XLIV Jornadas de Automática 2023

https://jautomatica.es

Extracción de modelos 3D basado en CNN y nubes de puntos para mapeado

Mendez, A., Mora, A., Barber, R.

RoboticsLab, Departamento de Ingeniería de Sistemas y Automática, Universidad Carlos III de Madrid, Av. de la Universidad, 30, 28911 Leganés, España.

To cite this article: Mendez, A., Mora, A., Barber, R. 2023. 3D Model Extraction based on CNN and Point Clouds for Mapping. XLIV Jornadas de Automática, 655-660. https://doi.org/10.17979/spudc.9788497498609.655

Resumen

La obtención de modelos 3D precisos es importante en robótica. Una amplia gama de aplicaciones como mapear objetos requieren de un conocimiento preciso de su forma y disposición. Sin embargo, puede ser difícil lograr una alta precisión, especialmente cuando se trata de datos reales que incluyen factores como oclusiones, desorden y ruido, influyendo en el resultado final. En el estado del arte, se combina deep learning en imágenes 2D con segmentación de nubes de puntos. Sin embargo, los estudios comparativos al respecto son muy limitados. En este trabajo, implementamos y comparamos varios métodos existentes, aplicando mejoras para que éstos funcionen correctamente en entornos complejos. Analizamos cuatro formas de obtener modelos 3D, tres de ellos basados en detección por bounding box y un cuarto basado en la obtención de una máscara mediante instance segmentation. Se realiza una comparación tanto cualitativa como cuantitativa, presentando las fortalezas y los límites de cada método, además de un ejemplo práctico de uso.

Palabras clave: Percepción y sensorización; construcción de mapas; navegación de robots, programación y visión; robots móviles; robótica inteligente.

3D model extraction based on CNN and point clouds for mapping.

Abstract

Obtaining accurate 3D models is important in robotics. A wide range of applications such as mapping objects require precise knowledge of their shape and layout. However, it can be difficult to achieve high accuracy, especially when it comes to real data that includes factors such as occlusions, clutter and noise, influencing the final result. In the state of the art, deep learning on 2D images is combined with 3D point cloud segmentation. However, comparative studies in this area are very limited. In this paper, we implement and compare several existing methods, applying improvements to make them work properly in complex environments. We analyze four ways to obtain 3D models, three of them based on bounding box detection and a fourth one based on obtaining a mask by instance segmentation. A qualitative and quantitative comparison is made, presenting the advantages and disadvantages of each method, as well as a practical example of use.

Keywords: Perception and sensing; Map building; Robot Navigation, Programming and Vision; Mobile robots; Intelligent robotics.

1. Introducción

Un aspecto esencial para la comprensión del entorno de un robot es la extracción de modelos 3D, ya que permiten comprender la geometría y la disposición espacial de los objetos para tareas como la creación de mapas semánticos (Qi et al., 2020). Otro ejemplo es la estimación del agarre de 6-GdL a partir de una vista parcial del objeto para una pinza (Alliegro et al., 2022). En ambos casos, un error en la extracción del modelo tiene un efecto directo en los resultados. Esta extracción precisa a partir de datos del mundo real resulta ser una tarea difícil,

*Autor para correspondencia: albmende@pa.uc3m.es

especialmente en presencia de factores como oclusiones. En este trabajo, exploramos el potencial de combinar la detección de objetos mediante CNNs (Convolutional Neural Networks) con la segmentación de nubes de puntos para extraer modelos de objetos 3D precisos. El objetivo es evaluar el rendimiento de técnicas existentes y estudiar su aplicabilidad a la robótica.

Aunque algunos métodos como (Chen et al., 2021) detectan directamente objetos 3D en nubes de puntos, son costosos y requieren grandes cantidades de datos para su entrenamiento. Además, dado que su uso no está tan extendido, no hay muchos modelos entrenados disponibles. Por ello, su uso es más común en aplicaciones como la conducción autónoma, ya que los sensores LiDAR tienen un mayor rango de visión que las cámaras. Sin embargo, para aplicaciones robóticas, la combinación de la detección de objetos en imágenes y la segmentación de nubes de puntos ofrece grandes ventajas.

Existen varias metodologías para detectar objetos en imágenes. En este artículo estudiamos las dos principales: detección por bounding boxes y segmentación por instancias. Tras detectar el objeto, su figura 3D se segmenta en la nube de puntos. Para ello, se utiliza la proyección de la nube sobre el plano de la imagen para determinar qué puntos 3D corresponden al objeto. Dependiendo de la técnica de detección, se obtendrá una nube de puntos local diferente tras la proyección, por lo que serán necesarias diferentes técnicas de filtrado. En el caso de la bounding box, será necesario eliminar los puntos de zonas del fondo u oclusiones, ya que la caja no se ajusta a la forma exacta del objeto. En el caso de segmentación por instancias, se prestará especial atención a los puntos situados en el borde de la máscara, ya que pueden estar fuera de los objetos. En este trabajo se evalúan y comparan estos enfoques para determinar su eficacia en aplicaciones reales. Además, se muestra un ejemplo de uso en el que la posición geométrica de cada objeto es guardada en un mapa de ocupación.

2. Estado del arte

En esta sección, revisamos varias técnicas de la literatura para extraer modelos en 3D. Estos trabajos se focalizan, sobre todo, en aplicaciones de mapeado. Nuestro objetivo es recopilar sus principales características para posteriormente comprobar su funcionamiento.

La extracción de modelos 3D para mapear permite, por ejemplo, crear mapas de alto nivel que incluyen información semántica (Wu et al., 2019). Algunos trabajos como (Zhang et al., 2021) determinan la ubicación de los objetos utilizando el centro de la bounding box para obtener la profundidad. Sin embargo, este punto no siempre pertenece al objeto en sí, por lo que un modelo 3D completo podría mejorar esta estimación.

Varios trabajos se basan en bounding boxes para segmentar las nubes. En (Wang et al., 2019b,a), se fusionan nubes obtenidas desde múltiples ángulos de visión. Se recorta la nube de puntos usando la bounding box y luego se segmenta utilizando Locally Convex Connected Patches (LCCP)(Stein et al., 2014), que segmenta la nube en pequeños bloques mediante supervoxels, que posteriormente se agrupan utilizando crecimiento de regiones. En (Bavle et al., 2020), se extraen planos del recorte de la nube para realizar visual SLAM en un UAV. Otros trabajos como (Xu et al., 2021; Liao et al., 2020) proponen eliminar los puntos del suelo de la nube de puntos utilizando RANSAC antes de aplicar un filtrado euclídeo. El principal problema de estos métodos es cómo seleccionar adecuadamente el umbral de distancia, que influirá en gran medida en el rendimiento de la segmentación. Los autores de (Nakajima and Saito, 2018) proponen segmentar la nube de puntos antes de recortarla. Seguidamente, los clusters se proyectan en el plano de la imagen y aquellos con un área mayor al umbral definido dentro de la caja se seleccionan como parte del objeto.

Respecto a la segmentación por instancias, en (Mascaro et al., 2022), la máscara resultante se utiliza para recortar la nube de puntos. No se proporciona información sobre el filtrado de la nube, lo que podría causar errores en la estimación final de la forