe y utiliza para iniciar su procesamiento. Por último, se inician los nodos de detección, segmentación y

reconstrucción, lo que da lugar a la ejecución completa del sistema.

3.2.1 Nodo detección

El nodo de detección se conecta a dos líneas de entrada de mensajes que proporcionan datos tipo LaserScan del LIDAR virtual y datos tipo PointCloud 2 del RGB-D, que componen la PCL_{RGB-D}. Como salida, se genera un mensaje tipo PointCloud 2 que contiene la PCL reducida, V_{tk} y los ángulos de segmentación (α1, α2), que se envían a través de una línea de mensaje de ROS.

3.2.1.1 Interpolación y extrapolación

En situaciones reales, los datos dentro del vector "r" a menudo contienen valores que no están definidos correctamente y necesitan ser corregidos. Este proceso involucra identificar valores que no concuerdan adecuadamente, como valores infinitos, que se encuentran entre dos valores que tienen sentido. Estos valores incoherentes surgen porque el LIDAR virtual se deriva del Kinect, que a su vez está sujeto a condiciones variables. Estas condiciones pueden resultar en discrepancias debidas a materiales específicos o circunstancias del entorno, llevando a la pérdida de datos (como valores infinitos). Una vez que se detecta un valor inconsistente, se retiene el último dato válido y se agrupan todos los ángulos sucesivos con valores de profundidad incoherente en un vector de sustitución. Cuando se obtiene un nuevo dato válido en el vector "r", se utilizan los dos valores extremos del vector de sustitución para interpolar el segmento con valores irregulares. Durante esta interpolación, todos los valores inconsistentes se reemplazan con valores correspondientes determinados por la tendencia observada entre los dos extremos.

En caso de que el vector r comience con valores de profundidad infinito, se registran los ángulos para los cuales se detectó el infinito hasta encontrar el primer valor real. Una vez encontrado, se continúa con la búsqueda del segundo valor real consecutivo, con los cuales se realiza la extrapolación. De manera similar, cuando los valores infinitos se encuentran en el extremo final del vector, se registra el último dato real encontrado y se comienza la búsqueda hacia atrás desde la posición anterior del valor real ya registrado. Esta búsqueda continúa hasta encontrar otro valor real con el cual se realiza la extrapolación a partir de

la línea de tendencia generada por los valores reales. Este proceso permite generar un vector r sin valores inconsistentes.

3.2.1.2 Promedio móvil y desviación

La aplicación del promedio móvil se llevó a cabo utilizando las herramientas estadísticas de la librería Numpy de Python. Para ello, se utilizó una matriz donde se guardan los valores del vector r correspondientes al ciclo actual y a los $n-1$ ciclos anteriores. En cada ciclo, y para cada conjunto de n datos, se crearon los vectores $\overline{\boldsymbol{r_k}}$ y $\boldsymbol{dr_k}$, que junto con el vector $\boldsymbol{\theta_k}$, los cuales forman el conjunto de datos del LIDAR virtual.

3.2.1.3 Ángulo de detección

La información proporcionada por los vectores $\overline{\boldsymbol{r_k}}$, $\boldsymbol{\theta_k}$ y $\boldsymbol{dr_k}$ en cada ciclo es utilizada para llevar a cabo un análisis detallado con el objetivo de detectar los puntos de máxima variabilidad. Este análisis implica identificar los valores de profundidad de $\overline{\boldsymbol{r_k}}$ que presenten los valores más altos de dr, lo que permite determinar los ángulos que definen las fronteras de la posición del objeto dinámico en el ciclo k-ésimo.

Para lograr esto, se establece un umbral como la mitad del valor máximo de dr presente en el $\boldsymbol{dr_k}$ de cada ciclo. Cualquier valor de dr que supere este umbral se clasifica como un punto de alta variabilidad, y se seleccionan los dos valores máximos (dr_{max1}, dr_{max2}) que no estén en posiciones consecutivas dentro del vector.

Luego, los datos de dr_k se dividen en zonas delimitadas por los puntos de alta variabilidad, lo que permite determinar el número máximo de puntos consecutivos de alta variabilidad, que corresponden a cambios bruscos en la escena debido a la presencia de objetos dinámicos. Después de delimitar estas zonas, se clasifican como estáticas o dinámicas utilizando los valores de dr_{max1} y dr_{max2}.

En caso de haber más de una zona clasificada como dinámica, se sigue un criterio para seleccionar la más adecuada, dando preferencia a la zona dinámica más cercana a la zona anteriormente seleccionada como dinámica. Si no hay zonas dinámicas registradas anteriormente, se selecciona la zona con el máximo número de puntos de alta variabilidad

consecutivos.

Este proceso permite obtener dos posiciones fronterizas ($i1$ e $i2$) que delimitan la zona dinámica y están asociadas a un ángulo θ correspondiente. A partir de estos ángulos y utilizando la ecuación (5), se pueden obtener los ángulos α1 y α2 del vector θ_k.

3.2.1.4 Reducción de PCL RGB-D

La estructura de la PCL_{RGB-D} consiste en datos que incluyen información sobre el color (R, G y B) y la posición (x, y, z) de cada punto en un conjunto de datos. Debido a que la PCL_{RGB-D} se compone de una gran cantidad de datos, procesarla en tiempo real puede ser difícil debido al peso computacional. Para abordar este problema, se utilizó el método voxelgrid para reducir la densidad de puntos en la PCL_{RGB-D}. Este método establece una cuadrícula tridimensional con una longitud de celda fija y solo conserva los datos de posición, lo que resulta en la generación de la $PCL_{reducida}$ en cada ciclo. En el proceso de voxelgrid, se promedian los puntos dentro de cada celda de la cuadrícula para obtener un punto representativo, lo que conserva las características del entorno.

3.2.2 Nodo segmentación

El nodo de segmentación se conecta a una línea de mensajes que recibe información del nodo de detección, específicamente un mensaje PointCloud 2 que incluye la $PCL_{reducida}$, el VT_k y los ángulos de segmentación ($\alpha 1, \alpha 2$). La salida del nodo de segmentación son dos mensajes PointCloud 2 que contienen la PCL_{est-k} y PCL_{din}, los cuales son generados después de la fusión de datos y la segmentación de la $PCL_{reducida}$ en ambos conjuntos de datos. Este proceso se realiza mediante técnicas de procesamiento de nube de puntos para la identificación y segmentación de características relevantes en la $PCL_{reducida}$.

3.2.2.1 Fusión de datos

El proceso de fusión de datos implica la necesidad de alinear los centros geométricos del LIDAR virtual y la cámara RGB-D. En el caso del LIDAR virtual, su apertura angular se extiende a 57 grados en el plano XY (zona morada), mientras que la cámara RGB-D tiene

una apertura tridimensional de 40 grados (zona verde). Esta disparidad en la apertura entre el LIDAR y la cámara RGB-D se traduce en un ángulo de detección preliminar del LIDAR virtual, lo que establece los límites de la nube de puntos de la zona dinámica (PCL_{ZD}) antes de que el objeto dinámico ingrese a la PCL_{RGB-D}. Dichos límites están determinados por los ángulos de detección ($\alpha 1$ y $\alpha 2$), que definen la segmentación de la zona dinámica en la $PCL_{Reducida}$ (Figura 5).

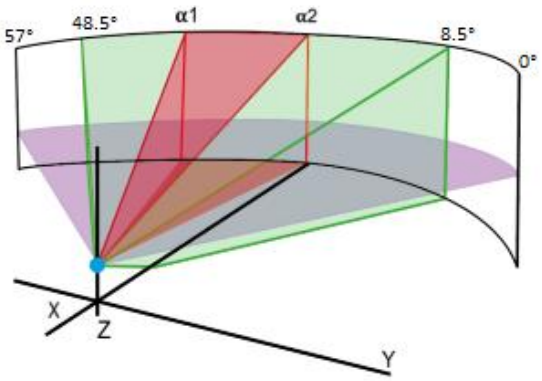


Figura 5. Ángulos de segmentación, apertura del RGB-D y plano de detección del LIDAR virtual

En el proceso de segmentación basado en la agrupación por profundidad, se lleva a cabo la segmentación de la nube de puntos de la zona dinámica (PCL_{ZD}), que abarca tanto los puntos del objeto dinámico como los alrededores de la zona estática ocluida. Durante este proceso de segmentación, se realiza un análisis de agrupación de puntos en función de su profundidad dentro de la PCL_{ZD}. Para lograr esto, los puntos se representan gráficamente en un vector a lo largo del eje x, dividiéndolo en cuadros de tamaño t_{vg}. A continuación, se procede a detectar las zonas y se selecciona aquella que se encuentra más cercana al origen, indicando la ubicación del objeto dinámico (Figura 6).

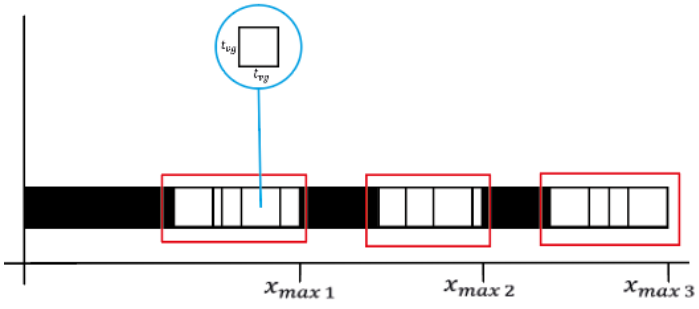


Figura 6. Segmentación por agrupación por profundidad.

A partir de este análisis, se determina el parámetro máximo de profundidad (X_{max}) para los puntos en relación con el eje x. Todos los puntos con una coordenada x menor a este valor se consideran parte de la nube de puntos del objeto dinámico (PCL_{Din}), mientras que aquellos con una coordenada x mayor se consideran parte de la nube de puntos de la zona estática que rodea a la zona ocluida (PCL_{Oek}). Al combinar la PCL_{zek} y la PCL_{Oek}, se obtiene una concatenación que resulta en la PCL_{est-k}, que representa la escena estática con los puntos correspondientes al objeto dinámico extraído.

Una vez obtenida la PCL_{est-k}, se lleva a cabo el proceso de traslación, donde cada punto de referencia visual (v_{rp}) en la PCL_{est-k} se multiplica por la Mt_k. Como resultado de este proceso, se obtienen los puntos de referencia trasladados (v_{rtp}), que conforman la PCL_k.

3.2.3 Nodo Reconstrucción

El nodo de reconstrucción se encuentra enlazado con la línea de mensajes proveniente del nodo de segmentación, el cual proporciona la nube de puntos segmentada (PCL_k). Como resultado, el nodo de reconstrucción genera una línea de mensajes de datos del tipo PointCloud 2 que contiene la reconstrucción de la escena (PCL_{KT}). Este proceso implica la concatenación sucesiva de las nubes de puntos para obtener la escena reconstruida.

Al utilizar la función de concatenación de la librería Numpy en Python entre la nube de puntos PCL_K y la nube de puntos reconstruida anterior PCL_{KT-1}, se obtiene la nube de puntosPCL_{Trans}. En esta nube, se observan áreas con una mayor densidad de puntos debido a que PCL_K y PCL_{KT-1} comparten algunas zonas. Esto conlleva a un aumento en el tamaño de la nubePCL_{Trans}.

Para reducir la densidad de puntos, se aplica el método del voxelgrid con un tamaño de celda tVG. Este proceso produce la nube de puntos PCL_K, que corresponde a la reconstrucción de la escena estática después de la reducción de densidad.

3.2.4 Correlación

La correlación se realiza fuera de línea y no forma parte del sistema Rec-HV. Se utiliza para evaluar la correlación lineal entre dos imágenes, en este caso las nubes de puntos 3D

PCL_{KT} y PCL_{Ref}, que deben estar en el mismo marco de referencia. El proceso de correlación se utiliza para determinar la correspondencia entre puntos en ambas imágenes y puede ser útil para validar la precisión y la alineación de la reconstrucción de la escena estática.

El proceso de transformación consiste en aplicar las ecuaciones (10) para obtener las representaciones en dos dimensiones de las nubes de puntos tridimensionales (PCL_{KT} y PCL_{Ref}), como se muestra en la Figura 7. Estas ecuaciones permiten proyectar los puntos de las nubes de puntos en un plano bidimensional, lo que simplifica la comparación y evaluación de la correlación entre las imágenes.

Las representaciones en dos dimensiones, Pt_{kt} y Pt_{ref}, se obtienen mediante la proyección de los puntos de las nubes de puntos PCL_{KT} y PCL_{Ref} en un plano de referencia común. Esta transformación facilita el cálculo de la correlación entre las imágenes, ya que se reduce la complejidad de trabajar con nubes de puntos tridimensionales completas.

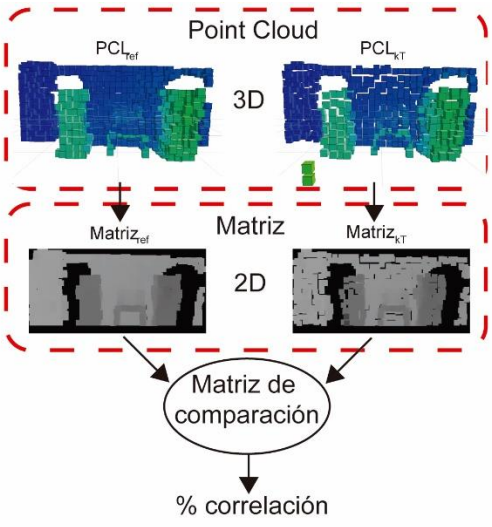


Figura 7. Correlación

Una vez que se obtienen las representaciones en dos dimensiones (PCL_{KT} y PCL_{Ref}) de las nubes de puntos tridimensionales, se utiliza la función matchTemplate de la biblioteca OpenCV en Python. Esta función realiza la convolución de las dos imágenes para encontrar la mejor coincidencia entre ellas.

La convolución es un proceso mediante el cual se desliza una imagen sobre otra y se calcula la similitud en cada posición. En el caso de la función matchTemplate, se busca

maximizar la coincidencia entre las dos imágenes mediante la evaluación de la similitud de patrones.

El resultado de la función matchTemplate es una imagen que muestra la respuesta de la convolución en cada posición. El máximo valor de esta imagen indica la posición donde se encontró la mejor coincidencia entre las dos imágenes. A partir de este valor máximo, se puede determinar la correlación entre las imágenes y realizar las correspondientes acciones en el sistema Rec-HV.

En resumen, la función matchTemplate de OpenCV se utiliza para encontrar la mejor coincidencia entre las imágenes 2D obtenidas a partir de las nubes de puntos tridimensionales, permitiendo evaluar la correlación y realizar las acciones necesarias en el sistema.

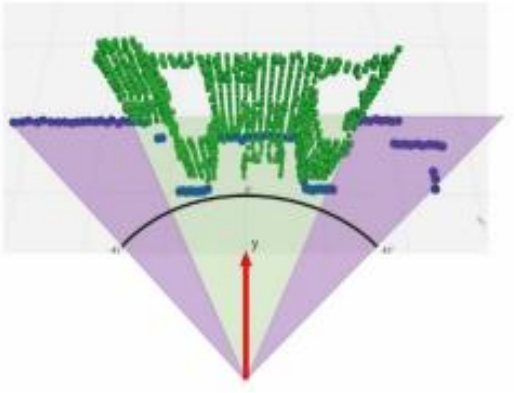
3.3 Materiales

3.3.1 Sensores

Para llevar a cabo esta investigación se empleó el sistema operativo robótico ROS V1.0 en Ubuntu 18.06 y una computadora ASUS VIVOBOOK X512F equipada con un procesador Core i5 de 3.9 GHz, que cuenta con cuatro núcleos reales y cuatro virtuales, una memoria RAM de 12 GB y una GPU MX230 Nvidia. Para la adquisición de datos se utilizó un sensor Kinect V1 tipo RGB-D, con un campo de visión horizontal de 57°, vertical de 43°, una imagen de profundidad de 320 x 240 y una imagen RGB de 640 x 480, con una tasa de muestreo de 30 fps, el cual se conectó a través de un adaptador USB tipo A, y un LIDAR virtual basado en el mismo Kinect V1.

Los sensores ahora tienen centros geométricos coincidentes gracias a la simulación del LIDAR, lo que les permite una visión coherente en el eje Y del RGB-D.

A



B

Figura 8. Instrumento de medición (A) Cámara RGB-D, este a su vez emula el LIDAR virtual. (B) Campo visual

Los instrumentos tienen diferentes amplitudes de campo visual: el RGB-D tiene 40° (zona verde en la Figura 8[B]) mientras que el LIDAR virtual cuenta con 57° (zona morada en la Figura 8 [B]). Como resultado, se puede detectar un ángulo de 8,5° hacia los lados en el plano X (Figura 8 [B]).

3.3.2 Calibración del RGB-D

Antes de capturar los datos, se calibró el sensor RGB-D. Debido a las características de fabricación y los materiales del dispositivo, se requiere la aplicación de ciertos parámetros de calibración para evitar problemas de aberración e inconsistencias entre la imagen de la cámara RGB y la de profundidad. Se utilizó una herramienta de ROS para este proceso, se tomó imágenes de una cuadrícula de guía ubicada a diferentes profundidades (Figura 9[A]). La herramienta identificó los puntos en la cuadrícula y los enmarca (Figura 9 [B]).

A

B

Figura 9. Imágenes usadas en el proceso de calibración. (A) imágenes de referencia. (B) imágenes donde se localizó la cuadricula para calibración

Este proceso produce un error de re proyección específico para cada instrumento, medido en píxeles dado por (11).

$$e_{rep} = 0.114397 \tag{11}$$

Además, el software genera una serie de matrices que se guardan en un archivo que se lee al inicio de la ejecución del software RGB-D. Esto ayuda a reducir el error del instrumento.

3.4 Base de datos para pruebas

La creación de una base de datos sólida y auténtica ha sido un paso esencial en el desarrollo de este proyecto. En este empeño, nos hemos sumergido en la cotidianidad, capturando escenarios internos que se encuentran impregnados de autenticidad. Estos escenarios han sido meticulosamente seleccionados para representar una variedad de condiciones, abarcando desde iluminación artificial hasta iluminación natural. Cada escenario incluye un único objeto dinámico, lo que añade una dimensión adicional de realidad a la base de datos, y una variedad de obstáculos que reflejan la complejidad de

los entornos reales (ver Tabla 2).

Los recorridos trazados a través de estos escenarios han sido cuidadosamente planificados para capturar la diversidad y la riqueza de situaciones del mundo real. Desde trayectorias lineales hasta movimientos diagonales, y variaciones en la profundidad que se despliegan de derecha a izquierda y viceversa, se ha buscado reflejar la complejidad y la variedad de movimientos que se pueden encontrar en la vida cotidiana (ver Figura 11).

La uniformidad ha sido clave en la estandarización del ritmo de marcha del objeto dinámico. Guiados por un metrónomo, estos objetos se mueven a un ritmo constante, lo que asegura la consistencia en el proceso y permite una evaluación precisa.

La variabilidad de la base de datos se ha enriquecido mediante la captura en diferentes momentos del día, desde las primeras luces de la mañana hasta la oscuridad de la noche. Esto no solo proporciona una amplia gama de condiciones lumínicas, sino que también desafía al sistema en diferentes contextos.

Adicionalmente, algunas escenas han sido configuradas con elementos que podrían influir negativamente en el proceso. Esta incorporación tiene un propósito específico: poner a prueba la capacidad del sistema para responder de manera efectiva ante condiciones adversas y demostrar su solidez.

3.4.1 Objetos dinámicos

Se utilizó un único objeto dinámico, personas que realizaron recorridos a diferentes velocidades (ver Figura 10).

A

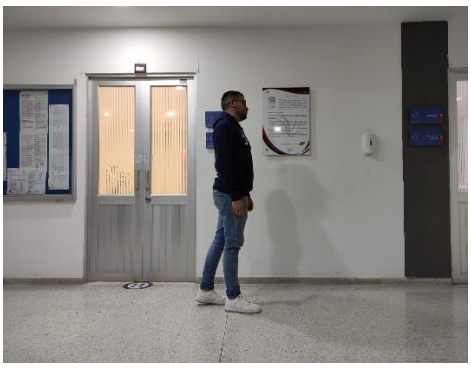
B

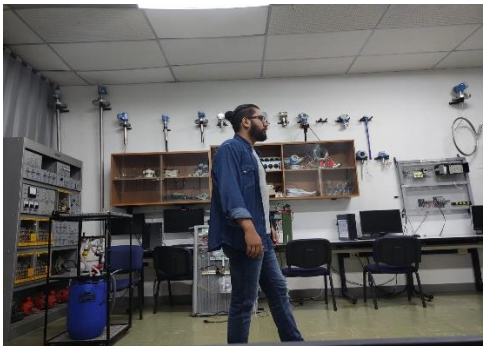
C

Figura 10. Objetos dinámicos A, B Y C

3.4.2. Escenas

Se llevaron a cabo pruebas con el sistema Rec-HV en diferentes escenarios con variadas características y niveles de complejidad. La siguiente tabla muestra los escenarios y sus características y la Figura 11 presenta los tipos de recorridos realizados en los nueve escenarios utilizados.

Escenas		Características
A		La escena muestra un fondo plano con pocos elementos, y un reflejo en acrílico que afecta la visualización del Kinect, cuenta con una profundidad de 3.2m
B		La escena consta de dos paredes paralelas con un hueco en el medio. Se limita la presencia de elementos que puedan interferir con la detección, pero la profundidad del hueco puede generar variaciones en la precisión de la detección, cuenta con una profundidad de 6.1m
C		La escena muestra una pared con diversas afectaciones que dificultan la visualización, como puertas con vidrios, cambios de iluminación y otros elementos. A pesar de estas interferencias, la profundidad de la pared se mantiene constante en 3.5 metros.

D		La escena presenta una pared plana, pero se ve considerablemente afectada en la detección debido a la presencia de un gran tablero que ocupa un espacio significativo en ella. A pesar de la interferencia, la profundidad de la pared se mantiene constante en 3.8 metros.
E		La escena está completamente llena de disturbios, como pantallas negras, numerosos elementos dispersos, ventanas, cambios de iluminación, entre otros. A pesar de esta situación caótica, la profundidad de la escena se mantiene en 4.8 metros. Sin embargo, la distribución de la profundidad puede ser irregular debido a la presencia de los elementos que se encuentran en la escena.
F		La escena presenta un grado de dificultad medio, con una pared de fondo y paredes a los lados. Además, hay una puerta en la misma escena. Se encuentran varios objetos que generan interferencia, y la iluminación es artificial. A pesar de estos desafíos, la escena se puede manejar con cierta complejidad, requiriendo medidas adicionales para optimizar la detección y la precisión del entorno, la escena cuenta con una profundidad de 4m

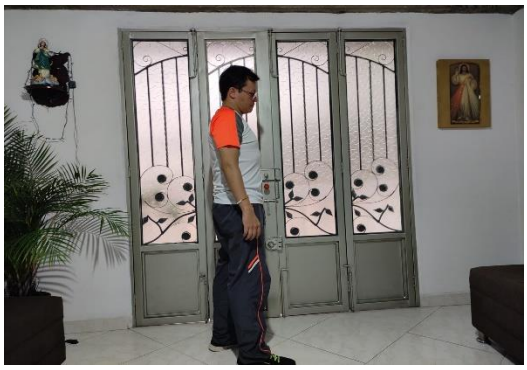
G		La escena se caracteriza por tener superficies de baja reflectividad, lo que significa que los objetos presentes no generan reflejos intensos. Además, la profundidad promedio de la escena es de 4 metros. Estas características facilitan la detección y seguimiento de objetos en el entorno, ya que los niveles de interferencia y distorsión son mínimos, lo que contribuye a una visualización clara y precisa.
H		La escena se ve afectada por disturbios, como pantallas apagadas, elementos inesperados y cambios abruptos de iluminación. A pesar de esta situación caótica, la profundidad de la escena se extiende hasta los 4.6 metros. Sin embargo, la distribución de esta profundidad puede ser irregular debido a la presencia de los elementos que se encuentran dispersos en la escena.
I		En un espacio reducido, iluminado por la tenue luz natural y lleno de diversos objetos de fondo, se despliega una escena intrigante y desafiante. Con una profundidad de solo 4.5 metros, esta escena se ve afectada por múltiples disturbios que amenazan con interrumpir un correcto funcionamiento.

Tabla 2. Características de los escenarios de prueba

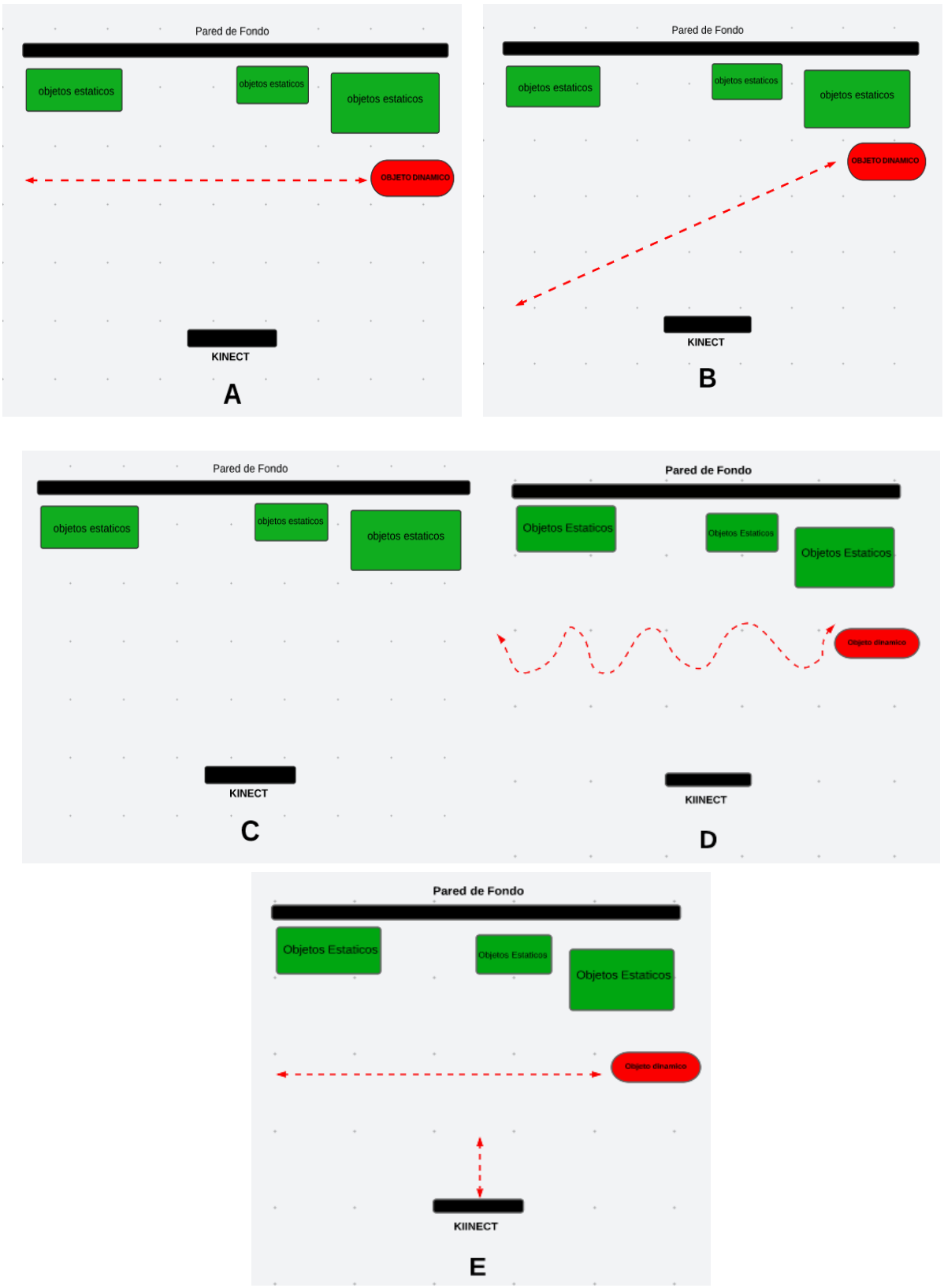


Figura 11. Recorridos seleccionados para los experimentos.

Capítulo 4.
Experimentos y Resultados

En este capítulo, se lleva a cabo la exposición y análisis de un conjunto de ocho experimentos diseñados para evaluar en profundidad el funcionamiento y desempeño del sistema denominado Rec-HV. Estos experimentos han sido cuidadosamente concebidos para abordar diversas situaciones y niveles de complejidad, con el propósito de obtener una comprensión exhaustiva de las capacidades y limitaciones del sistema en diferentes contextos. Para asegurar la rigurosidad de la evaluación, los experimentos se han ejecutado en ciclos, lo que significa que se han repetido en varias ocasiones para obtener una visión más completa y confiable de los resultados.

Cada ciclo de experimentación implica la observación y análisis de múltiples instancias del sistema Rec-HV en entornos estáticos que exhiben una amplia variedad de características. Además, en cada uno de estos ambientes, se introduce un elemento crucial: un objeto dinámico. Este objeto se mueve a lo largo de rutas predefinidas en las diferentes escenas. Esta adición de objetos dinámicos permite simular situaciones más realistas, ya que, en el mundo real, no todo permanece estático y los sistemas deben lidiar con elementos en movimiento.

En el proceso de evaluación, se han empleado métricas específicas diseñadas para medir y cuantificar diferentes aspectos del rendimiento del sistema. Estas métricas están diseñadas para evaluar tanto aspectos cuantitativos como cualitativos de los resultados obtenidos en cada experimento. En conjunto, esta serie de experimentos y evaluaciones tiene como objetivo brindar una visión completa y detallada de cómo el sistema Rec-HV se comporta en una variedad de situaciones, permitiendo identificar sus fortalezas y áreas en las que podría requerir mejoras.

En la Tabla 3 se presentan los objetivos y métricas de evaluación de cada uno de los 8 experimentos propuestos para evaluar el algoritmo.

Experimento	Objetivo	Métricas
1	Determinar el funcionamiento del LIDAR virtual cuando se varía el rango del mismo.	Variación de altura máxima y mínima (m) LIDAR virtual.
2	Determinar la relación entre la variación del tamaño del voxelgrid y tres variables: tiempo de ejecución, tiempo de segmentación y tamaño de la PCL. Se analizará cómo estas variables varían en función del tamaño del voxelgrid	Variación del tiempo de ejecución (s), del tiempo de segmentación (s) y del número de puntos en la PCL en relación al tamaño del voxelgrid (m).
3	Evaluar el ángulo de segmentación de la zona dinámica en escenarios con un objeto dinámico, mediante la detección en cada ciclo. Se buscará determinar la variación de este ángulo en cada ciclo.	Número de escenas en las cuales se detectó de manera precisa el objeto dinámico.
4	Evaluar la variación de las PCL estáticas y dinámicas obtenidas por la detección y eliminación del objeto dinámico al aplicar la segmentación en dos etapas	Variación en el número de puntos de la PCL tanto en la escena estática como en la zona dinámica.
5	Evaluar la variación de las PCL estáticas y dinámicas obtenidas por la detección y eliminación del objeto dinámico al aplicar la segmentación en dos etapas	Variación del porcentaje de correlación.
6	Evaluar el resultado de la detección, eliminación y reconstrucción de escenarios reales que contienen un objeto dinámico, mediante la aplicación	El tiempo promedio de ejecución (s), el tiempo promedio de segmentación (s), el número de puntos por ciclo, el porcentaje de reducción promedio de la PCL, el

	de Rec-HV	porcentaje de ciclos con puntos de alta variabilidad, el porcentaje de la PCL obtenida en la segmentación por ángulo, el porcentaje de la PCL obtenida en la segmentación por agrupación por profundidad, el número de puntos de la PCL inicial (PCL obtenida en el primer ciclo), el número de puntos de la PCL final (PCL final obtenida por la concatenación) y el porcentaje de incremento de puntos en la PCL de reconstrucción.
7	Determinar el error de posicionamiento del objeto dinámico en distintas trayectorias.	Error máximo de posicionamiento (m), el error absoluto de posicionamiento (m) y el error cuadrático medio (m).
8	Evaluar el sistema a 3 velocidades.	Detección a distintas velocidades.

Tabla 3. Métricas de los experimentos.

4.1 Experimento 1

En este experimento, se lleva a cabo el análisis del LIDAR virtual, teniendo en cuenta factores controlados y no controlados. Se realizan pruebas para validar su correcto funcionamiento y su similitud con un LIDAR convencional. El objetivo principal es evaluar la capacidad del LIDAR virtual para proporcionar mediciones precisas y confiables, comparándolas con las obtenidas por un LIDAR convencional.

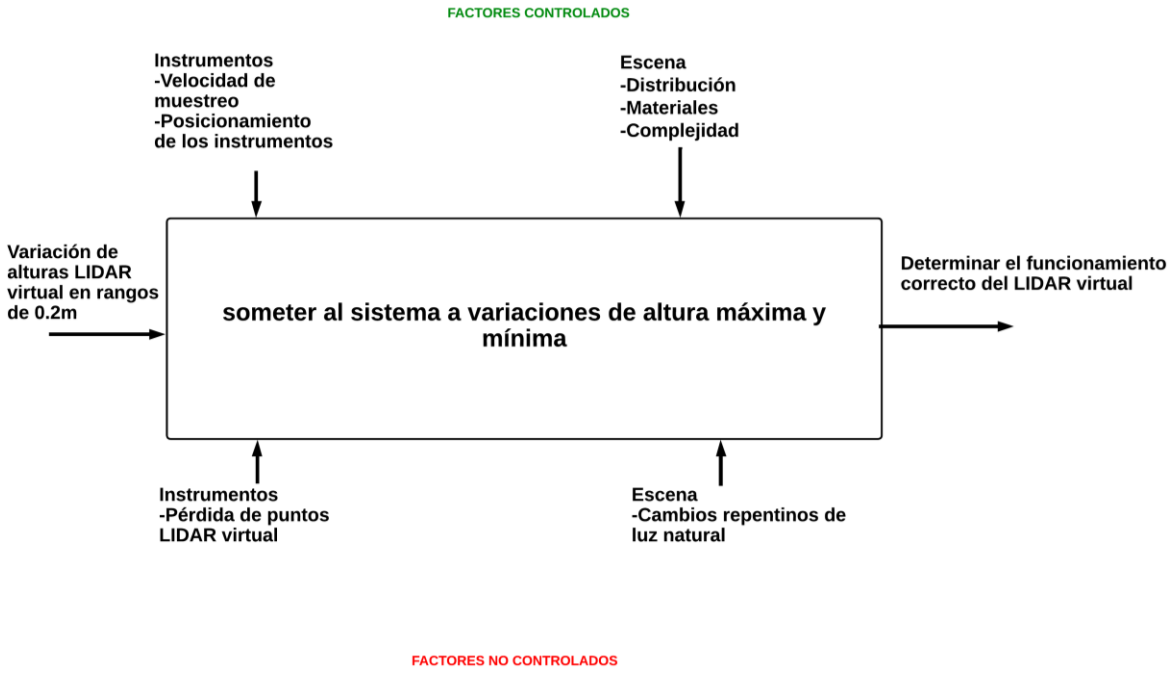
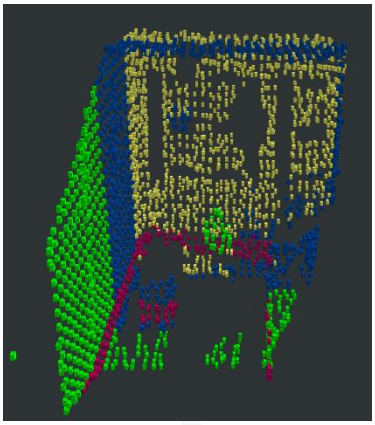


Figura 12. Factores controlados y no controlados experimento 1

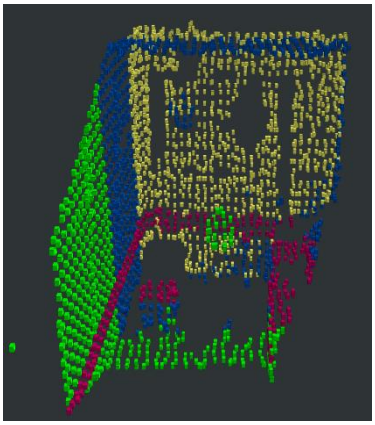
El experimento para comprobar el funcionamiento del LIDAR virtual, se lleva a cabo en el escenario de la Figura 13. Este experimento busca obtener pruebas concretas sobre el desempeño del LIDAR virtual en situaciones donde haya variaciones en su rango de operación. La intención es demostrar su capacidad para detectar y reconocer elementos en dicho escenario, dejando evidencia de su rendimiento en condiciones desafiantes.

Para estas pruebas, se varia la altura máxima y mínima del LIDAR virtual en un rango de -1m a 1m, expresadas en metros, en incrementos de 0.2 metros (franja violeta) (Figura 14). Donde se determinará la variación del rango del LIDAR virtual y visualizará los cambios resultantes. Así, se podrá observar cómo la visión se ve afectada en función del rango de detección seleccionado.

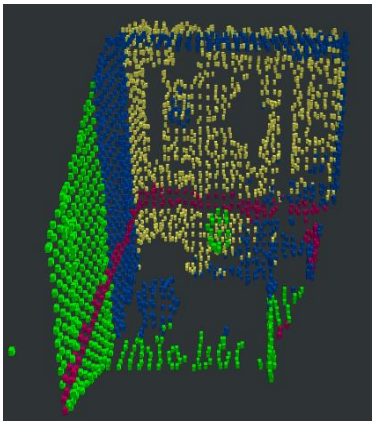
Figura 13. Escena para LIDAR virtual



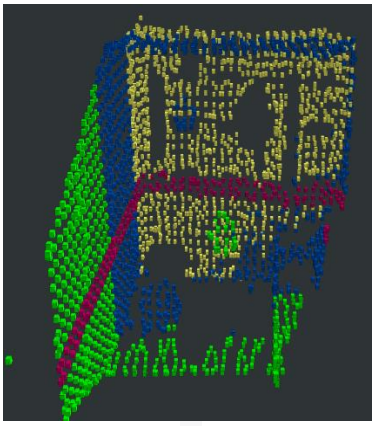
A



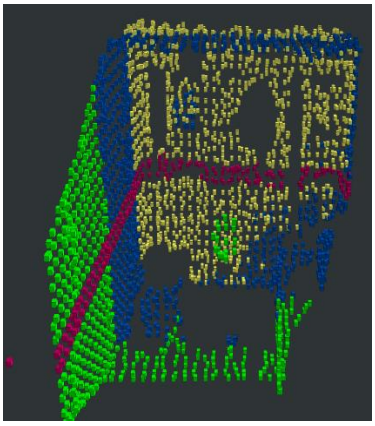
B



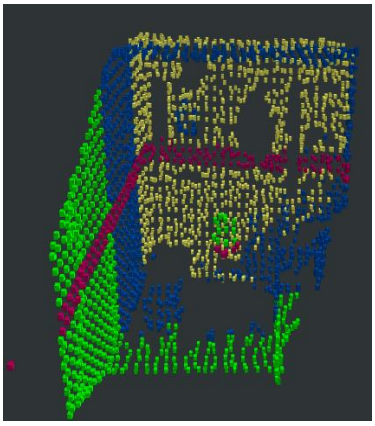
C



D



E



F

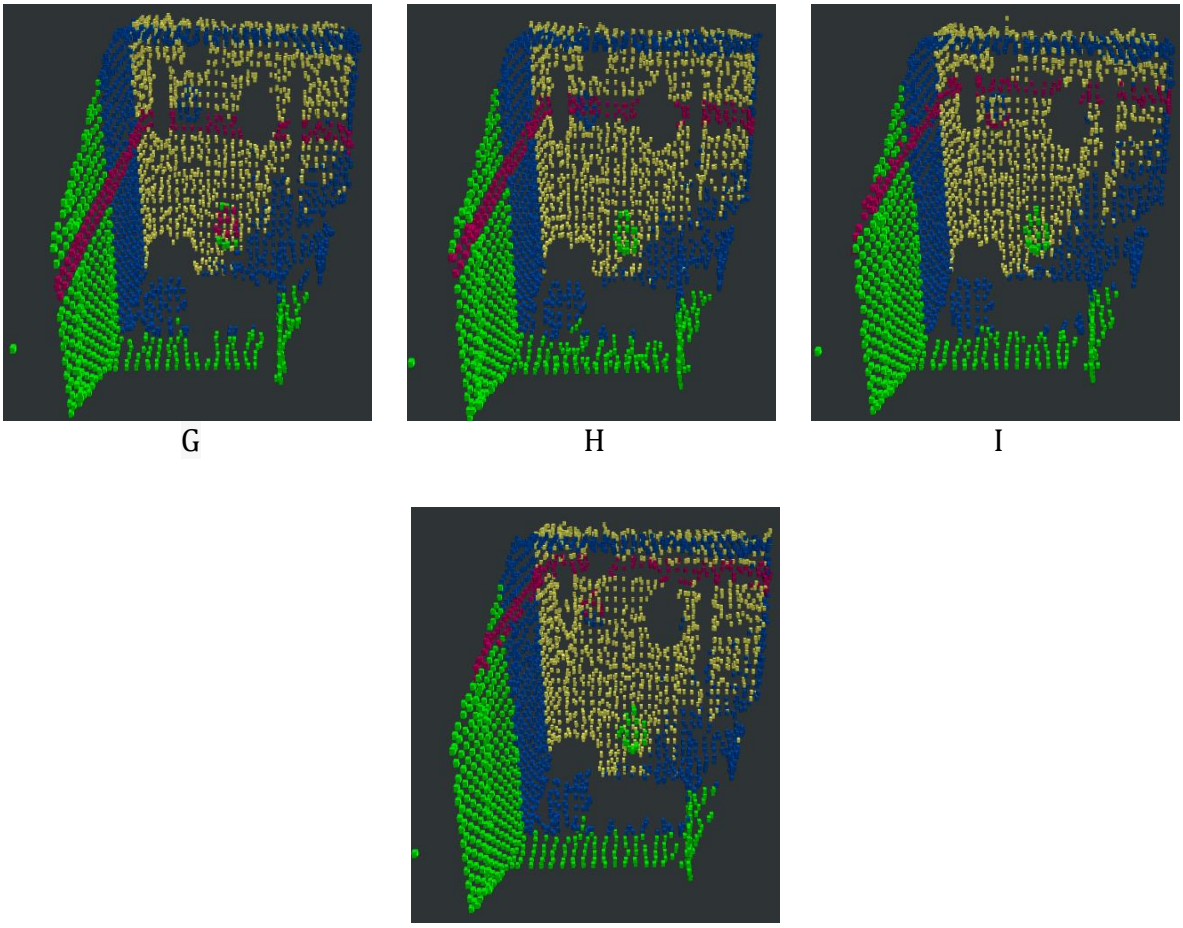
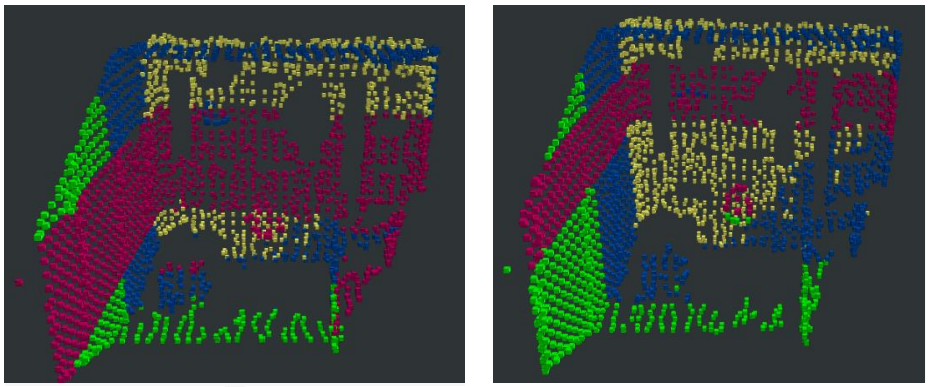


Figura 14. Variación de la altura máxima y mínima en LIDAR Virtual con variación de 0.2m en el rango de 1m a -1m.

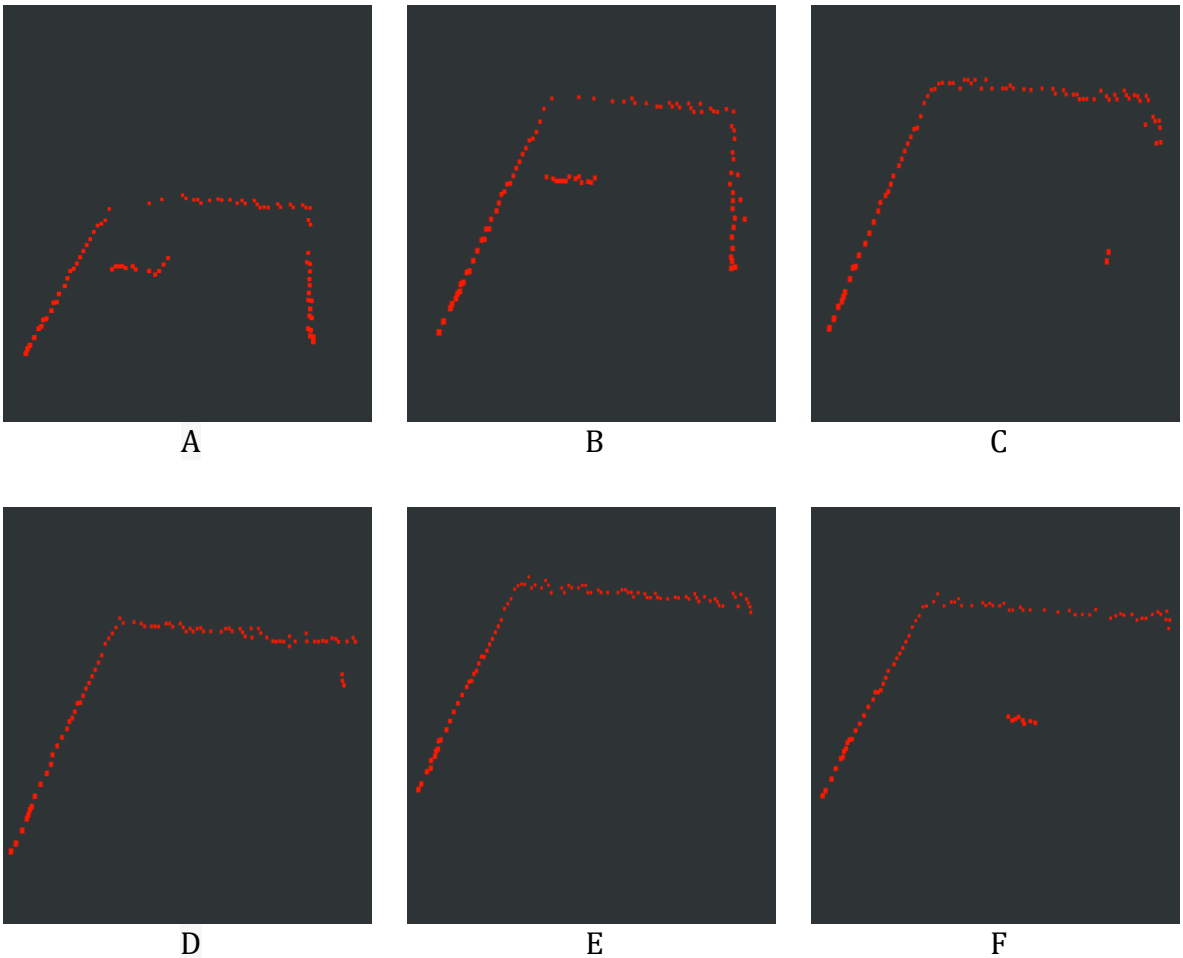
Como resultado del experimento, en la Figura 15 se destaca la zona donde el LIDAR virtual detectó los objetos presentes en la escena, como los globos y el sillón (franja violeta). Esta referencia visual es importante para verificar el correcto funcionamiento del LIDAR virtual, ya que es capaz de identificar distintas distancias entre los objetos detectados.



A B

Figura 15. (A) zona detección globo-sillón (-0.7m a 0.5m), (B) zona detección globo-globo (0.2m a 0.8m)

A continuación, se muestra exclusivamente la franja del LIDAR virtual (LIDAR virtual generado) para comprender mejor su funcionamiento y su capacidad de medir distancias en el entorno (Figura 16).



A B C

D E F

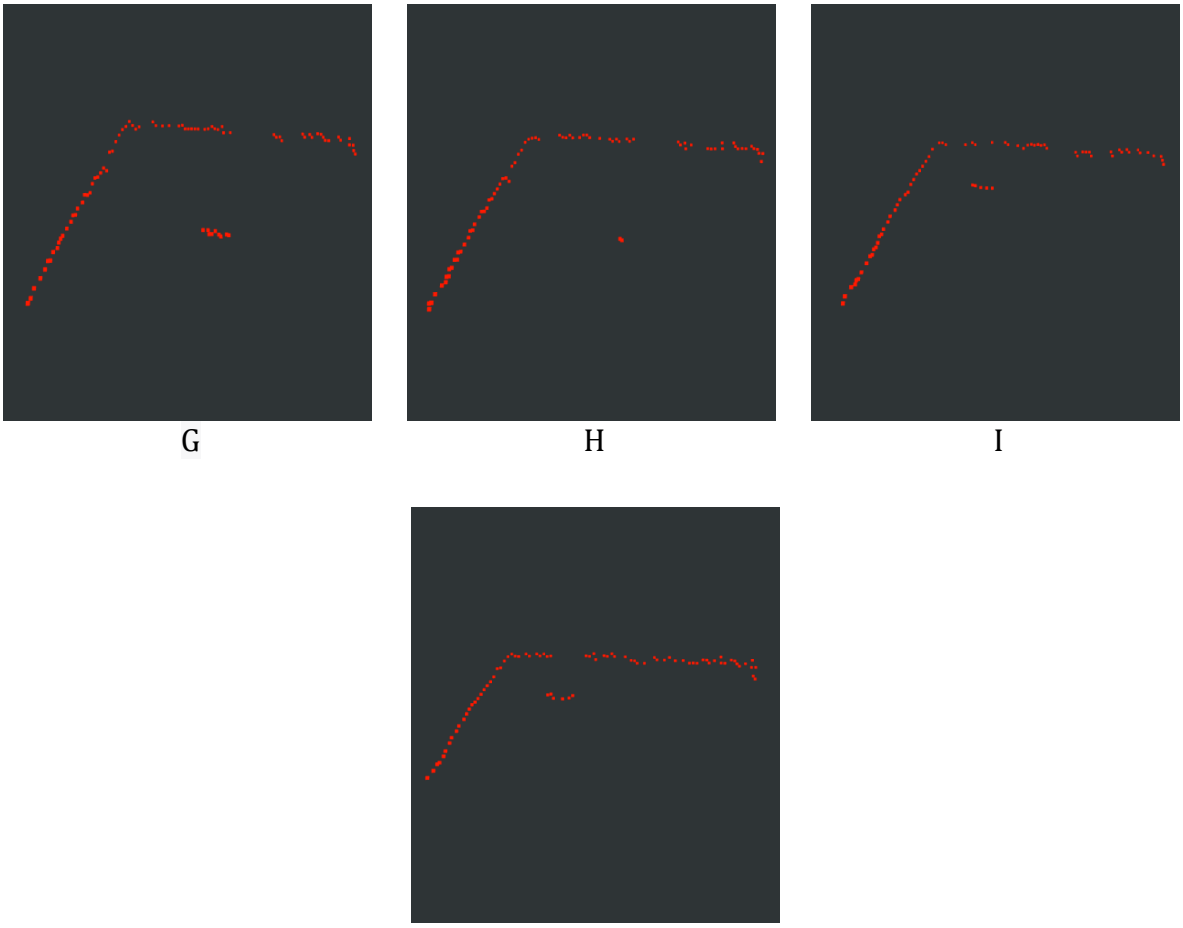
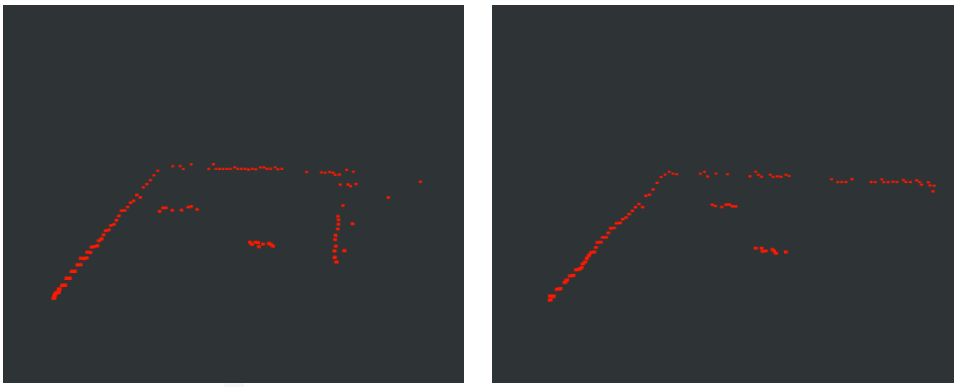


Figura 16. LIDAR generado por cada variación

En la Figura 17, se muestra el rango en el que el globo y el sillón fueron detectados utilizando exclusivamente el LIDAR virtual generado. En esta representación, se presenta únicamente la información capturada por el LIDAR virtual, lo que permite visualizar con mayor claridad la detección de los objetos y las distancias asociadas en la escena.



A B

Figura 17. (A) zona detección LIDAR generado globo-sillón (-0.7m a 0.5m), (B) zona detección LIDAR generado globo-globo (0.2m a 0.8m)

Los resultados mostraron que el LIDAR virtual demostró una efectiva capacidad de detección de objetos en la escena, como globos y un sillón, y pudo identificar diferentes distancias entre los objetos detectados. La representación visual del LIDAR virtual proporcionó una clara visualización de la detección de objetos y las distancias correspondientes en el entorno evaluado.

En conclusión, los resultados respaldan el correcto funcionamiento y desempeño del LIDAR virtual en términos de resolución de distancia. El sistema demostró una alta precisión en la percepción del entorno, lo que lo hace idóneo para aplicaciones que requieren una percepción espacial precisa, como la navegación autónoma, la cartografía 3D y la detección de obstáculos en tiempo real.

4.2 Experimento 2

En este experimento, los datos del RGB-D se emplearán como entrada, mientras que los tiempos de ejecución y segmentación del sistema serán medidos como salida. Se identifican una serie de factores, algunos controlados y otros no controlados, que podrían influir en los resultados del experimento.

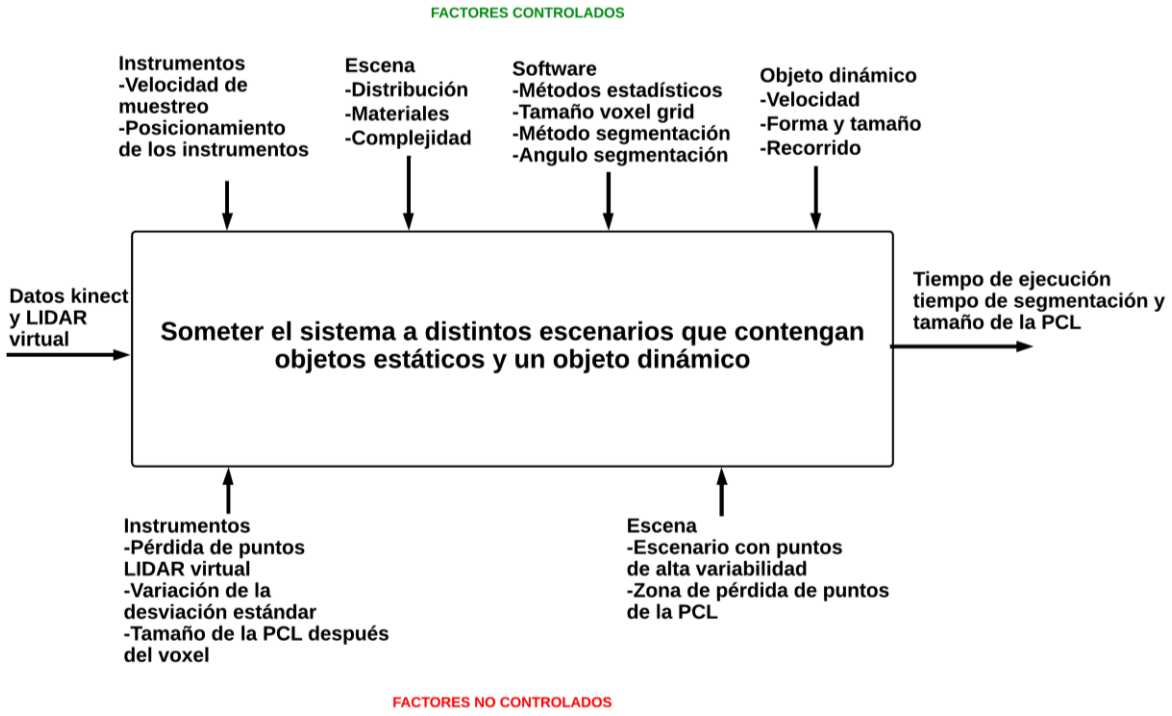


Figura 18. Factores controlados y no controlados experimento 2

En este experimento, se lleva a cabo pruebas con variaciones entre 340 y 530 ciclos. La primera prueba consiste en evitar la aplicación del voxelgrid para obtener una referencia en tiempo de ejecución. Posteriormente, se realizan 12 pruebas adicionales con diferentes valores para el voxelgrid, en incrementos de 0.01.

Durante todas las pruebas, no se introducirán objetos dinámicos y los ángulos de segmentación se mantendrán constantes para garantizar condiciones uniformes en todos los ciclos.

Se implementó un sistema en línea que redujo la densidad de puntos mediante el uso del voxelgrid. Esto permitió ajustar los tiempos de ejecución del sistema Rec-HV. Se mantuvo el sensor en la misma posición, con vista hacia la escena estática A (Tabla 2), y se utilizó un rango de ángulo de segmentación de 19° a 38°, sin presencia de objetos dinámicos.

Se realizó una prueba exhaustiva de referencia para determinar la capacidad de reducción del tiempo de segmentación. Se procesó la nube de puntos (PCL) sin aplicar la técnica de voxelgrid (tamaño promedio de 199000), y el tiempo promedio de ejecución fue de 3.75 segundos, utilizado como valor de referencia para el análisis comparativo.

Luego, se llevaron a cabo pruebas adicionales utilizando 12 valores diferentes de

voxelgrid, que abarcaban un rango desde 0.05m hasta 0.16m de arista. Se obtuvieron múltiples mediciones para cada valor de voxelgrid, registrando el promedio del tiempo de ejecución, el promedio del tiempo de segmentación y el promedio de la PCL generada. Los valores promedio mostraron una variación significativa, oscilando entre 340 y 530 ciclos al aplicar cada valor de voxelgrid, como se ilustra en la Figura 19.

Estos resultados respaldan la eficacia del enfoque propuesto y proporcionan una perspectiva cuantitativa del impacto de la técnica de voxelgrid en la reducción del tiempo de segmentación de la PCL. Esta información resulta relevante para optimizar y mejorar el rendimiento de futuros procesos de segmentación en aplicaciones tecnológicas.

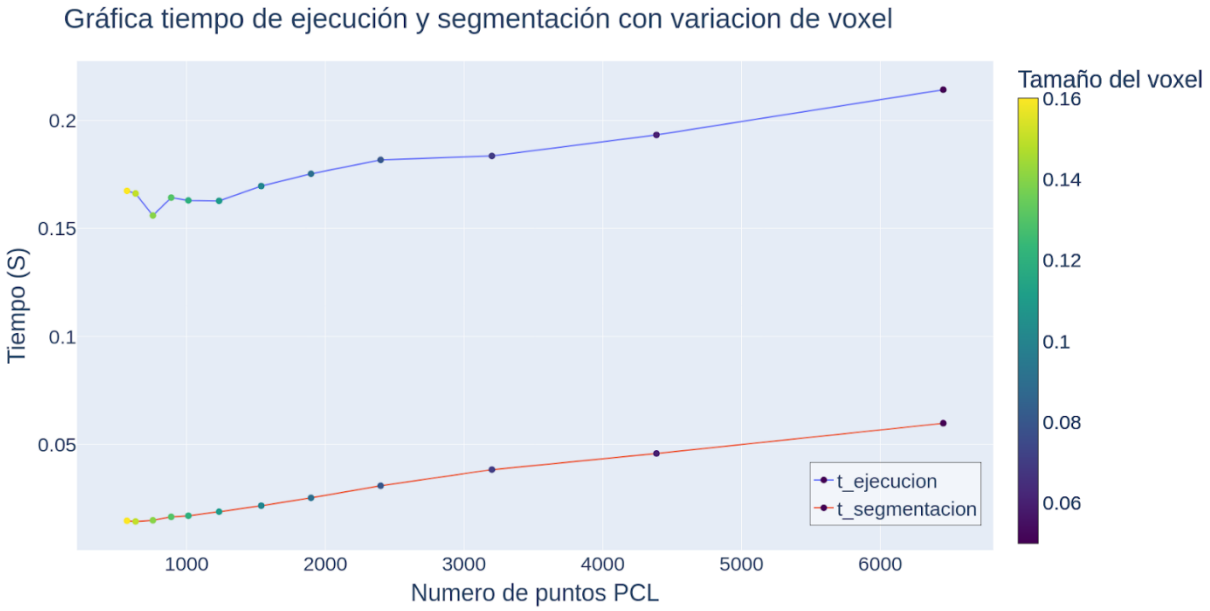


Figura 19. Tiempo promedio de ejecución, tiempo promedio de segmentación, tamaño promedio de la PCL para cada valor de voxelgrid.

En conclusión, este estudio demostró con éxito la efectividad del sistema en tiempo real. La aplicación de la técnica de voxelgrid fue clave para optimizar el rendimiento, al reducir la densidad de puntos y mejorar los tiempos de ejecución del sistema Rec-HV.

Los hallazgos cuantitativos obtenidos en este estudio son esenciales para mejorar y optimizar futuros procesos de segmentación en aplicaciones tecnológicas que requieran una percepción espacial precisa y eficiente. La estrategia efectiva del voxelgrid se ha demostrado como una herramienta valiosa para reducir la complejidad de la nube de puntos y mejorar la eficiencia del sistema.

4.3 Experimento 3

El vector r de cada ciclo se usará como entrada y el resultado deseado será el ángulo de detección. Es importante considerar los diferentes factores externos que pueden influir en el experimento.

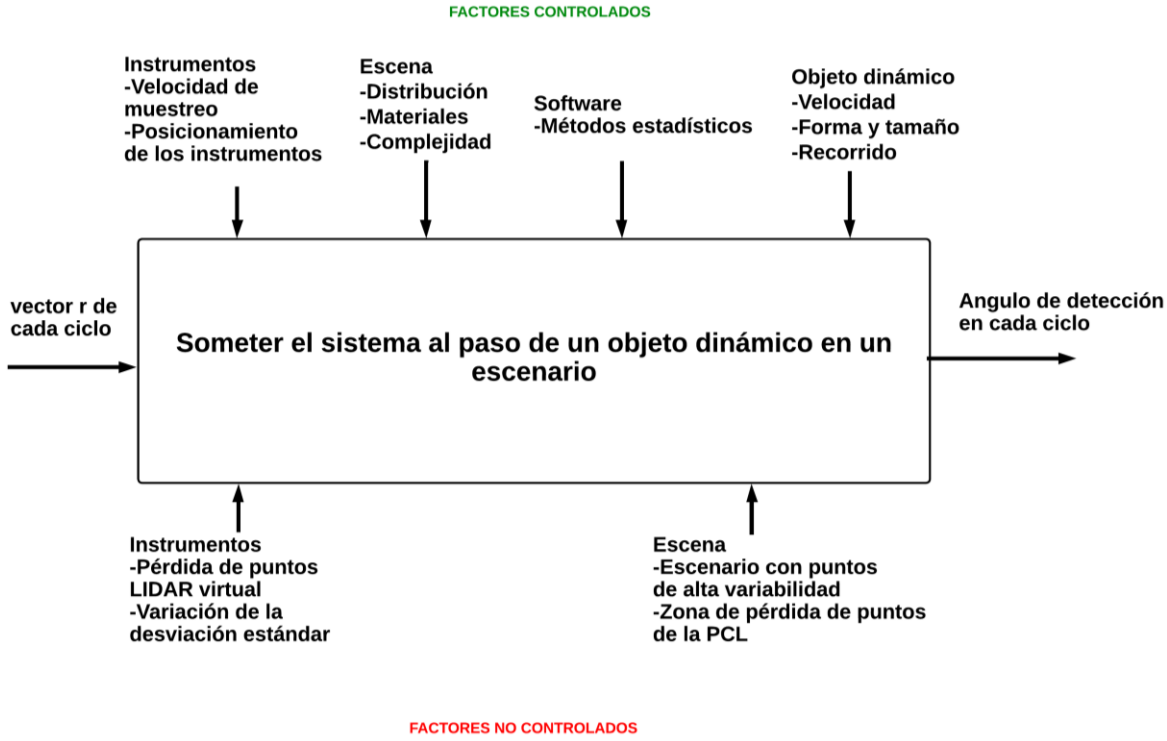


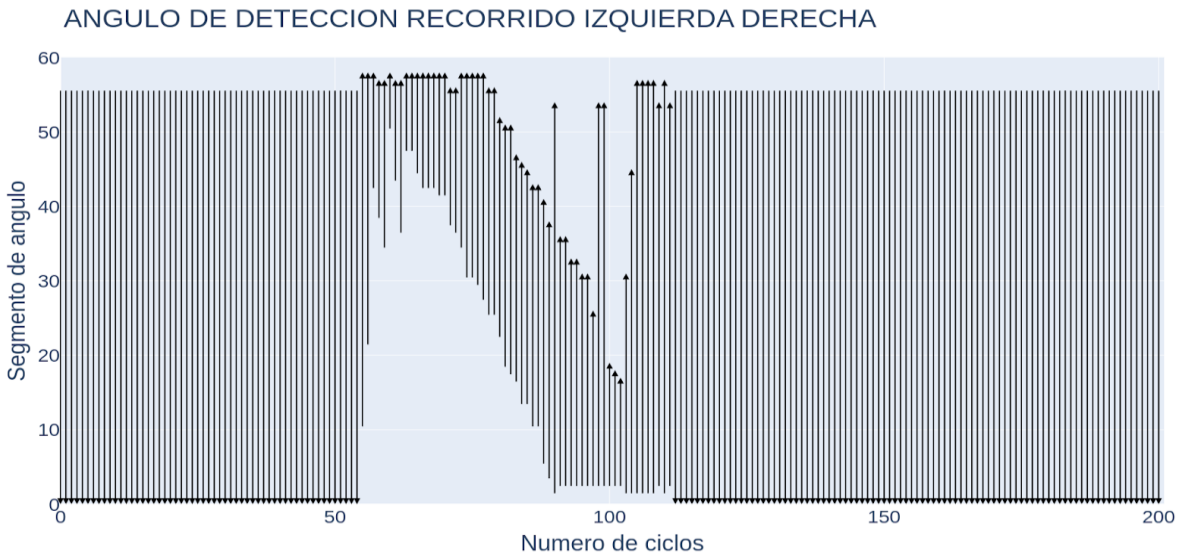
Figura 20. Factores controlados y no controlados experimento 3

En este experimento se tendrá presencia del objeto dinámico cruzando la escena con un recorrido establecido, ver (Figura 11 [A]).

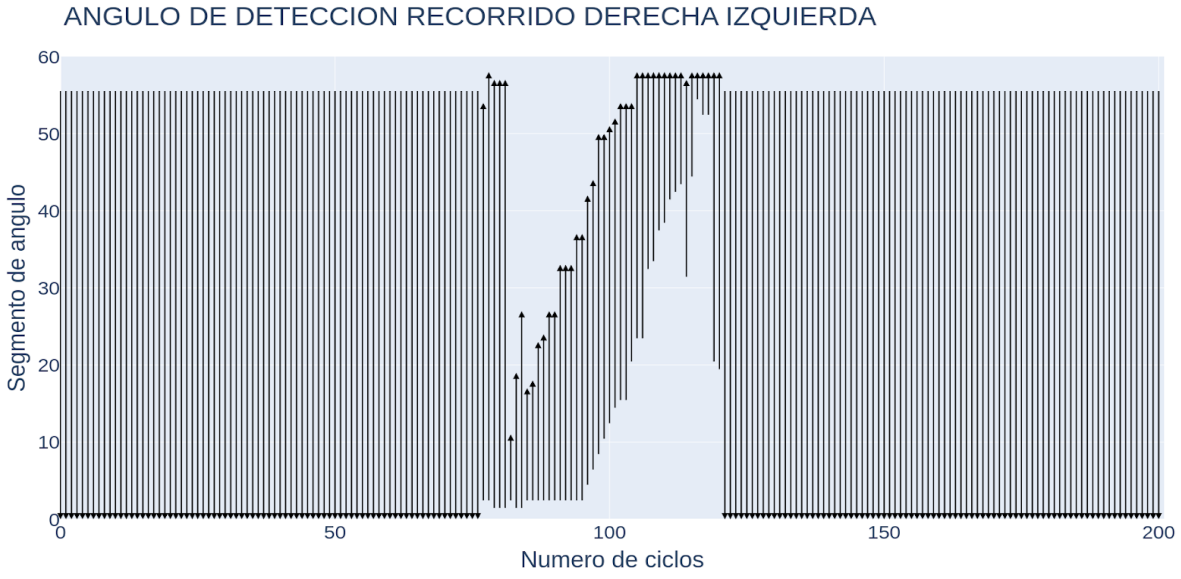
Para este experimento, se utilizó el escenario E (Tabla 2) junto con un objeto dinámico humano (Figura 10 [A]). El objeto dinámico realizó un recorrido que provocó alteraciones en la desviación estándar (dr), lo que resultó en zonas con puntos de alta variabilidad. Estas zonas fueron utilizadas para obtener los ángulos de segmentación en cada ciclo de ejecución, lo que permitió realizar el seguimiento del objeto dinámico.

En la primera prueba, se observó progresivamente la detección del objeto dinámico mientras cruzaba la escena en el 100% de los ciclos en los que estuvo presente. El

recorrido se realizó de izquierda a derecha y constó de 200 ciclos (Figura 21 [A]). Además, se llevó a cabo una prueba adicional con un recorrido de derecha a izquierda compuesto por 200 ciclos, en la cual también se obtuvo una detección progresiva del objeto dinámico en el 100% de los ciclos (Figura 21[B]).



A



B

Figura 21. Seguimiento del ángulo segmentado en cada ciclo donde fue detectado el objeto dinámico, (A) recorrido de izquierda a derecha del objeto dinámico, (B) recorrido de derecha a izquierda del objeto dinámico.

En conclusión, este experimento evaluó el rendimiento del sistema en la detección de un objeto humano en movimiento en el escenario E con una repetición de izquierda a derecha y viceversa (**¡Error! No se encuentra el origen de la referencia.**). Los resultados demostraron que el sistema posee una capacidad constante y efectiva para detectar y seguir objetosen movimiento en tiempo real. Con una tasa de detección del 100% durante el recorrido de izquierda a derecha y un porcentaje significativo del 98.5% durante el recorrido inverso, se destaca la precisión y versatilidad del sistema en la detección de objetos dinámicos. Estos hallazgos son de gran importancia para aplicaciones que requieren la detección y seguimiento de objetos en movimiento en entornos complejos, como la navegación autónoma y la seguridad en la detección de personas.

4.4 Experimento 4

En este experimento, los datos del LIDAR virtual y de la RGB-D se utilizarán como entrada para obtener las PCL de la zona estática y la PCL de la zona dinámica. El sistema podrá seleccionar automáticamente el ángulo de segmentación basado en las zonas de alta variabilidad que encuentre, lo que aumentará los factores no controlados. Es importante considerar los múltiples factores que pueden influir en este experimento. El cual estará constituido por pruebas donde se tendrá un escenario estático con objetos estáticos y la presencia el objeto dinámico el cual realiza un recorrido establecido, ver (Figura 11 [A]).

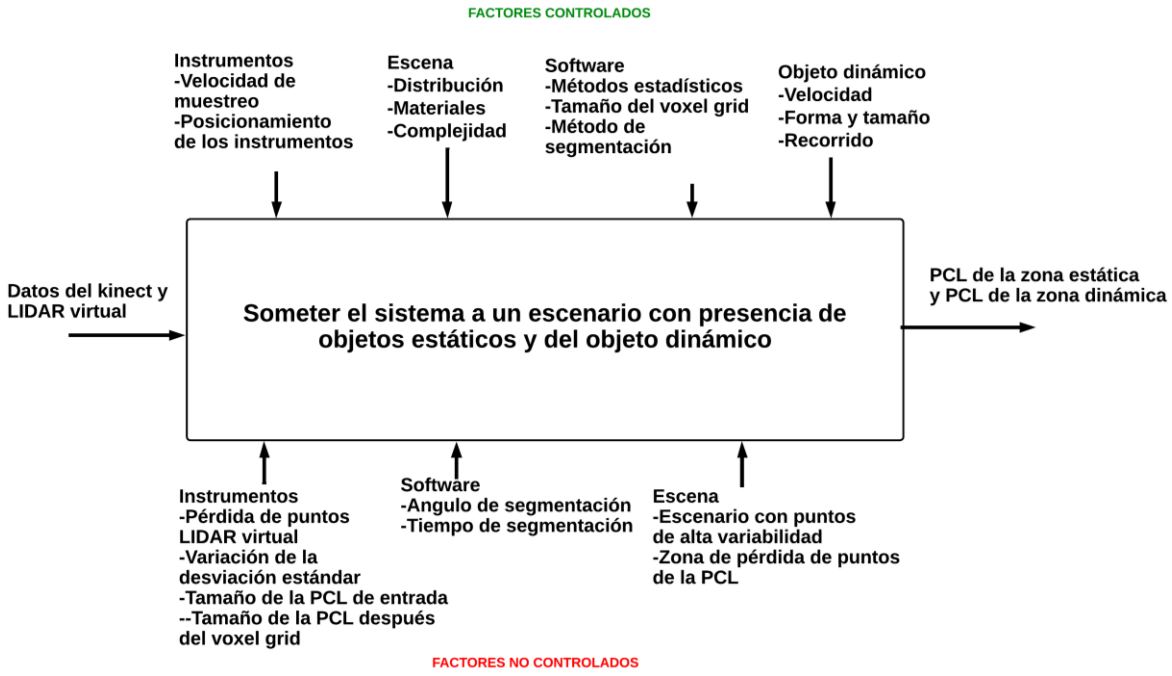
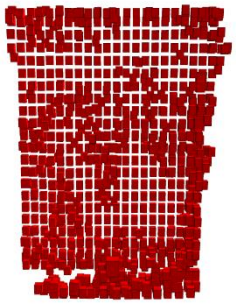


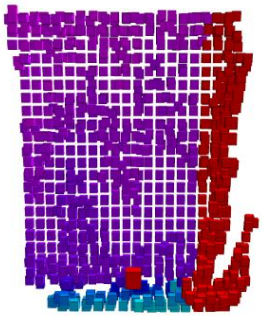
Figura 22. Factores controlados y no controlados experimento 4

En el experimento, se aplicaron dos etapas de segmentación y se realizó un seguimiento fotograma a fotograma del objeto dinámico. Se utilizó el escenario A (Tabla 2. Características de los escenarios de prueba) y un objeto dinámico (Figura 10 [A]). El objeto realizó recorridos de derecha a izquierda, cruzando el escenario de forma paralela al eje y (Figura 11[A]).

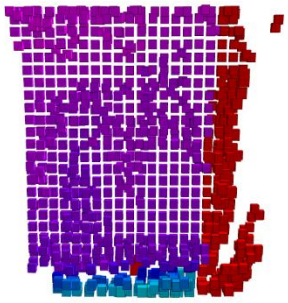
Durante la prueba, se registró la variación de la PCL en las áreas estática y dinámica aplicando una sola etapa de segmentación basada en el ángulo de detección (Figura 21). A partir de esto, se realizaron segmentaciones de los puntos en la zona dinámica donde se encontraba el objeto dinámico (denominada PCL_{ZD}, representada en rojo en la Figura 23), así como en la zona estática (denominada PCL_{ZE}, representada en morado en la Figura 23).



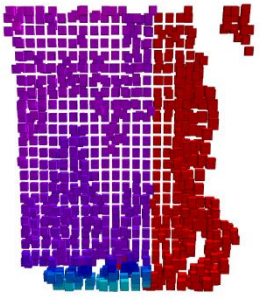
1



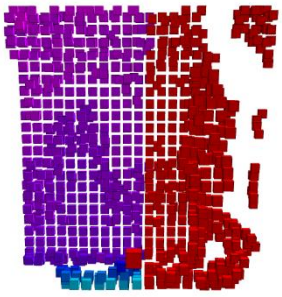
2



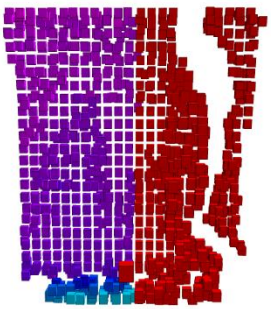
3



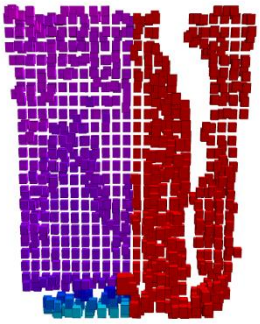
4



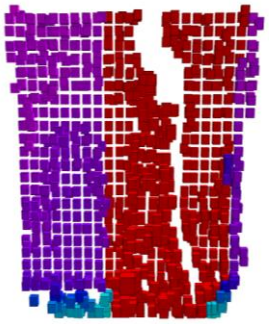
5



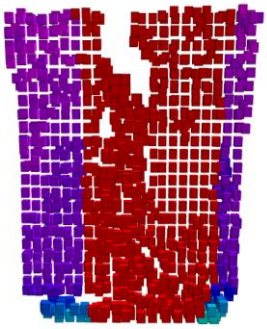
6



7



8



9

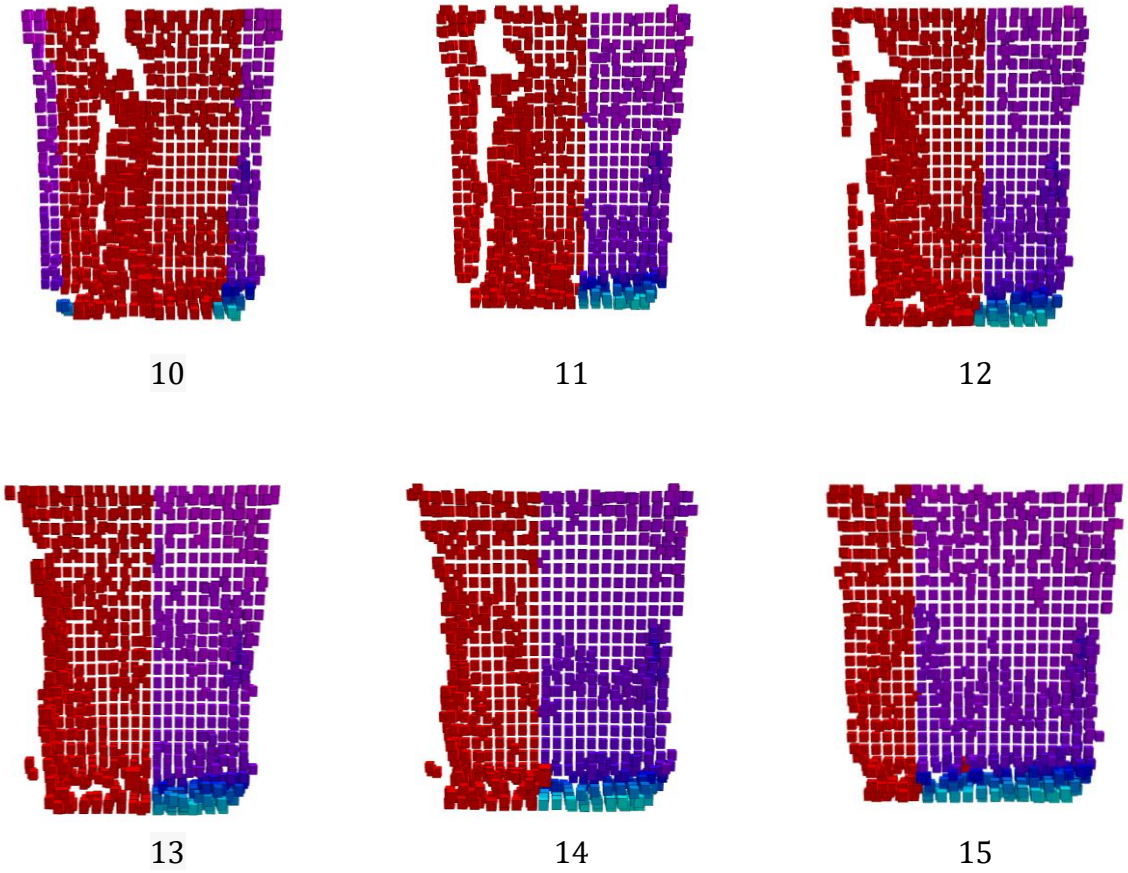


Figura 23. Aplicación del ángulo de detección sobre la PCL

La segmentación se realiza a partir de la nube de puntos completa, la cual tiene un tamaño variable en cada ciclo de ejecución. Utilizando el ángulo de segmentación como referencia, se obtiene la variación de la nube de puntos en la zona dinámica y la zona estática (Figura 24).

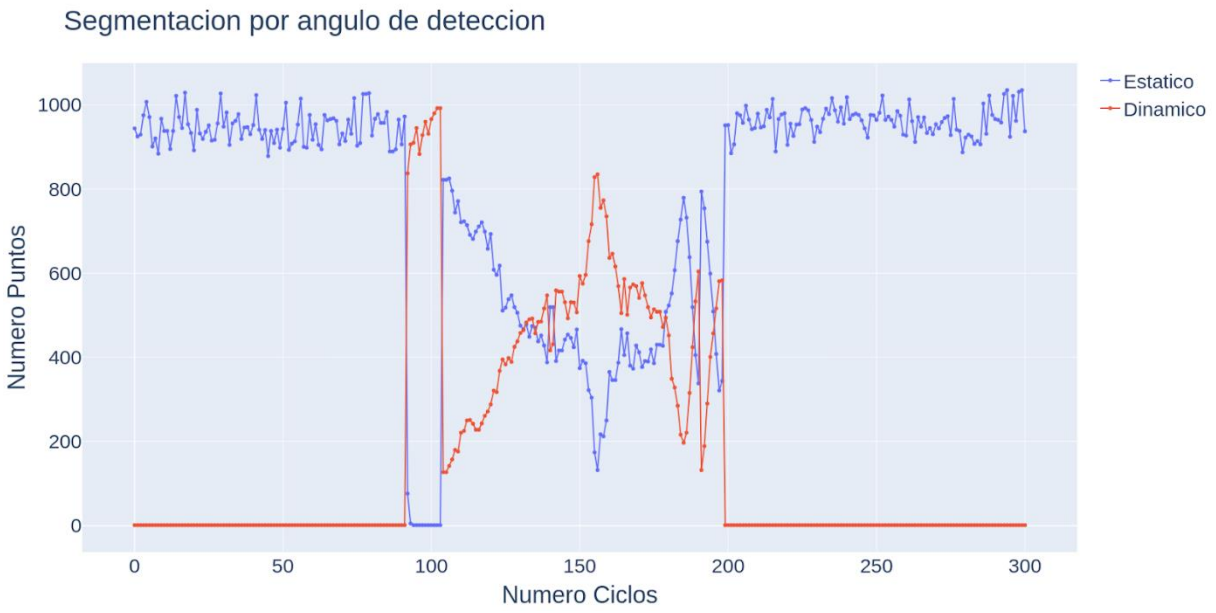
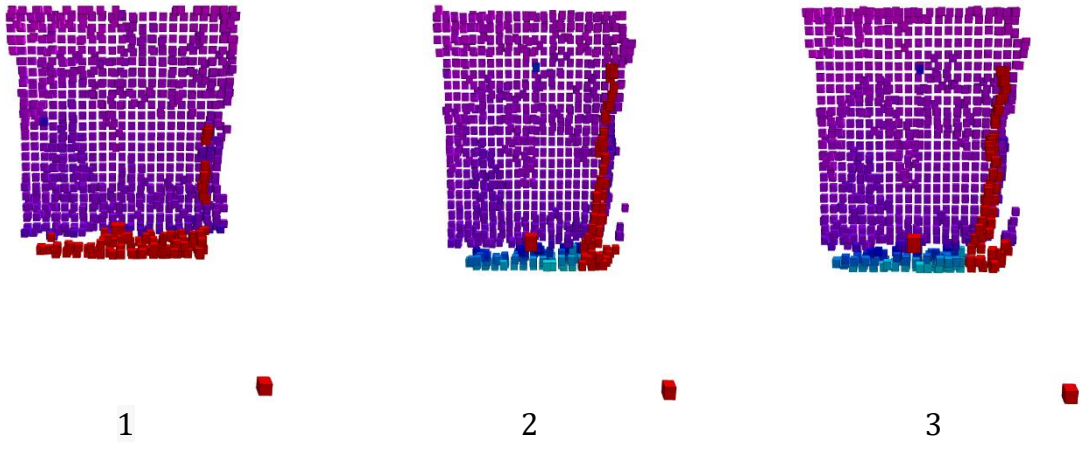
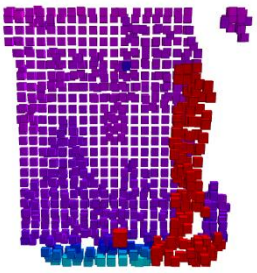


Figura 24. Variación de la nube de puntos recorrido de derecha a izquierda.

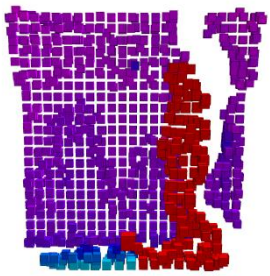
En la misma prueba, además de la segmentación por ángulo de detección, se aplicó una segunda segmentación utilizando el método de agrupación por profundidad. Este enfoque permite una segmentación más precisa del objeto dinámico (Figura 25).



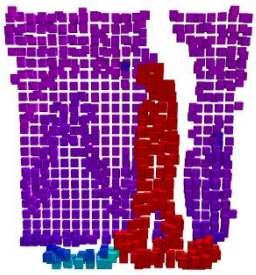
1 2 3



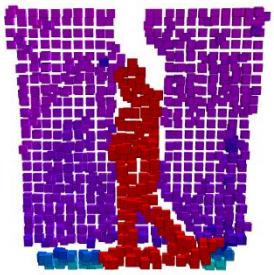
4



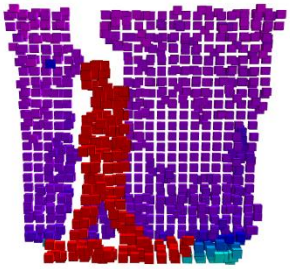
5



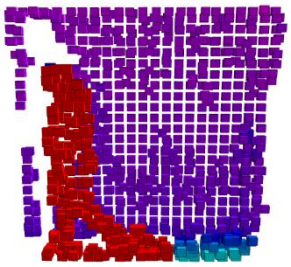
6



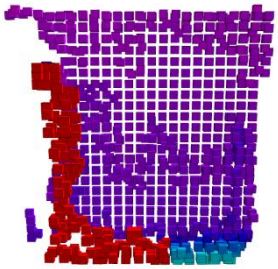
7



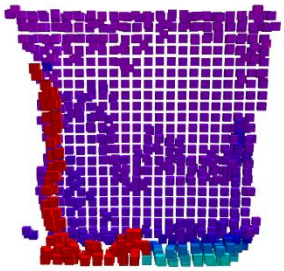
8



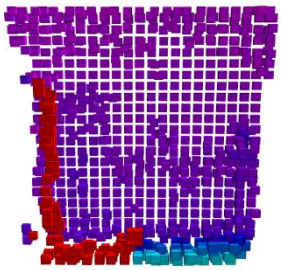
9



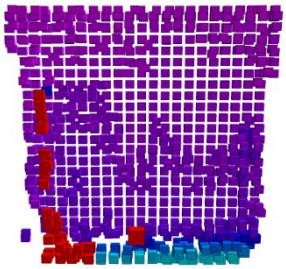
10



11



12



13

Figura 25. Seguimiento de la segmentación de objetos dinámico aplicando el método de segmentación en dos etapas.

Esta segmentación proporciona información más detallada sobre el número de puntos que componen el objeto dinámico, al mismo tiempo que reduce la pérdida de puntos correspondientes a la escena estática (Figura 26).

Figura 26. Variación del número de puntos por la segmentación de dos etapas.

Una vez que se definido el ángulo de segmentación, se obtuvieron las PCL_{ZE} y PCL_{ZD}, que representan las PCL resultantes del primer proceso de segmentación, como se muestra en la Figura 23. Al observar esta secuencia de imágenes, se puede apreciar que el objeto dinámico se encuentra predominantemente dentro de la zona roja, con la excepción de

algunas imágenes en las que algunos puntos del objeto dinámico quedan fuera de dicha zona.

Posteriormente, al analizar la Figura 25, donde se aplicó la segmentación en dos etapas en la misma prueba, se evidencia que esta segunda etapa logra segmentar el objeto dinámico con mayor precisión, generando la PCL_{Din} y la PCL_{Est-k}.

En esta prueba, se destaca el beneficio de aplicar la segmentación en dos etapas, ya que permite obtener una PCL de la zona estática, donde se recuperan una mayor cantidad de puntos (Figura 24 y Figura 26). Con la segmentación en una sola etapa, se obtiene una PCL dinámica de aproximadamente 830 puntos, lo que representa más del 70% de la PCL total. En contraste, con la segmentación en dos etapas, se logra obtener una PCL dinámica de alrededor de 200 puntos, correspondiente aproximadamente al 20% de la PCL total.

Estos hallazgos demuestran que la segmentación en dos etapas permite una mejor delimitación del objeto dinámico, resultando en una PCL más precisa y representativa de la zona estática y dinámica. Esta metodología puede ser de utilidad para aplicaciones que requieren un análisis más detallado y una distinción clara entre las partes estáticas y dinámicas de la escena capturada por el LIDAR virtual.

4.5 Experimento 5

En este experimento, se utilizarán los datos de la RGB-D y el LIDAR virtual como entrada para calcular el porcentaje de correlación entre la nube de puntos (PCL) reconstruida y la PCL de referencia. Es esencial tener en cuenta que existirán factores no controlados que podrían influir en el resultado del experimento.

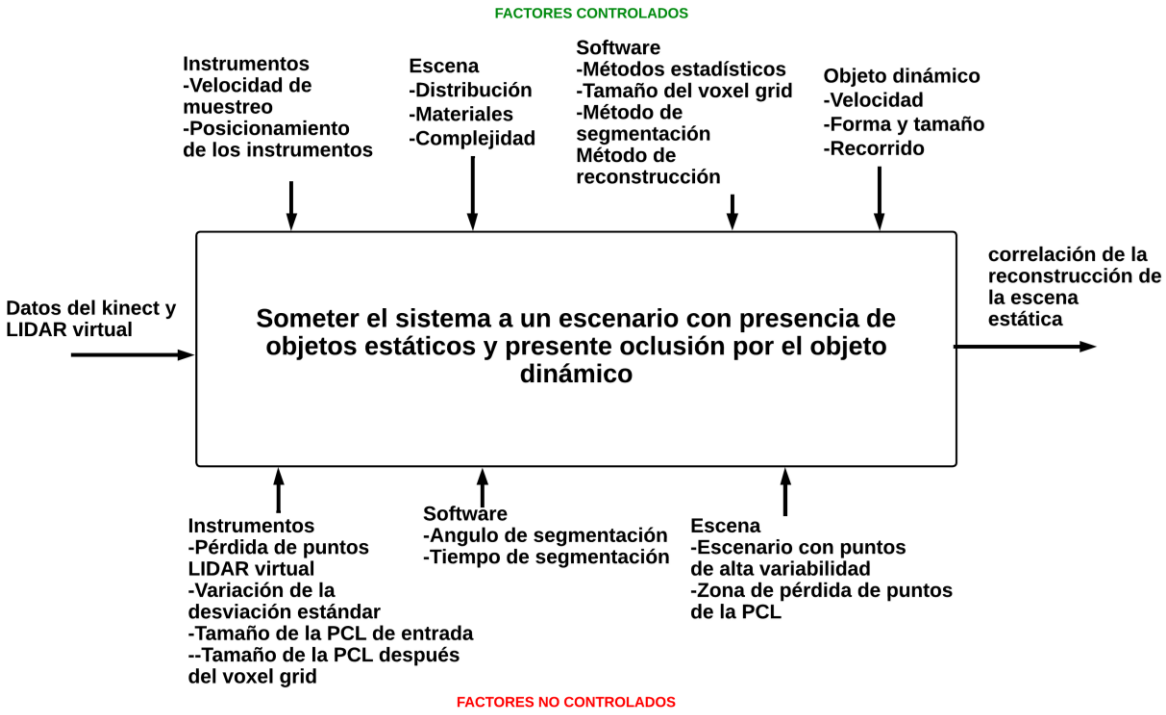


Figura 27. Factores controlados y no controlados experimento 5

Para este experimento, se llevarán a cabo pruebas con 25 ciclos cada una. En cada ciclo de prueba, se capturará una escena de referencia que posteriormente se comparará con la escena obtenida en cada ciclo. Durante cada prueba, el objeto dinámico realizará un recorrido establecido (Figura 11 [A]).

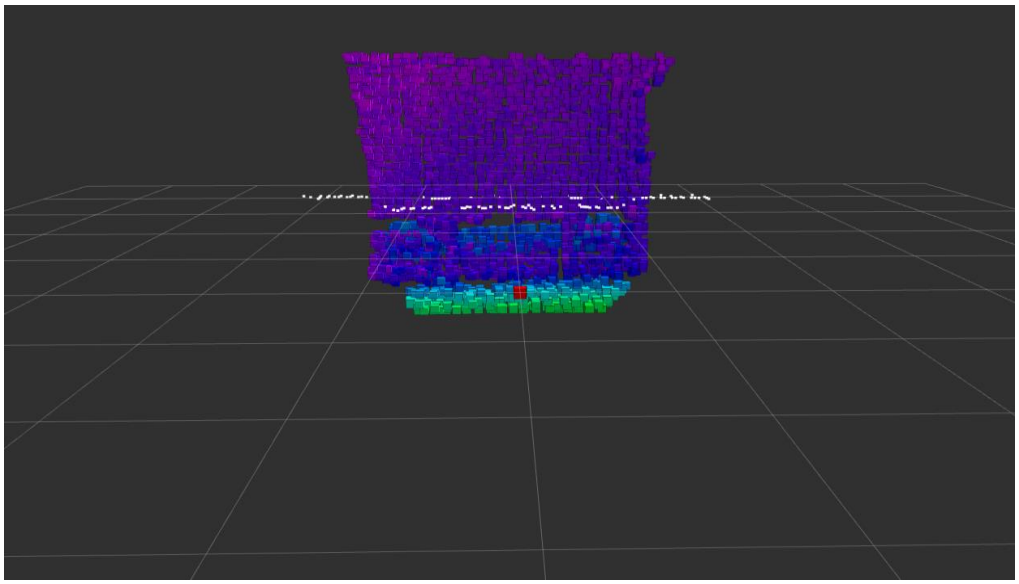
Se aplicarán dos etapas de software en el proceso. En la primera etapa, se obtendrán en línea las nubes de puntos (PCL) reconstruidas. En la segunda etapa, fuera de línea, se aplicará un método para determinar la correlación entre la escena estática reconstruida y la escena de referencia, permitiendo así evaluar el grado de similitud entre ambas.

El proceso de reconstrucción consiste en la concatenación sucesiva de las escenas para obtener una escena completa compuesta por los puntos captados en ciclos anteriores. En este experimento, se utilizó el escenario E (Tabla 2) junto con un objeto dinámico humano (Figura 10 [A]). El objeto realizó recorridos de derecha a izquierda, cruzando el escenario de forma paralela al eje Y (Figura 11 [A]).

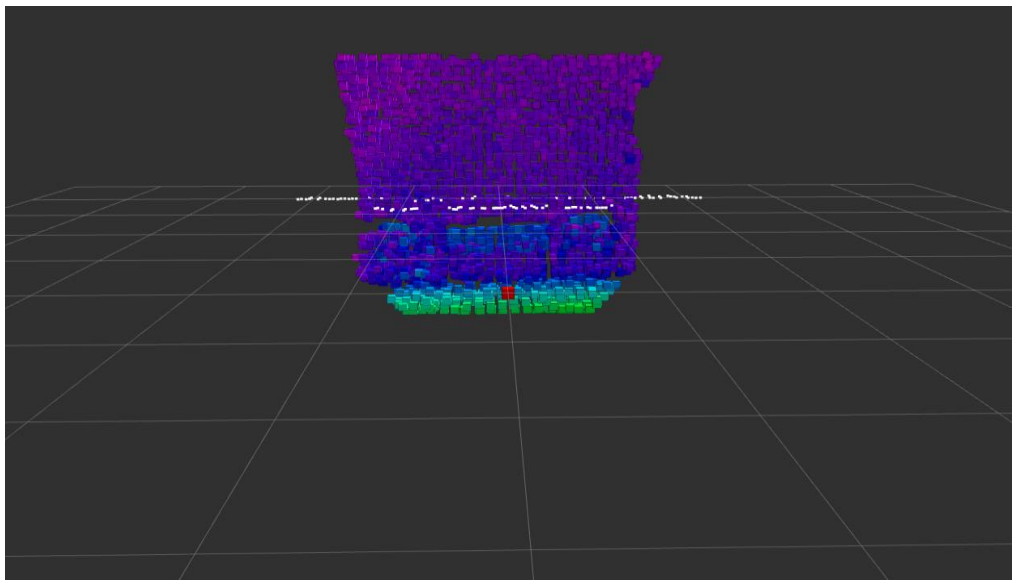
Se realizaron 10 pruebas en las cuales se llevó a cabo la concatenación y reducción de las PCL consecutivas de aproximadamente 48 ciclos. En cada prueba, se capturó primero la escena estática sin el objeto dinámico y posteriormente se capturaron las escenas

reconstruidas sucesivamente.

Durante las pruebas, se obtuvo la escena estática inicial como referencia (Figura 28[A]) y se generaron las escenas reconstruidas paso a paso para cada ciclo. Debido a la gran cantidad de escenas, se presenta una escena estática reconstruida correspondiente al ciclo final de una de las pruebas (Figura 28[B]).



A



B

Figura 28. PCL escena estática(A), PCL escena dinámica(B)

Con el fin de analizar la calidad de la reconstrucción, se calculó la correlación entre la escena estática inicial y cada una de las escenas reconstruidas obtenidas en cada ciclo y para cada prueba (Figura 29). En promedio, se obtuvo una correlación del 98.5% entre la escena estática inicial y las escenas reconstruidas. Esto indica un buen nivel de concordancia entre la escena original y las reconstrucciones realizadas.

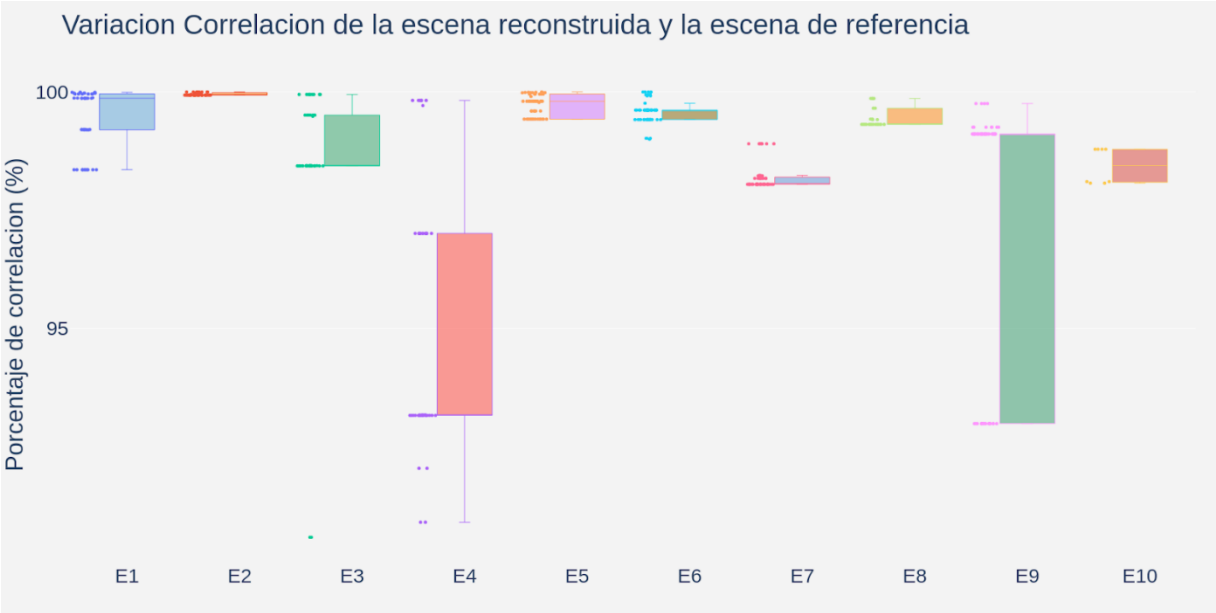


Figura 29.Variacion de la correlación entre la escena reconstruida y la escena de referencia.

En esta sección, se utiliza la escena estática como punto de referencia (Figura 28[A]). Después de realizar la reconstrucción de la escena (Figura 28[B]), se lleva a cabo una comparación visual entre ambas. Se observa una coherencia en la distribución general de las características (Tabla 2[E]). Sin embargo, se aprecia una diferencia en la textura, lo cual indica la presencia de errores inherentes al proceso de reconstrucción sucesiva. Es importante tener en cuenta que siempre existirá un factor de error debido a las limitaciones de precisión de los instrumentos utilizados.

Estos resultados demuestran que, a pesar de los desafíos inherentes al proceso de reconstrucción, el sistema Rec-HV logra una reconstrucción precisa y confiable de la escena, manteniendo una alta correlación entre la escena estática y la reconstruida. Esto evidencia la capacidad del sistema para capturar y representar de manera adecuada la información tridimensional de la escena durante el proceso de reconstrucción.

4.6 Experimento 6

En este experimento se utilizarán los datos de la RGB-D y del LIDAR virtual de distintos escenarios reales como entrada. Estos escenarios presentarán características diversas y se obtendrá como resultado la reconstrucción de la escena estática y tablas de parámetros resultantes de la ejecución. Debido a que se trata de entornos reales con objetos dinámicos, existirán más variables no controladas.

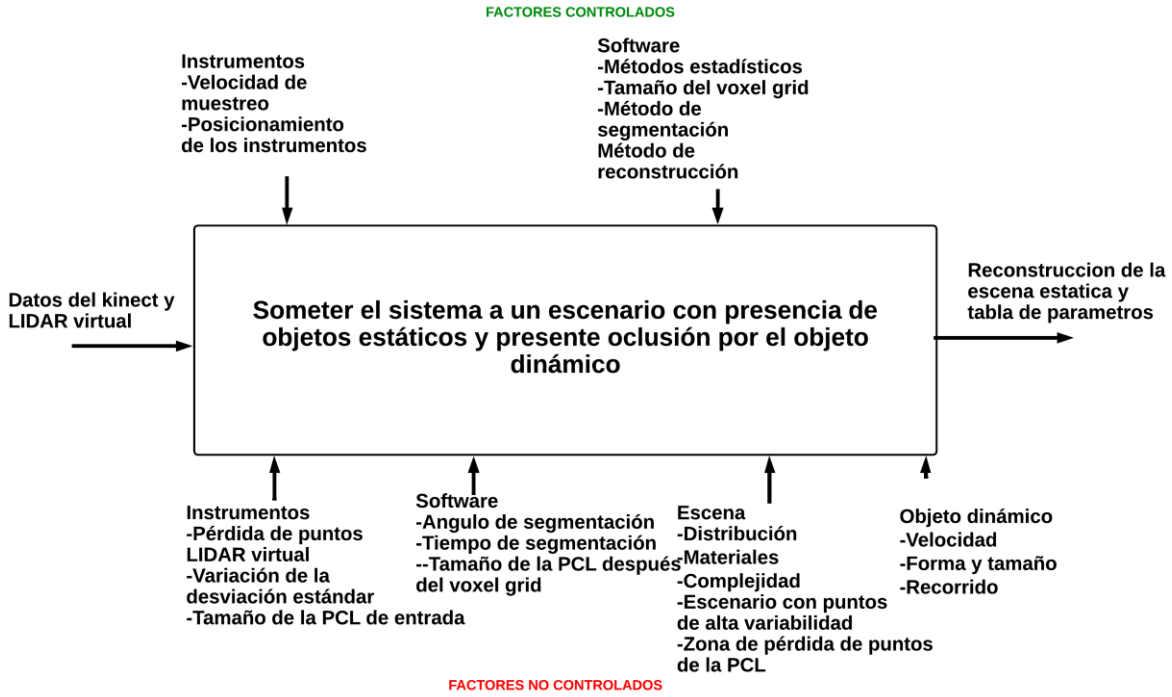


Figura 30. Factores controlados y no controlados experimento 6

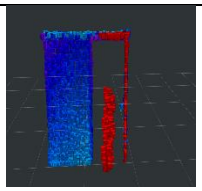
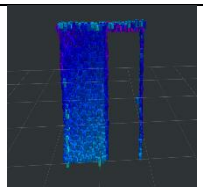
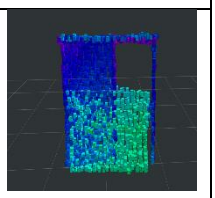
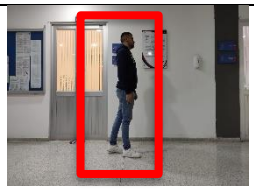
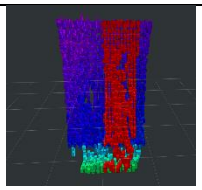
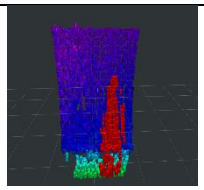
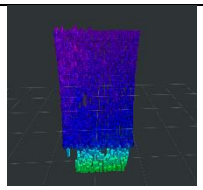
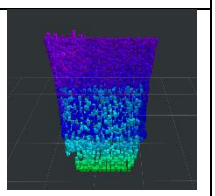
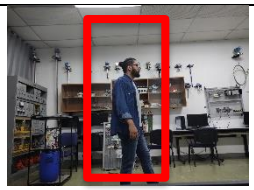
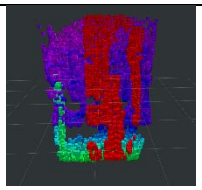
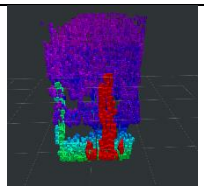
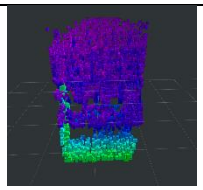
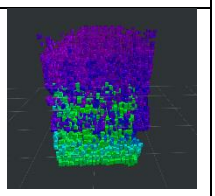
Este experimento tendrá pruebas que constan de 25 ciclos cada una. Cada prueba se desarrollará en un escenario distinto, seleccionado entre entornos interiores cotidianos. Estos entornos presentan objetos estáticos con diferentes formas y materiales, y un objeto dinámico cuya velocidad y forma no podrán ser controladas completamente. El recorrido no será lineal y tendrá variaciones en la profundidad, pero no incluirá paradas ni cambios de dirección. Se obtendrá la PCL reconstruida de cada prueba y se registrarán varios parámetros, como el tiempo promedio de ejecución, el número de puntos por ciclo, la reducción promedio de la PCL, los ciclos con puntos de alta variabilidad, el porcentaje de la PCL obtenida en la segmentación por ángulo, el porcentaje de la PCL obtenida en la segmentación por agrupación por profundidad, la PCL inicial (obtenida en el primer ciclo), la PCL final obtenida por concatenación (PCL_{Kt}) y el porcentaje de incremento de

puntos en la PCL de reconstrucción. Figura 11[D].

Para este experimento, se utilizaron los escenarios B, C, E, F, G, H, I (Tabla 2) y se empleó un humano como objeto dinámico (Figura 10 [A] [B] [C]). Se llevaron a cabo 10 pruebas en cada escenario, en las cuales el objeto dinámico realizó recorridos de derecha a izquierda e izquierda a derecha, cruzando el escenario sin seguir una trayectoria completamente lineal.

En cada uno de los escenarios, se logró realizar un seguimiento constante del objeto dinámico. Se aplicó la segmentación por ángulo (Figura 31, columna segmentación por ángulo) y la segmentación por agrupación por profundidad (Figura 31, columna segmentación por profundidad). Estas técnicas permitieron obtener una reconstrucción en la que no se observó una perturbación significativa causada por el paso del objeto dinámico en la escena (Figura 31, reconstrucción con aplicación del método).

Con el fin de proporcionar un punto de comparación, también se realizó la concatenación consecutiva de la nube de puntos sin aplicar el método de eliminación del objeto dinámico (Figura 31, columna reconstrucción sin el método). Esto permite tener una perspectiva de cómo se vería la reconstrucción sin utilizar el método de eliminación del objeto dinámico.

	Escena	**Segmentación por ángulo**	**Segmentación por agrupación de profundidad**	**Con Rec-HV**	**Sin Rec-HV**
B					
C					
E					

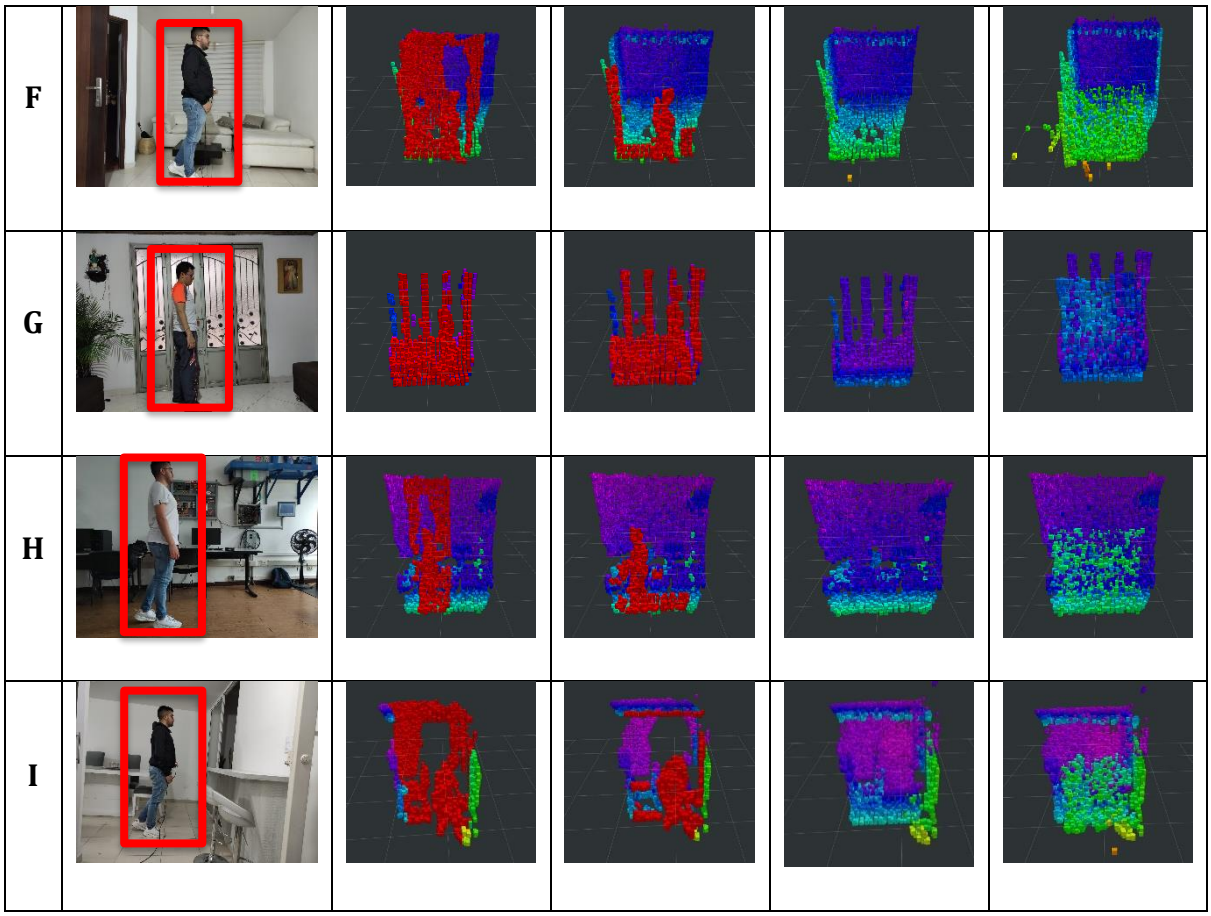


Figura 31. Evaluación en Escenarios con Objetos Estáticos y Dinámicos para Reconstrucción

Para tener un punto de comparación y evaluar la reducción en el tiempo de ejecución, se realizó el proceso de segmentación y reconstrucción sin reducir la densidad de puntos. Esto resultó en un tiempo promedio de ejecución de 3.75 segundos por ciclo. En contraste, nuestro software con las especificaciones seleccionadas logró un tiempo promedio de ejecución de 0,152370872 segundos. Durante esto, se procesaron entre 889,2600277 y 3949,839729 puntos por ciclo, lo que representó una reducción promedio del 98.64% en la PCL (nube de puntos). Además, se detectaron ciclos con puntos de alta variabilidad entre 11,20% y 26.19% de las pruebas realizadas (Tabla 4). Al aplicar la segmentación por ángulo, se encontró un promedio entre el 30,75%.

60,97%% de los puntos de la PCL fueron segmentados y formaron parte de la PCL_{ZD} (zona dinámica). Posteriormente, al aplicar la segmentación por agrupación sobre la PCL_{ZD}, el porcentaje de puntos segmentados tuvo un promedio entre 3,68%

Y 57,07% (Tabla 4). Esto demuestra cómo el proceso de segmentación por agrupación

permite refinar aún más la segmentación y reducir la cantidad de puntos considerados como parte del objeto dinámico. Sin embargo, se observó una anomalía en la escena I (Tabla 4), donde se registró un alto nivel de ruido y valores diferentes, especialmente en los ciclos de puntos con una alta variabilidad, alcanzando un valor del 100%. Esto genera una zona conflictiva que no se tiene en cuenta al establecer los rangos de porcentajes. No obstante, los valores en las segmentaciones son similares a las demás escenas. Se hace mención de este punto específico debido a la detección del 100% de puntos de alta variabilidad.

Escena	Tiempo promedio de ejecución	Puntos por ciclo	Reducción promedio de pointcloud	Ciclos con puntos de alta variabilidad	Segmentación por ángulo	Segmentación por agrupación
B	0,18512886	3949,839729	98,02%	26,19%	17,79%	17,11%
C	0,14999351	2190,864717	98,97%	15,72%	36,19%	21,26%
E	0,134091003	3214,418985	97,88%	22,91%	30,75%	9,65%
F	0,176720318	2872,591928	98,81%	20,63%	50,75%	3,68%
G	0,112457946	889,2600277	99,49%	11,20%	60,97%	57,07%
H	0,140198466	2558,504303	98,68%	24,44%	36,97%	13,09%
I	0,168006	3023,998	98,67%	100%	23,89%	7,09%

Tabla 4 Variación de los tiempos de ejecución y ciclos de segmentación y porcentaje de segmentación en cada prueba.

La aplicación del proceso de reconstrucción en cada ciclo genera una PCL_{kt} (PCL de reconstrucción) que aumenta progresivamente su número de puntos. Comenzando desde una PCL inicial, se obtiene la PCL_{kt} a medida que se realizan las sucesivas concatenaciones de ciclos. Este proceso resultó en un aumento de entre el 146,36%.

y el 205,15% en comparación con la PCL inicial, como se muestra en la Tabla 5. Esto indica que la reconstrucción sucesiva agrega puntos adicionales a la PCL, lo que contribuye a obtener una representación más completa de la escena a lo largo de los ciclos.

Escena	PCL Inicial	PCL KT	Porcentaje de incremento
B	1187,608597	3246,140271	173,33%
C	1387,357143	3798,928571	173,82%
E	2041,616393	6230,081967	205,15%
F	2175,319101	5359,074157	146,36%

G	458,0138504	1155,15097	152,21%
H	1679,232759	4802,301724	185,98%
I	1562,74844	4479,14137	186,62%

Tabla 5. Variación de la PCL al aplicar la reconstrucción.

4.7 Experimento 7

En este experimento, se llevarán a cabo pruebas en un escenario con objetos estáticos y la presencia de un objeto dinámico. El objeto dinámico realizará tres recorridos, con un primer recorrido de izquierda a derecha, seguido de un segundo recorrido de derecha a izquierda (Figura 11 [A]). Finalmente, habrá un tercer recorrido de izquierda a derecha con variaciones en la profundidad y viceversa (Figura 11 [B]). Cada prueba consta de 25 ciclos.

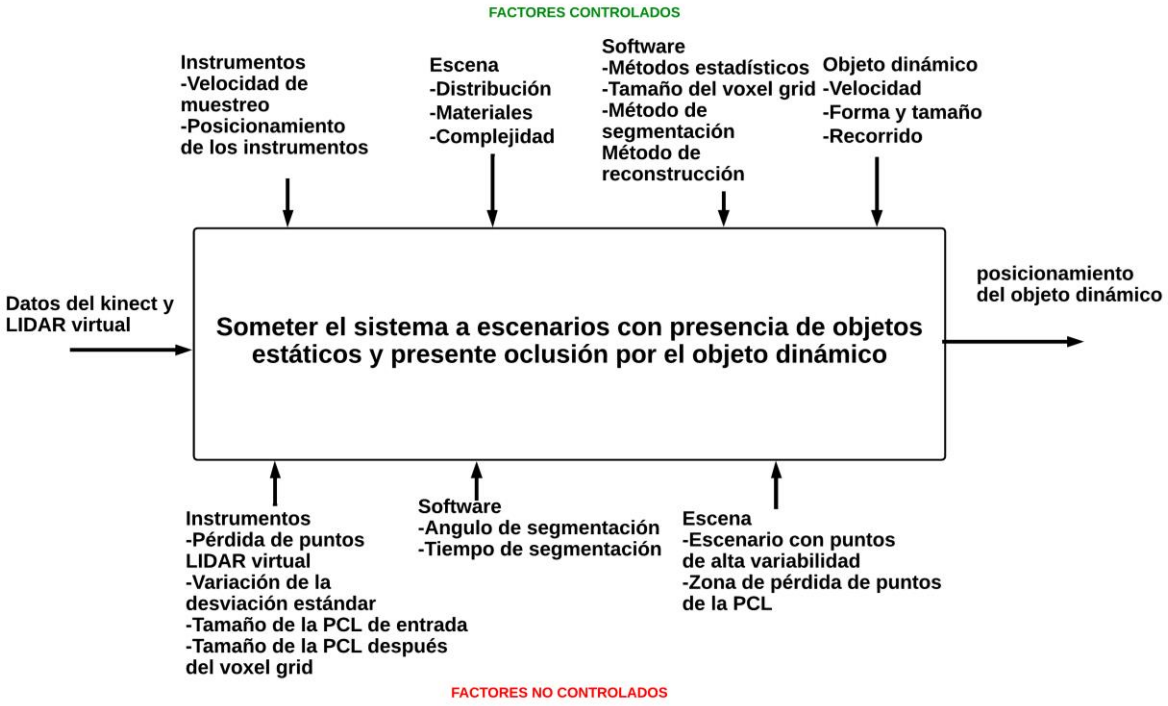
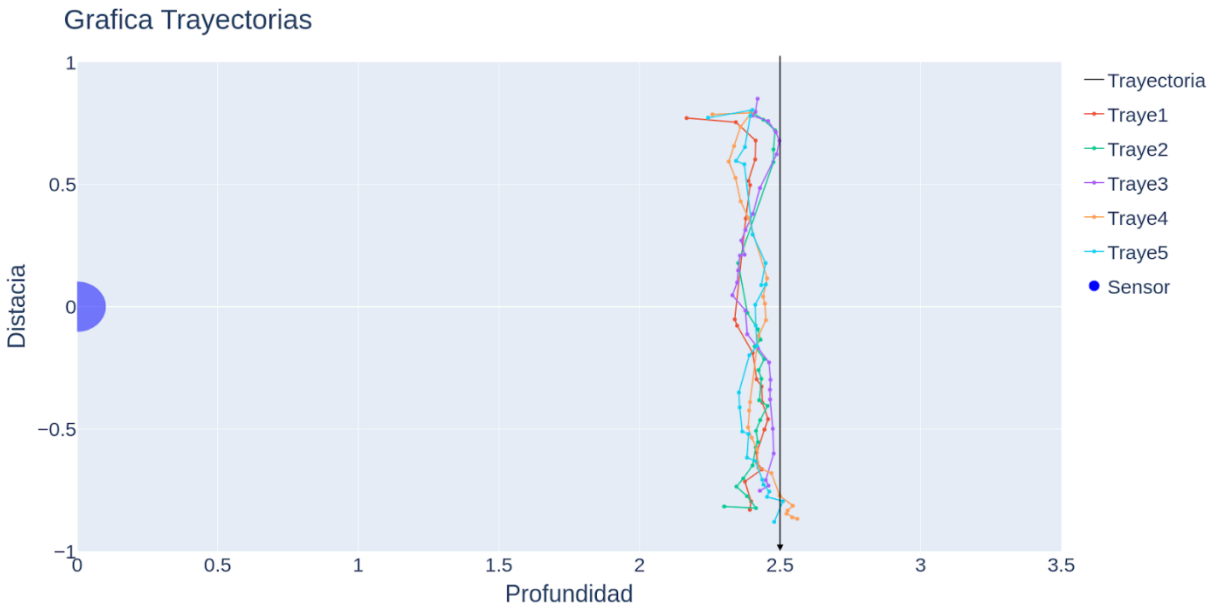


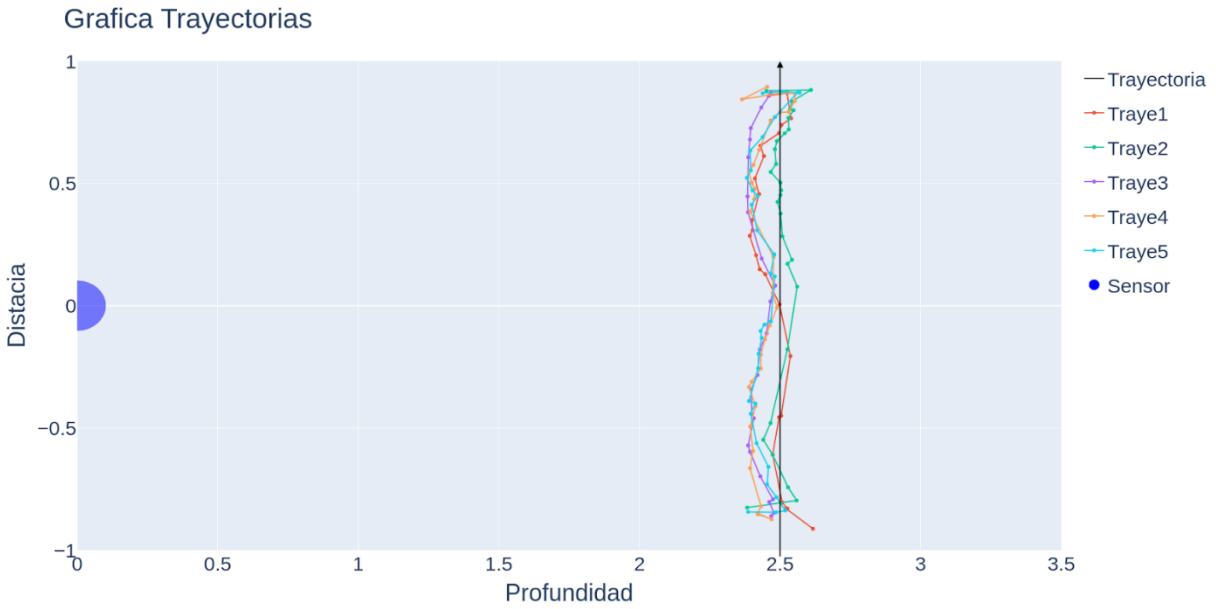
Figura 32. Factores controlados y no controlados experimento 7

En este experimento, se utilizó el escenario E (Tabla 2) y se empleó un objeto dinámico humano (Figura 10 [A]). Se llevaron a cabo un total de 15 pruebas en las que el objeto

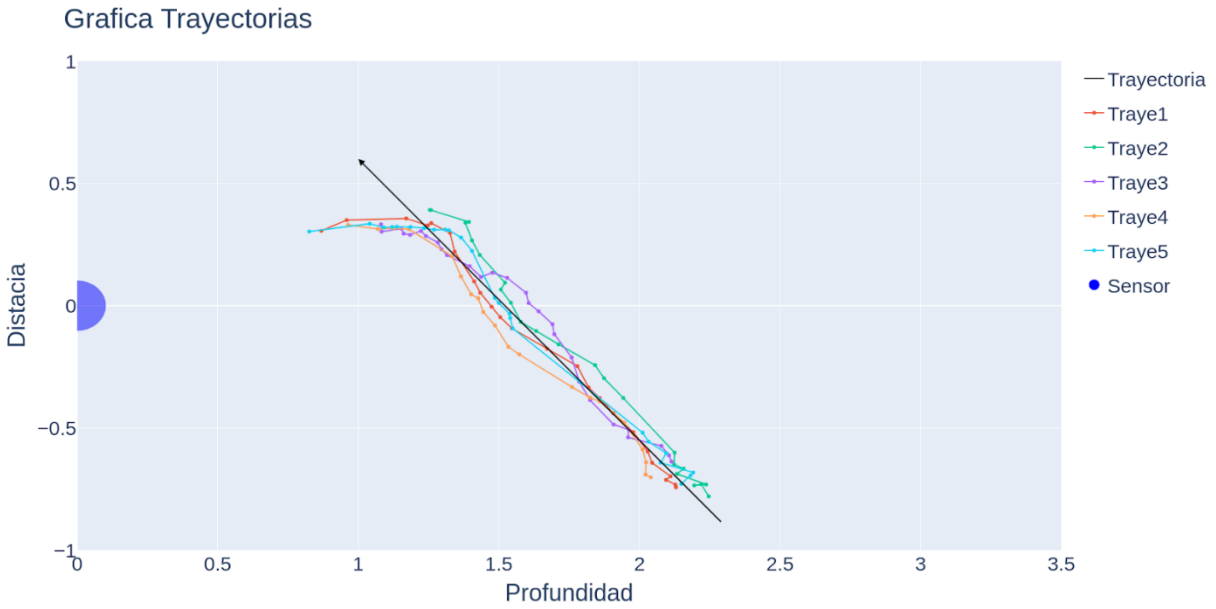
dinámico atravesó la escena siguiendo diferentes trayectorias. Mediante la aplicación de la segmentación en dos etapas, se determinaron los puntos que pertenecen a la PCL del objeto dinámico, lo que permitió obtener un centroide representativo calculado a partir de las coordenadas promedio de dichos puntos. Este centroide se utilizó para determinar la posición del objeto dinámico en cada ciclo de las pruebas. Las cuales se distribuyeron de la siguiente manera: se realizaron 10 pruebas con un recorrido lineal perpendicular al eje y (Figura 11[A]), de las cuales 5 se realizaron en sentido de izquierda a derecha (Figura 33 [A]) y las otras 5 en sentido de derecha a izquierda (Figura 33 [B]). Además, se llevaron a cabo 5 pruebas con un recorrido transversal a la escena (Figura 11[B]), en sentido de izquierda a derecha (Figura 33 [C]).



A



B



C

Figura 33. Seguimiento al recorrido del objeto dinámico, (A) seguimiento de 5 recorridos paralelos eje y de izquierda a derecha. (B) seguimiento de 5 recorridos paralelos eje y de derecha a izquierda. (C) seguimiento a 5 recorridos transversales del objeto dinámico.

En la Tabla 6 se muestran los resultados del análisis de los recorridos presentados en las Figura 33 (A y B), en cuanto al error máximo, el error absoluto y el error cuadrático medio de la medida de la profundidad.

Estos valores proporcionan información sobre la precisión de la medida de la profundidad en relación con la posición real del objeto dinámico durante los recorridos.

Recorrido	Máximo error(m)	Error absoluto(m)	Error cuadrático medio(m)
1	0,1169753075	0,04844215055	0,06078271913
2	0,1170909405	0,03377063469	0,04399305823
3	0,1154819941	0,07270123383	0,08035369722
4	0,135198491	0,06804939024	0,0765747302
5	0,1178064346	0,06428356456	0,0724577924
Promedio	0,1205106335	0,05744939477	0,06683239944

A

Recorrido	Máximo error(m)	Error absoluto(m)	Error cuadrático medio(m)
1	0,3321323054	0,1091302111	0,1241599631
2	0,1987804307	0,0861853563	0,09561625022
3	0,1695423201	0,0779609084	0,09178707666
4	0,2403152784	0,09033857801	0,105883148
5	0,2556906044	0,09542013968	0,1076808779
Promedio	0,2392921878	0,09180703871	0,1050254632

B

Tabla 6.Máximo de error, error absoluto y error cuadrático medio(A) recorridos de izquierda a derecha, (B) de derecha a izquierda

La segmentación en dos etapas se implementó como estrategia para lograr el seguimiento preciso del objeto dinámico en la escena. Al analizar los resultados, se observa una consistencia en el seguimiento del objeto durante los recorridos. Sin embargo, se identifica una desviación en los extremos del recorrido, donde el sistema tiende a alejarse de la trayectoria real. Esta desviación se debe a la captación de puntos estáticos en el fondo al alcanzar los límites del campo visual del sensor. Esta inclusión de puntos adicionales

altera los cálculos promedio utilizados para determinar la posición del objeto dinámico. En el caso de la prueba de recorrido transversal (Figura 33), se observa una mayor consistencia en el seguimiento en el lado más alejado del fondo, lo que sugiere que el sistema es más efectivo para detectar el objeto dinámico cuando está más separado de la escena estática.

Estos hallazgos resaltan la relevancia de la segmentación en dos etapas para lograr un seguimiento coherente del objeto dinámico en la escena. No obstante, es importante tener en cuenta la desviación observada en los extremos debido a la inclusión de puntos estáticos del fondo. Asimismo, los valores de error obtenidos en el análisis del posicionamiento subrayan la necesidad de mejorar la precisión del seguimiento del objeto dinámico con el fin de reducir aún más los errores de posicionamiento en futuras aplicaciones.

4.8 Experimento 8

Durante este experimento, se utilizará datos de RGB-D y LIDAR virtual como entrada, y se evaluará la capacidad de detección de objetos dinámicos en diferentes condiciones. Durante el transcurso del experimento, habrá factores no controlados que podrían influir en los resultados obtenidos.

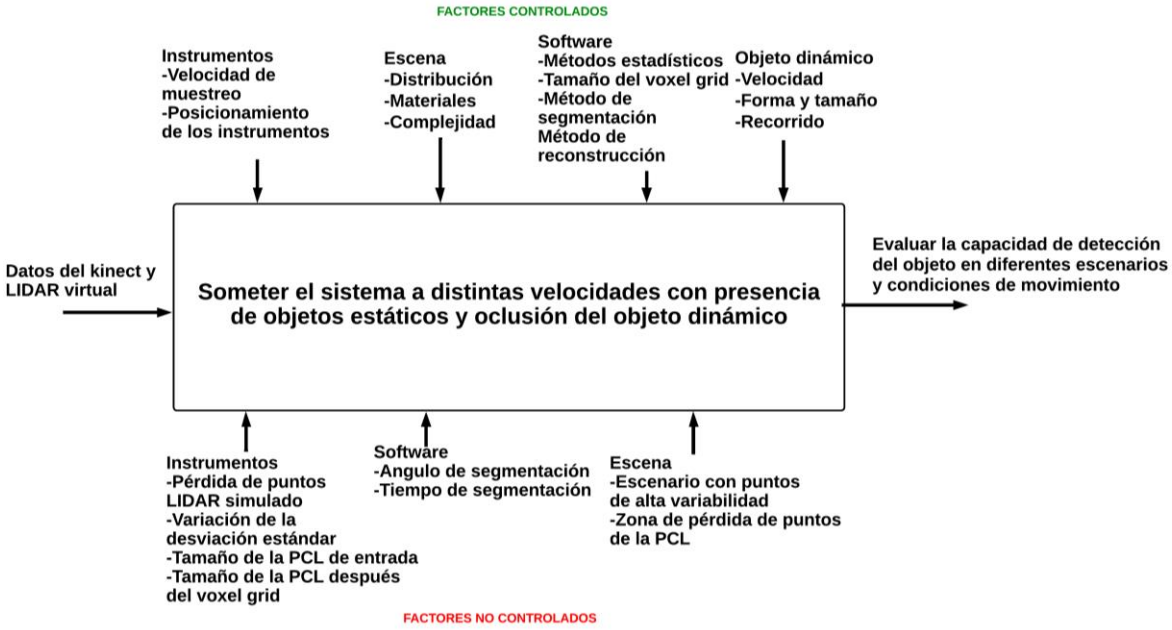


Figura 34. Factores controlados y no controlados experimento 8

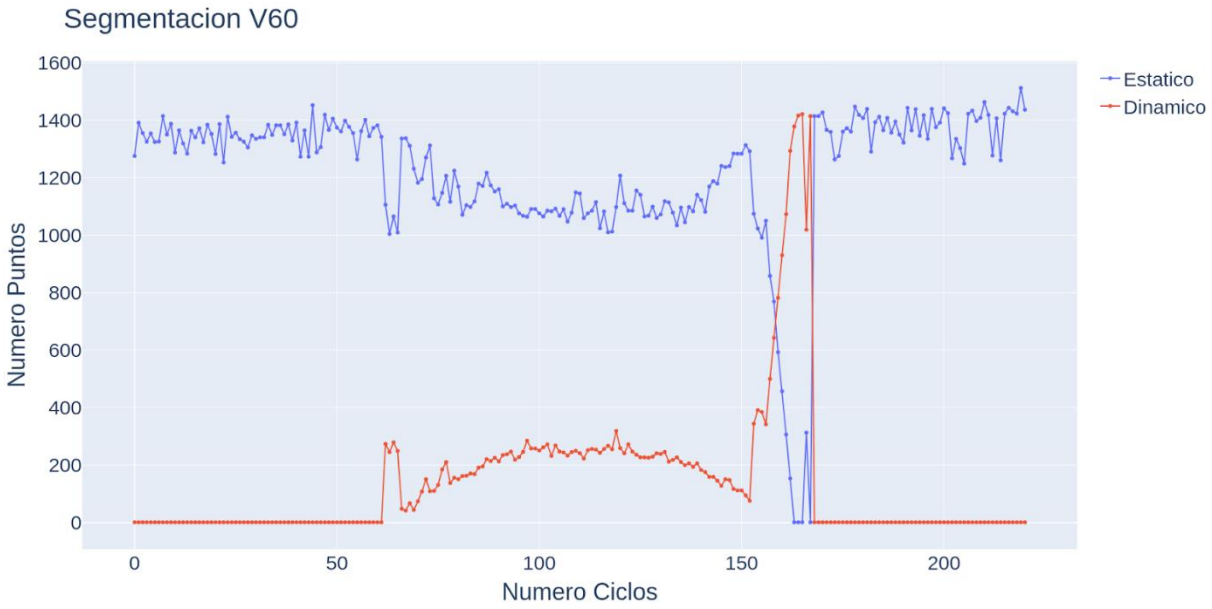
En este experimento se realizaron tres pruebas a diferentes velocidades con el objetivo de determinar si el sistema es capaz de lograr una reconstrucción precisa, a pesar de los posibles disturbios que puedan generarse.

Con el propósito de demostrar las diferentes velocidades de un objeto dinámico y garantizar su detección, se llevó a cabo un experimento en la escena E (Tabla 2), que consistió únicamente en una pared plana. El recorrido se realizó de izquierda a derecha (Figura 11[A]) y se realizaron 220 ciclos en total. Para lograr la precisión en el recorrido, se utilizó un metrónomo desde un celular como referencia, brindando una guía constante y asegurando que el objeto se mantuviera en compás y ajustara su ritmo de manera precisa.

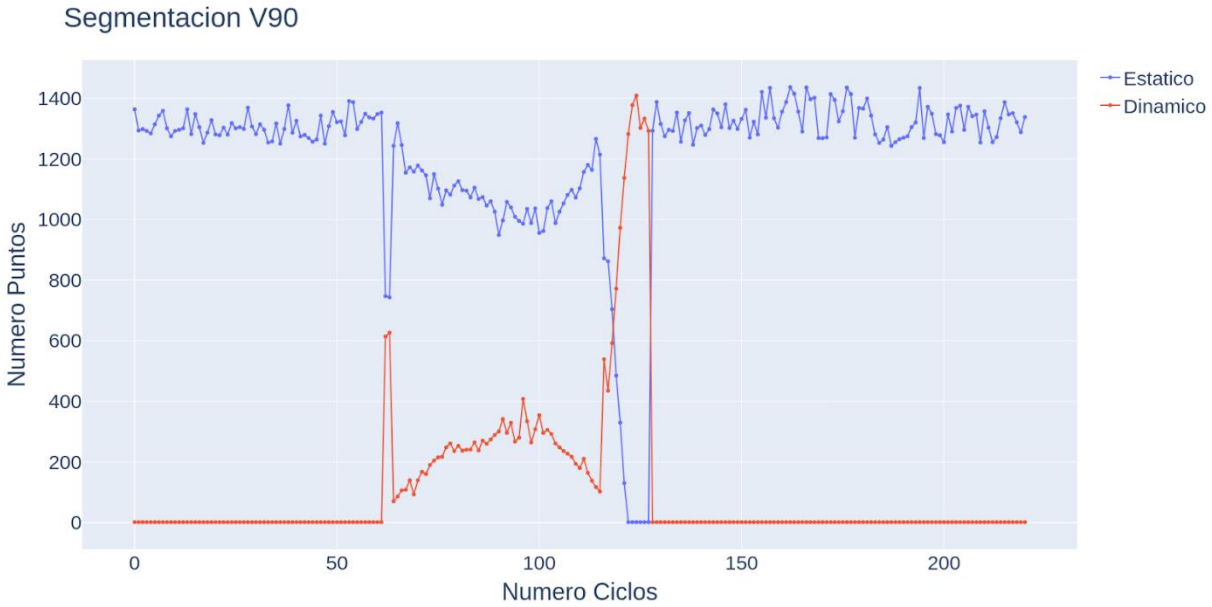
En cada experimento, se varió el tempo del metrónomo a tres velocidades diferentes: 60, 90 y 110 compases por minuto. Esta variación en la velocidad permitió evaluar la capacidad de detección del objeto en diferentes escenarios y condiciones de movimiento. Se observó que, a velocidades más bajas, se completaron más ciclos, mientras que, a velocidades más altas, se redujo la cantidad de ciclos.

Estos resultados revelaron una relación inversa entre la velocidad del objeto y la cantidad de ciclos realizados. A menor velocidad, se registraron más ciclos, lo que indica una menor

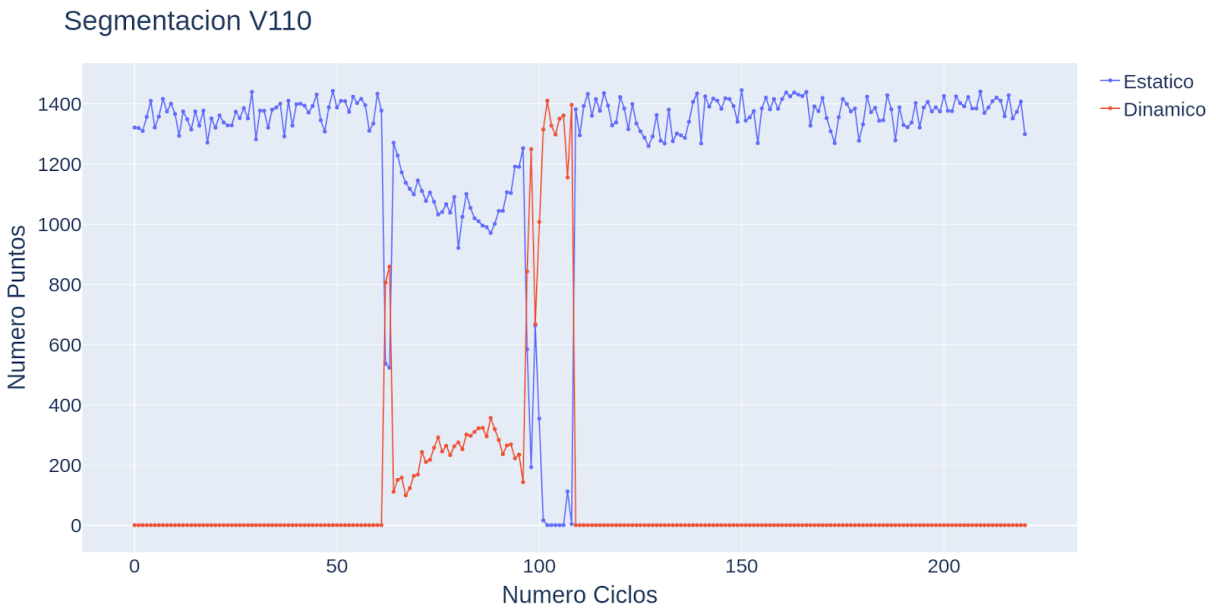
distancia recorrida en un período de tiempo dado. Por otro lado, a mayor velocidad, se completaron menos ciclos, lo que indica una mayor distancia recorrida en el mismo período de tiempo. A continuación, se muestra, los resultados de los experimentos, (Figura 35).



A



B



C

Figura 35. Seguimiento al recorrido del objeto dinámico, (A) seguimiento de 5 recorridos paralelos eje y de izquierda a derecha. (B) seguimiento de 5 recorridos paralelos eje y de derecha a izquierda. (C) seguimiento a 5 recorridos transversales del objeto dinámico.

La prueba de las velocidades realizada, como se puede observar en la Figura 35. Seguimiento al recorrido del objeto dinámico, (A) seguimiento de 5 recorridos paralelos eje y de izquierda a derecha. (B) seguimiento de 5 recorridos paralelos eje y de derecha a izquierda. (C) seguimiento a 5 recorridos transversales del objeto dinámico.. Seguimiento al recorrido del objeto dinámico, (A) seguimiento de 5 recorridos paralelos eje y de izquierda a derecha. (B) seguimiento de 5 recorridos paralelos eje y de derecha aizquierda. (C) seguimiento a 5 recorridos transversales del objeto dinámico, arrojó resultados satisfactorios. Se pudo comprobar que existe una relación inversa entre la velocidad del objeto dinámico y la cantidad de ciclos realizados. A velocidades más bajas, se completaron más ciclos, mientras que a velocidades más altas se registró una disminución en la cantidad de ciclos como se evidencia en la (Tabla 7). Estos hallazgos resaltan la importancia de tener en cuenta la velocidad al diseñar sistemas de detección, y subrayan la necesidad de optimizar algoritmos y estrategias para mejorar la precisión en la detección de objetos en movimiento en distintos escenarios y velocidades.

	Promedio PCL_IN	**Promedio PCL_DINAMICA**	**Porcentaje PCL_DINAMICA**	**Ciclos Objeto dinámico**
V60	2002	280	13.98%	107
V90	1976	266	13.46%	67
V110	1989	243	12.22%	48

Tabla 7. Resultados para cada velocidad

Para efectuar una comparación entre el trabajo previamente citado y el presente estudio, se han dispuesto la (Tabla 8) que exhibe los resultados de los experimentos realizados en ambos contextos, permitiendo así una evaluación detallada de las diferencias y similitudes entre ambas investigaciones.

VARIABLE	**REC-HV-1(dinámico robot)**	**REC-HV-2 (dinámica persona)**
Tiempo de ejecución sin voxelgrid	4.21 seg	3.75 seg
Rango tiempo de ejecución	0.14 - 0.18 seg	0.15 - 0.22 seg
Rango tiempo de segmentación	0.01 - 0.1 seg	0.01 - 0.06 seg
Rango correlación (min-max)	78 - 95 %	92 - 100 %
Promedio de correlación	86.43 %	98.5 %
Rango error máximo, recorrido A-B	(0,1199 - 0,1323), (0,1088 - 0,2099) m	(0,1154 - 0,1351), (0,1695 - 0,3321) m
Rango error absoluto,	(0,0447 - 0,0658), (0,0459 -	(0,0337 - 0,0727), (0,0779 -

recorrido A-B	0,0624) m	0,1091) m
Rango error cuadrático, recorrido A-B	(0,0599 - 0,0778), (0,0562 - 0,0848) m	(0,0439 - 0,0803), (0,0917 - 0,1241) m

Tabla 8. Comparativo trabajo anterior/actual

Capítulo 5.
Conclusiones y Recomendaciones

5.1 Conclusiones

- El análisis de los puntos LIDAR virtual en diversos escenarios revela que su variabilidad puede ser afectada por múltiples factores, siendo la complejidad del entorno uno de los más significativos. Sin embargo, se ha constatado que la aplicación de técnicas estadísticas en el procesamiento de datos permite mitigar dicha variabilidad y recuperar zonas afectadas. Esto resulta especialmente relevante para identificar áreas que experimentan alteraciones debido a la presencia de objetos dinámicos en la escena. Por lo tanto, el enfoque estadístico se posiciona como una estrategia eficaz para obtener un mayor entendimiento de los datos LIDAR y su comportamiento en diferentes situaciones.

- Tras el minucioso análisis de las escenas estáticas reconstruidas y sometidas a un método de reducción de puntos, se constata que, a pesar de la drástica disminución en el número de puntos, se logra preservar las características principales de la escena. Esta preservación incluye la distribución espacial, la forma y la constitución, lo cual respalda la propuesta planteada en esta investigación de procesar nubes de puntos reducidas como una estrategia efectiva para disminuir la carga computacional. Este enfoque demuestra la capacidad de obtener resultados satisfactorios al procesar una cantidad reducida de puntos, sin comprometer la calidad ni la capacidad de análisis de la escena.

- Este estudio ha logrado exitosamente la sustitución de un LIDAR físico por un LIDAR virtual mediante la utilización de la cámara RGB-D (Kinect). Esta transición ha demostrado la versatilidad y adaptabilidad del sistema, al mismo tiempo que ha contribuido de manera significativa a la mejora de su eficiencia, disminuyendo notablemente el peso computacional. Esta optimización no solo aumenta la eficiencia general del sistema, sino que también amplía las posibilidades de aplicaciones en

tiempo real y en sistemas con recursos computacionales limitados, subrayando el potencial de la tecnología en la captura y procesamiento de datos 3D en entornos dinámicos.

- Después de analizar los resultados obtenidos en la investigación, se concluye que se ha contribuido a la comprensión del proceso de reconstrucción de escenas estáticas con objetos dinámicos. Se destacan los aportes del análisis de la variabilidad en los datos de los sensores en presencia de objetos dinámicos. Esto ha dado lugar a un nuevo método de detección, segmentación y eliminación de objetos dinámicos, implementado en un software desarrollado. Es importante destacar que este software es de bajo costo computacional y se ejecuta en línea a 25Hz, lo cual se logra mediante la capacidad de ajustar la velocidad de muestreo en el aplicativo. Esto representa un avance en los sistemas autónomos y en la integración de sistemas robóticos en entornos con humanos, ya que se han reducido los tiempos de ejecución y se ha aumentado la frecuencia de muestreo logrando que incluso en computadoras con recursos limitados para este tipo de sistemas se desenvuelva muy bien.

- Después de evaluar el rendimiento del sistema durante su ejecución, se llegó a la conclusión de que ejecutarlo en un solo hilo no era factible. Para abordar esta limitación, se implementó una distribución de procesos en múltiples nodos, aprovechando los hilos disponibles en los núcleos del procesador. Esta estrategia permitió optimizar el uso de los recursos de hardware disponibles y garantizar una ejecución eficiente del sistema.

- Tras analizar los resultados obtenidos en la correlación de los diferentes escenarios, se llega a una conclusión contundente: el método desarrollado para la detección y eliminación de objetos dinámicos en esta investigación demuestra ser altamente efectivo. Se ha logrado alcanzar una correlación máxima del 98.5%, en comparación con otros resultados [70] que sirvieron como base para este estudio, lo cual confirma su destacada capacidad para superar la oclusión y reconstruir con precisión la escena estática. Este método permite recuperar de manera precisa la forma, estructura y distribución de dicha escena, asegurando la fidelidad de los resultados obtenidos.

5.2 Recomendaciones

- Se sugiere ampliar el área de preprocesamiento como estrategia para lograr una mayor zona de detección. Esto implica el uso de dos dispositivos Kinect para ampliar el campo visual y abarcar un área más extensa. Asimismo, se recomienda explorar la posibilidad de implementar y simular múltiples dispositivos LIDAR virtual simultáneamente. Esta estrategia permitiría obtener una cobertura más amplia y detallada de la escena, mejorando la precisión y la capacidad de detección del sistema. La combinación de diferentes sensores y tecnologías de captura de datos puede potenciar el rendimiento del sistema y brindar resultados más robustos en términos de detección y reconstrucción de escenas.

- Se sugiere la implementación de un controlador o una red neuronal que permita al sistema adaptarse a diversos escenarios y realizar una sintonización automática. Dado que el proceso de sintonización puede variar según las características específicas de cada escenario, contar con un controlador o una red neuronal flexible y adaptable sería beneficioso. Esto permitiría al sistema ajustar automáticamente sus parámetros y configuraciones según las condiciones del entorno, optimizando así su rendimiento y capacidad de detección. La utilización de técnicas de aprendizaje automático y algoritmos de control inteligente podría brindar una solución eficaz para lograr una adaptación dinámica y precisa en distintos escenarios.

- En futuras investigaciones, sería beneficioso considerar la implementación del sistema ORB-SLAM para llevar el sistema a un nivel más avanzado. ORB-SLAM es un sistema de localización y mapeo simultáneo basado en características visuales, el cual ha demostrado ser efectivo en la detección y seguimiento de objetos dinámicos en entornos complejos. Al integrar ORB-SLAM en el sistema existente, se podrían obtener mejoras significativas en la capacidad de detección y seguimiento de objetos en tiempo real, lo cual es crucial en aplicaciones donde la interacción con objetos dinámicos es fundamental. La combinación de las capacidades de detección y mapeo del sistema actual con las características de ORB-SLAM permitiría alcanzar un nivel superior de precisión y robustez en la detección de objetos en entornos cambiantes.

- Como contribución fundamental de este trabajo, se han introducido nuevos conjuntos

de datos cuidadosamente preparados y grabados (datasets) que se encuentran disponibles para su utilización, ya sea en formato físico o virtual, según las necesidades de las investigaciones futuras. Se recomienda enfáticamente que en proyectos venideros se considere la posibilidad de expandir y enriquecer estos datasets mediante la captura de escenarios adicionales, particularmente aquellos de mayor complejidad. Esta ampliación permitirá un entrenamiento más exhaustivo del sistema, lo cual fortalecerá su capacidad para abordar desafíos y dificultades más variados y realistas en aplicaciones prácticas, impulsando así el desarrollo continuo de la tecnología en este campo.

Glosario y abreviaturas

Hz: Hertz.

LIDAR: detección por imagen de laser.

Plano XY: Plano sobre los ejes X y Y.

PCL: Point cloud nube de puntos.

Rec-HV: Reconstruction for point high Variability

RGB: red, green, blue.

RGB-D: red, green, blue y depht.

ROS: sistema operativo robótico.

RVIZ: visualización de ROS.

SLAM: Localización y Mapeo Simultaneo

V-SLAM: Visual SLAM.

SLAM-Dinámico: SLAM en ambientes dinámicos.

Pointcloud_to_laserscan: conversión nube de puntos 3D a escaneo laser 2D

2D: dos dimensiones.

3D: tres dimensiones.

Glosario de ecuaciones.

α1 y α2: Ángulos correspondientes a las dr_{max1} , dr_{max2} .

β: Ángulo de rotación sobre el eje y del k-ésimo ciclo.

dr: Desviación estándar.

$\boldsymbol{dr_{max1}}$, $\boldsymbol{dr_{max2}}$: Desviaciones estándar máximas.

$\boldsymbol{dr_k}$: Vector de desviación estándar para el kk- ésimo ciclo.

i: posición en el vector r

i1 y i2: posiciones de los dr_{max1} , dr_{max2} en el vector dr_k.

k: número de lectura del ciclo del LIDAR.

l: longitud del vector.

$\boldsymbol{Mt_k}$: Matrices de trasformación homogénea del k-esimo ciclo.

$\boldsymbol{Mt_{k-1}}$: Matrices de trasformación homogénea para el k-ésimo -1 ciclo

n: número de ciclos.$\boldsymbol{O_0}$: punto de origen de marco de referencia 0.

$\boldsymbol{O_k}$: punto de origen de marco de referencia del k-ésimo ciclo.

$\boldsymbol{O_{k-1}}$: punto de origen de marco de referencia del k-ésimo-1 ciclo.

p: número de puntos de la PCL del k-esimo ciclo.

$\boldsymbol{PCL_{Est-k}}$: nube de puntos de la zona estática.

$\boldsymbol{PCL_{Din}}$: nube de puntos del objeto dinámico.

$\boldsymbol{PCL_k}$: nube de puntos reducida después de aplicar la traslación y rotación.

$\boldsymbol{PCL_{kt}}$: PCL de reconstrucción del k-esimo ciclo.

$\boldsymbol{PCL_{kt-1}}$: PCL de reconstrucción del k-esimo-1 ciclo.

$\boldsymbol{PCL_{Oek}}$: nube de puntos de la zona estática extraída de la zona determinada como dinámica en la segmentación por ángulo.

$\boldsymbol{PCL_{Reducida}}$: Nube de puntos reducida.

$\boldsymbol{PCL_{RGB-D}}$: nube de puntos de la cámara RGB-D

$\boldsymbol{PCL_{ref}}$: PCL de escena de referencia.

$\boldsymbol{PCL_{Trans}}$: PCL transitoria.

$\boldsymbol{PCL_{ZD}}$: nube de puntos de la zona dinámica.

$\boldsymbol{PCL_{zek}}$: nube de puntos de la zona estática obtenida de la primera segmentación.

$\boldsymbol{Pt_{KT}}$: Imagen de 2 dimensiones de la proyección de la $\boldsymbol{PCL_{kt}}$.

$\boldsymbol{Pt_{ref}}$: Imagen de 2 dimensiones de la proyección de la $\boldsymbol{PCL_{kt}}$.

q: número de puntos de la PCL del k-esimo-1 ciclo.

r: profundidad.

r: vector de profundidad.

$\boldsymbol{r_k}$: vector profundidad del k-esimo ciclo

$\overline{\boldsymbol{r_k}}$: vector de promedios de profundidad para el k-ésimo ciclo

$\boldsymbol{r_{pk}}$: radio de un punto de la PCL del k-esimo ciclo.

$\boldsymbol{t_{vg}}$: tamaño del voxel grid.

$\boldsymbol{vt_{kx}}$: traslación de la coordenada x del k-ésimo ciclo.

$\boldsymbol{V_{rp}}$: vector con las coordenadas de cadapunto de la PCL_r.

$\boldsymbol{vt_{ky}}$: traslación de la coordenada y del k-ésimo ciclo.

$\boldsymbol{vt_{kz}}$: traslación de la coordenada z del k-ésimo ciclo.

$\boldsymbol{vt_k}$: Vector traslación de parámetros de traslación del k-ésimo ciclo.

$\boldsymbol{x_{max}}$: Máximo de profundidad para la

segmentación por agrupación

$\boldsymbol{X_{pk-1}}$, $\boldsymbol{Y_{pk-1}}$, $\boldsymbol{Z_{pk-1}}$: coordenadas x, y, z de un punto de PCL_{Est} del k-ésimo-1 ciclo.

$\boldsymbol{x}$: eje x.

$\boldsymbol{y}$: eje y.

z: eje z.

$\boldsymbol{\alpha_{pt}}$: Ángulo de un punto de la $PCL_{Reducida}$con respecto al plano XY

θ: ángulo de una medida de profundidad del LIDAR virtual

$\boldsymbol{\theta_{min}}$: ángulo de inicio de muestreo delLIDAR virtual.

$\boldsymbol{\theta_k}$: vector de ángulos del k-esimo ciclo.

Δθ: ángulo de incremento.

$\boldsymbol{\varphi}$: ángulo de rotación sobre el eje x del k-ésimo ciclo.

γ: ángulo de rotación sobre el eje z del k-ésimo ciclo.

$\boldsymbol{X_{pk}}$, $\boldsymbol{Y_{pk}}$, $\boldsymbol{Z_{pk}}$: coordenadas x, y, z de un punto de PCL_{Est} del k-ésimo ciclo.

Anexo 1. Error de re proyección

Durante el proceso de calibración utilizando imágenes patrón, se observó que el sistema RGB-D presentaba un error de re proyección ($e_{rep} = 0.114397$) medido en términos de un pixel.

Dado que el RGB-D tiene una apertura de 57° y un total de 640 píxeles en el eje y, se puede calcular la relación de 0.089° por píxel.

Utilizando (12), se puede determinar el error de re proyección para un grado de apertura del RGB-D ($e_{\propto}$).

$$e_{\alpha} = e_{rep} \frac{0.089°}{1px}. \tag{12}$$

$$e_{\alpha} = 0.01018135°$$

Con este valor de e_{α} se puede calcular el error de re proyección bajo una de las condiciones de las pruebas realizadas, ver Figura 36.Figura 36. Error de reproyección.

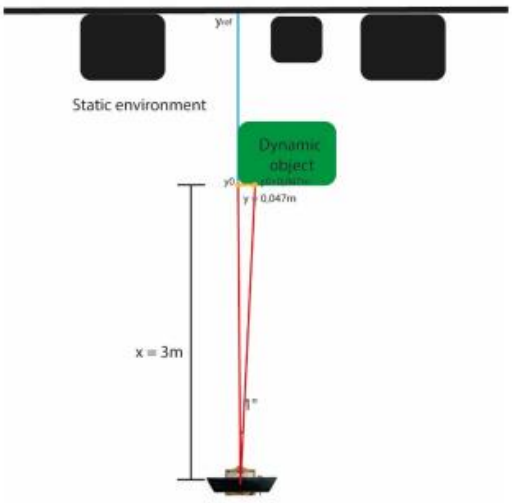


Figura 36. Error de reproyección.

La Ecuación (13) establece que por cada grado en el eje y, se cubre un fragmento del objeto

dinámico cuando este realiza un recorrido a una distancia de 3 metros respecto al sensor en el eje x.

$$y \approx \tan 1° \ . \ 3m = 0.047m \tag{13}$$

La ecuación (14) establece que el error de reproyección en el eje y (e_y) se obtiene a partir del corrimiento en el segmento captado ($e_\propto$).

$$\begin{aligned} e_y = \ & \tan 0.01018135° \ . \ 3m = 0.004797m \\ & y0 \approx \ y_{ref} \pm 0.004797m \end{aligned} \tag{14}$$

Donde y_0 es la medida captada por el sensor que sufre una alteración en referencia a y_{ref} que es la medida del entorno real.

Anexo 2. Matrices de calibración.

Mono pinhole calibration.

$D = [0.13826258934353136 \ -0.2799756919968165 \ 0.0020722559856376937 \ 0.008884031820898908 \ 0.0]$

$K = [522.0276786799784 \ 0.0 \ 305.8050534416858 \ 0.0 \ 518.1331643401821 \ 248.56968926367864 \ 0.0 \ 0.0 \ 1.0]$

$R = [1.0 \ 0.0 \ 0.0 \ 0.0 \ 1.0 \ 0.0 \ 0.0 \ 0.0 \ 1.0]$

$P = [528.4050903320312, 0.0, 309.9429018153569, 0.0, 0.0, 528.0941772460938, 248.96161180673516, 0.0, 0.0, 0.0, 1.0, 0.0]$

Image

width = 640

height = 480

Narrow_stereo

$$\text{camera matrix} = \begin{bmatrix} 522.027679 & 0.000000 & 305.805053 \\ 0.000000 & 518.133164 & 248.569689 \\ 0.000000 & 0.000000 & 1.000000 \end{bmatrix}$$

$$\text{distortion} = [0.138263 \quad -0.279976 \quad 0.002072 \quad 0.008884 \quad 0.000000]$$

$$\text{rectification} = \begin{bmatrix} 1.000000 & 0.000000 & 0.000000 \\ 0.000000 & 1.000000 & 0.000000 \\ 0.000000 & 0.000000 & 1.000000 \end{bmatrix}$$

$$\text{camera matrix} = \begin{bmatrix} 528.405090 & 0.000000 & 309.942902 & 0.000000 \\ 0.000000 & 528.094177 & 248.961612 & 0.000000 \\ 0.000000 & 0.000000 & 1.000000 & 0.000000 \end{bmatrix}$$

Anexo 3. Repositorio.

En el repositorio de videos de YouTube, se pueden explorar contenidos que muestran los resultados obtenidos en este proyecto de investigación.

Prueba LIDAR virtual:
https://youtu.be/Jjrfn_PRwGg

Prueba funcionamiento Rec_HV:
https://youtu.be/GyjjQyWB9a4

Prueba correlación:
https://youtu.be/GP8Cbbh8KfA

Github del proyecto:
https://github.com/tjuan45/-Installation_guide_2.0/blob/main/README.md

Referencias

[1] C. -Z. Sun, B. Zhang, J. -K. Wang and C. -S. Zhang, "A Review of Visual SLAM Based on Unmanned Systems," 2021 2nd International Conference on Artificial Intelligence and Education (ICAIE), 2021, pp. 226-234, doi: 10.1109/ICAIE53562.2021.00055.

[2] J. Esparza-Jiménez, M. Devy, and J. Gordillo, "Visual EKF-SLAM from Heterogeneous Landmarks," Sensors, vol. 16, no. 4, p. 489, Apr. 2016, doi: 10.3390/s16040489.

[3] X. Lu, H. Wang, S. Tang, H. Huang, and C. Li, "DM-SLAM: Monocular SLAM in Dynamic Environments," Appl. Sci., vol. 10, no. 12, p. 4252, Jun. 2020, doi: 10.3390/app10124252.

[4] J. P. M. Covolan, A. C. Sementille, and S. R. R. Sanches, "A mapping of visual SLAM algorithms and their applications in augmented reality," Proc. - 2020 22nd Symp. Virtual Augment. Reality, SVR 2020, pp. 20–29, 2020, doi: 10.1109/SVR51698.2020.00019.

[5] J. Zhang, X. Zhao, Z. Chen, and Z. Lu, "A Review of Deep Learning-Based Semantic Segmentation for Point Cloud," IEEE Access, vol. 7. pp. 179118–179133, 2019, doi: 10.1109/ACCESS.2019.2958671.

[6] Y. Kim, H. Lim, and S. C. Ahn, "Multi-body ICP: Motion segmentation of rigid objects on dense point clouds," in 2015 12th International Conference on Ubiquitous Robots and Ambient Intelligence, URAI 2015, 2015, pp. 532–536, doi: 10.1109/URAI.2015.7358823.

[7] C. Yin et al., "Removing dynamic 3D objects from point clouds of a moving RGB-D camera," 2015 IEEE Int. Conf. Inf. Autom. ICIA 2015 - conjunction with 2015 IEEE Int. Conf. Autom. Logist., no. August, pp. 1600–1606, 2015, doi: 10.1109/ICInfA.2015.7279541.

[8] T. Zhang, H. Zhang, Y. Li, Y. Nakamura, and L. Zhang, "FlowFusion: Dynamic Dense RGB-D SLAM Based on Optical Flow," Proc. - IEEE Int. Conf. Robot. Autom., pp. 7322–7328, 2020, doi: 10.1109/ICRA40945.2020.9197349.

[9] C. Sheng, S. Pan, W. Gao, Y. Tan, and T. Zhao, "Dynamic-DSO: Direct Sparse Odometry Using Objects Semantic Information for Dynamic Environments," Appl. Sci., vol. 10, no. 4, p. 1467, Feb. 2020, doi: 10.3390/app10041467.

[10] W. Xie, P. X. Liu and M. Zheng, "Moving Object Segmentation and Detection for Robust

RGBD-SLAM in Dynamic Environments," in IEEE Transactions on Instrumentation and Measurement, vol. 70, pp. 1-8, 2021, Art no. 5001008, doi: 10.1109/TIM.2020.3026803.

[11] I. Salazar, S. Pertuz, and F. Martínez, "Multi-modal RGB-D Image Segmentation from Appearance and Geometric Depth Maps," TecnoLógicas, vol. 23, no. 48, pp. 143–161, 2020, doi: 10.22430/22565337.1538.

[12] F. Nie, W. Zhang, Z. Yao, Y. Shi, F. Li, and Q. Huang, "LCPF: A Particle Filter Lidar SLAM System With Loop Detection and Correction," IEEE Access, vol. 8, pp. 20401–20412, 2020, doi: 10.1109/ACCESS.2020.2968353.

[13] R. Scona, M. Jaimez, Y. R. Petillot, M. Fallon, and D. Cremers, "StaticFusion: Background Reconstruction for Dense RGB-D SLAM in Dynamic Environments," in 2018 IEEE International Conference on Robotics and Automation (ICRA), May 2018, pp. 3849–3856, doi: 10.1109/ICRA.2018.8460681.

[14] Markham et al. (2018). Visual SLAM and Structure from Motion in Dynamic Environments: A Survey. En: ACM Computing Surveys, 51(37).

[15] J. Liu, M. Zhang, S. Zhang, J. Wang and X. Guo, "Unknown Environment Mapping by LIDAR Based on Improved Pointcloud-to-Laserscan," IEEE Access, vol. 9, pp. 7883-7892, 2021, doi: 10.1109/ACCESS.2021.3051461.

[16] S. Yang, Y. Liu, B. Li and W. Chen, "Real-time object detection based on LIDAR point cloud and improved point cloud to laser scan conversion," Measurement, vol. 183, pp. 74-85, 2021, doi: 10.1016/j.measurement.2021.109906.

[17] L. Chen, Y. Sun, H. Zhang and L. Cheng, "Obstacle detection based on improved pointcloud-to-laserscan and deep learning for autonomous vehicles," Sensors, vol. 19, no. 6, p. 1316, 2019, doi: 10.3390/s19061316.

[18] H. Zhang, S. Zhao, T. Jiang, Y. Sun and J. Wang, "Precise Localization of Autonomous Vehicle Using LIDAR and Pointcloud-to-Laserscan," IEEE Transactions on Intelligent Transportation Systems, vol. 22, no. 5, pp. 3012-3022, May 2021, doi: 10.1109/TITS.2020.3020301.

[19] Y. Cai *et al.*, "An unsupervised segmentation method based on dynamic threshold neural P systems for color images," *Inf. Sci. (Ny).*, vol. 587, pp. 473–484, Mar. 2022, doi:

10.1016/j.ins.2021.12.058.

[20] T. Lei, P. Liu, X. Jia, X. Zhang, H. Meng, and A. K. Nandi, "Automatic Fuzzy Clustering Framework for Image Segmentation," *IEEE Trans. Fuzzy Syst.*, vol. 28, no. 9, pp. 2078–2092, Sep. 2020, doi: 10.1109/TFUZZ.2019.2930030.

[21] X. Jia, T. Lei, X. Du, S. Liu, H. Meng, and A. K. Nandi, "Robust Self-Sparse Fuzzy Clustering for Image Segmentation," *IEEE Access*, vol. 8, pp. 146182–146195, 2020, doi: 10.1109/ACCESS.2020.3015270.

[22] Y. Liu and J. Miura, "RDS-SLAM: Real-Time Dynamic SLAM Using Semantic Segmentation Methods," *IEEE Access*, vol. 9, pp. 23772–23785, 2021, doi: 10.1109/ACCESS.2021.3050617.

[23] R. Zhang, G. Li, T. Wunderlich, and L. Wang, "A survey on deep learning-based precise boundary recovery of semantic segmentation for images and point clouds," *Int. J. Appl. Earth Obs. Geoinf.*, vol. 102, p. 102411, 2021, doi: 10.1016/j.jag.2021.102411.

[24] C. Zong, H. Wang, and ZhiboWan, "An improved 3D point cloud instance segmentation method for overhead catenary height detection," *Comput. Electr. Eng.*, vol. 98, Mar. 2022, doi: 10.1016/j.compeleceng.2022.107685.

[25] Y. Zhang, D. Sidibé, O. Morel, and F. Mériaudeau, "Deep multimodal fusion for semantic image segmentation: A survey," *Image Vis. Comput.*, vol. 105, 2021, doi: 10.1016/j.imavis.2020.104042.

[26] M. Taghizadeh and A. Chalechale, "A comprehensive and systematic review on classical and deep learning based region proposal algorithms," *Expert Syst. Appl.*, vol. 189, no. February 2021, p. 116105, 2022, doi: 10.1016/j.eswa.2021.116105.

[27] X. Chen, Q. An, X. Han, Y. Ban, and L. Li, "Control of distributed segmentation of indoor point cloud via homogenization clustering network," *J. Franklin Inst.*, Dec. 2021, doi: 10.1016/j.jfranklin.2021.12.001.

[28] H. Li, Z. Sun, Y. Wu, and Y. Song, "Semi-supervised point cloud segmentation using self-training with label confidence prediction," *Neurocomputing*, vol. 437, pp. 227–237, May 2021, doi: 10.1016/j.neucom.2021.01.091.

[29] N. Y. Khanday and S. A. Sofi, "Taxonomy, state-of-the-art, challenges and applications

of visual understanding: A review," *Comput. Sci. Rev.*, vol. 40, p. 100374, 2021, doi: 10.1016/j.cosrev.2021.100374.

[30] L. Huan, X. Zheng, and J. Gong, "GeoRec: Geometry-enhanced semantic 3D reconstruction of RGB-D indoor scenes," *ISPRS J. Photogramm. Remote Sens.*, vol. 186, no. March, pp. 301–314, 2022, doi: 10.1016/j.isprsjprs.2022.02.014.

[31] U. Bermejo, A. Almeida, A. Bilbao-Jayo, and G. Azkune, "Embedding-based real-time change point detection with application to activity segmentation in smart home time series data," *Expert Syst. Appl.*, vol. 185, p. 115641, 2021, doi: 10.1016/j.eswa.2021.115641.

[32] B. Vishnyakov et al., "Real-time semantic slam with DCNN-based feature point detection, matching and dense point cloud aggregation," Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci. - ISPRS Arch., vol. 43, no. B2-2021, pp. 399–404, 2021, doi: 10.5194/isprs-archives-XLIII-B2-2021-399-2021.

[33] Y. Xu and U. Stilla, "Toward Building and Civil Infrastructure Reconstruction from Point Clouds: A Review on Data and Key Techniques," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 14, pp. 2857–2885, 2021, doi: 10.1109/JSTARS.2021.3060568.

[34] Q. Wu, H. Yang, M. Wei, O. Remil, B. Wang, and J. Wang, "Automatic 3D reconstruction of electrical substation scene from LIDAR point cloud," *ISPRS J. Photogramm. Remote Sens.*, vol. 143, no. April, pp. 57–71, 2018, doi: 10.1016/j.isprsjprs.2018.04.024.

[35] J. Cho, S. Kang, and K. Kim, "Real-time precise object segmentation using a pixel-wise coarse-fine method with deep learning for automated manufacturing," *J. Manuf. Syst.*, vol. 62, no. November 2021, pp. 114–123, 2022, doi: 10.1016/j.jmsy.2021.11.004.

[36] Q. Li, X. Wang, T. Wu, and H. Yang, "Point-line feature fusion based field real-time RGB-D SLAM," *Comput. Graph.*, vol. 107, pp. 10–19, Oct. 2022, doi: 10.1016/j.cag.2022.06.013.

[37] H. Wan, Z. Fan, X. Yu, M. Kang, P. Wang, and X. Zeng, "A real-time branch detection and reconstruction mechanism for harvesting robot via convolutional neural network and image segmentation," *Comput. Electron. Agric.*, vol. 192, p. 106609, Jan. 2022, doi: 10.1016/j.compag.2021.106609.

[38] B. Vishnyakov et al., "Real-time semantic slam with DCNN-based feature point

detection, matching and dense point cloud aggregation," Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci. - ISPRS Arch., vol. 43, no. B2-2021, pp. 399–404, 2021, doi: 10.5194/isprs-archives-XLIII-B2-2021-399-2021.

[39] C. Li and X. Guo, "Topology-Change-Aware Volumetric Fusion for Dynamic Scene Reconstruction," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 12361 LNCS, pp. 258–274, 2020, doi: 10.1007/978-3-030-58517-4_16.

[40] C. Wang et al., "DymSLAM: 4D Dynamic Scene Reconstruction Based on Geometrical Motion Segmentation," IEEE Robot. Autom. Lett., vol. 6, no. 2, pp. 550–557, 2021, doi: 10.1109/LRA.2020.3045647.

[41] A. Mustafa, M. Volino, H. Kim, J. Y. Guillemaut, and A. Hilton, "Temporally Coherent General Dynamic Scene Reconstruction," Int. J. Comput. Vis., vol. 129, no. 1, pp. 123–141, 2021, doi: 10.1007/s11263-020-01367-2.

[42] S. Sun, D. Xu, H. Wu, H. Ying, and Y. Mou, "Multi-view stereo for large-scale scene reconstruction with MRF-based depth inference," *Comput. Graph.*, vol. 106, pp. 248–258, 2022, doi: 10.1016/j.cag.2022.06.009.

[43] C. Theodorou, V. Velisavljevic, V. Dyo y F. Nonyelu, "Visual SLAM algorithms and their application for AR, mapping, localization and wayfinding", *Array*, vol. 15, p. 100222, septiembre de 2022. Accedido el 5 de marzo de 2023. [En línea]. Disponible: https://doi.org/10.1016/j.array.2022.100222

[44] S. Li, D. Zhang, Y. Xian, B. Li, T. Zhang y C. Zhong, "Overview of deep learning application on visual SLAM", *Displays*, p. 102298, septiembre de 2022. Accedido el 5 de marzo de 2023. [En línea]. Disponible: https://doi.org/10.1016/j.displa.2022.102298

[45] X. Ye, S. Chen y R. Xu, "DPNet: Detail-preserving network for high quality monocular depth estimation", *Pattern Recognition*, vol. 109, p. 107578, enero de 2021. Accedido el 5 de marzo de 2023. [En línea]. Disponible: https://doi.org/10.1016/j.patcog.2020.107578

[46] Y. Gryaditskaya, M. Sypesteyn, J. W. Hoftijzer, S. Pont, F. Durand y A. Bousseau, "OpenSketch", *ACM Transactions on Graphics*, vol. 38, n.º 6, pp. 1–16, noviembre de 2019. Accedido el 5 de marzo de 2023. [En línea]. Disponible: https://doi.org/10.1145/3355089.3356533

[47] A. Yu *et al.*, "Attention aware cost volume pyramid based multi-view stereo network for 3D reconstruction", *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 175, pp. 448–460, mayo de 2021. Accedido el 5 de marzo de 2023. [En línea]. Disponible: https://doi.org/10.1016/j.isprsjprs.2021.03.010

[48] Pradeep, A. Mahajan, V. Bharti, H. P. Singh, L. Josyula y P. Kumar, "Construction of a 3D map of indoor environment", *Procedia Computer Science*, vol. 125, pp. 124–131, 2018. Accedido el 5 de marzo de 2023. [En línea]. Disponible: https://doi.org/10.1016/j.procs.2017.12.018

[49] C. A. Velásquez Hernández y F. A. Prieto Ortiz, "A real-time map merging strategy for robust collaborative reconstruction of unknown environments", *Expert Systems With Applications*, vol. 145, p. 113109, mayo de 2020. Accedido el 5 de marzo de 2023. [En línea]. Disponible: https://doi.org/10.1016/j.eswa.2019.113109

[50] D. Reiser, M. Vázquez-Arellano, D. S. Paraforos, M. Garrido-Izard y H. W. Griepentrog, "Iterative individual plant clustering in maize with assembled 2D LIDAR data", *Computers in Industry*, vol. 99, pp. 42–52, agosto de 2018. Accedido el 5 de marzo de 2023. [En línea]. Disponible: https://doi.org/10.1016/j.compind.2018.03.023

[51] M. J. Gallant y J. A. Marshall, "Automated rapid mapping of joint orientations with mobile LIDAR", *International Journal of Rock Mechanics and Mining Sciences*, vol. 90, pp. 1–14, diciembre de 2016. Accedido el 5 de marzo de 2023. [En línea]. Disponible: https://doi.org/10.1016/j.ijrmms.2016.09.014

[52] Lin, X., Quan, Z., Wang, Z. J., Ma, T., & Zeng, X. . "KGNN: Knowledge Graph Neural Network for Drug-Drug Interaction Prediction". In *IJCAI* (Vol. 380, pp. 2739-2745) (2020, July).

[53] M. I. Chacon-Murguia, A. Guzman-Pando, G. Ramirez-Alonso y J. A. Ramirez-Quintana, "A novel instrument to compare dynamic object detection algorithms", *Image and Vision Computing*, vol. 88, pp. 19–28, agosto de 2019. Accedido el 5 de marzo de 2023. [En línea]. Disponible: https://doi.org/10.1016/j.imavis.2019.04.006

[54] C. Sahin, G. Garcia-Hernando, J. Sock y T.-K. Kim, "A review on object pose recovery: From 3D bounding box detectors to full 6D pose estimators", *Image and Vision Computing*, vol. 96, p. 103898, abril de 2020. Accedido el 5 de marzo de 2023. [En línea]. Disponible: https://doi.org/10.1016/j.imavis.2020.103898.

[55] T. Huynh-The, C.-H. Hua, N. A. Tu y D.-S. Kim, "Learning 3D spatiotemporal gait feature by convolutional network for person identification", *Neurocomputing*, vol. 397, pp. 192–202, julio de 2020. Accedido el 5 de marzo de 2023. [En línea]. Disponible: https://doi.org/10.1016/j.neucom.2020.02.048

[56] L. Wang *et al.*, "Drosophila-inspired 3D moving object detection based on point clouds", *Information Sciences*, vol. 534, pp. 154–171, septiembre de 2020. Accedido el 5 de marzo de 2023. [En línea]. Disponible: https://doi.org/10.1016/j.ins.2020.05.006

[57] J. Zhao, H. Xu, H. Liu, J. Wu, Y. Zheng y D. Wu, "Detection and tracking of pedestrians and vehicles using roadside LIDAR sensors", *Transportation Research Part C: Emerging Technologies*, vol. 100, pp. 68–87, marzo de 2019. Accedido el 5 de marzo de 2023. [En línea]. Disponible: https://doi.org/10.1016/j.trc.2019.01.007

[58] N. H. Cordeiro y E. C. Pedrino, "A new methodology applied to dynamic object detection and tracking systems for visually impaired people", *Computers & Electrical Engineering*, vol. 77, pp. 61–71, julio de 2019. Accedido el 5 de marzo de 2023. [En línea]. Disponible: https://doi.org/10.1016/j.compeleceng.2019.05.003

[59] Zhang, Z., Lu, S., & Xu, H. (2019). An improved method of 3D point cloud to 2D laser scan conversion for mobile robot. 2019 Chinese Control And Decision Conference (CCDC), 4474-4479. doi: 10.1109/CCDC.2019.8833046

[60] Zare, A., Tatar, N., & Huang, Y. (2020). Point Cloud to Laser Scan Conversion Using Gaussian Mixture Models. IEEE Robotics and Automation Letters, 5(2), 1493-1500. doi: 10.1109/LRA.2020.2968987

[61] Steiner, M., Kilgus, T., & Vincze, M. (2018). PointCloud to LaserScan Conversion for Autonomous Mobile Robots. 2018 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC), 90-95. doi: 10.1109/ICARSC.2018.8374163

[62] Hashimoto, T., Nakamura, Y., Amano, T., Nishida, Y., & Yuta, S. (2017). 3D Map Construction using LIDAR and Point Cloud to Laser Scan Matching. 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 3588-3593. doi: 10.1109/IROS.2017.8206261

[63] V. Androulakis, J. Sottile, S. Schafrik y Z. Agioutantis, "Navigation system for a semi-autonomous shuttle car in room and pillar coal mines based on 2D LIDAR scanners",

Tunnelling and Underground Space Technology, vol. 117, p. 104149, noviembre de 2021. Accedido el 5 de marzo de 2023. [En línea]. Disponible: https://doi.org/10.1016/j.tust.2021.104149

[64] Z. Zijiang, W. Jingwen, C. Haoran, J. Lin y Y. Shuo, "Mapping method of single LIDAR for indoor degraded environment", *Computers and Electrical Engineering*, vol. 103, p. 108284, octubre de 2022. Accedido el 5 de marzo de 2023. [En línea]. Disponible: https://doi.org/10.1016/j.compeleceng.2022.108284

[65] B. Holý, "Registration of lines in 2D LIDAR scans via functions of angles", *IFAC-PapersOnLine*, vol. 49, n.º 5, pp. 109–114, 2016. Accedido el 5 de marzo de 2023. [En línea]. Disponible: https://doi.org/10.1016/j.ifacol.2016.07.098

[66] P. Kim, J. Chen y Y. K. Cho, "SLAM-driven robotic mapping and registration of 3D point clouds", *Automation in Construction*, vol. 89, pp. 38–48, mayo de 2018. Accedido el 5 de marzo de 2023. [En línea]. Disponible: https://doi.org/10.1016/j.autcon.2018.01.009

[67] H. Liu, X. Tang y S. Shen, "Depth-map completion for large indoor scene reconstruction", *Pattern Recognition*, vol. 99, p. 107-112, marzo de 2020. Accedido el 5 de marzo de 2023. [En línea]. Disponible: https://doi.org/10.1016/j.patcog.2019.107112

[68] C. D. Lombard y C. E. van Daalen, "Stochastic triangular mesh mapping: A terrain mapping technique for autonomous mobile robots", *Robotics and Autonomous Systems*, vol. 127, p. 103449, mayo de 2020. Accedido el 5 de marzo de 2023. [En línea]. Disponible: https://doi.org/10.1016/j.robot.2020.103449

[69] Xiao, J. Wang, X. Qiu, Z. Rong y X. Zou, "Dynamic-SLAM: Semantic monocular visual localization and mapping based on deep learning in dynamic environment", *Robotics and Autonomous Systems*, vol. 117, pp. 1–16, julio de 2019. Accedido el 5 de marzo de 2023. [En línea]. Disponible: https://doi.org/10.1016/j.robot.2019.03.012

[70] B. A. Montenegro, J. F. Flórez, and E. Muñoz, "Dynamic reconstruction in simultaneous localization and mapping based on the segmentation of high variability point zones," Systems Science & Control Engineering, vol. 10, no. 1, pp. 767-776, 2022, DOI: 10.1080/21642583.2022.2123062.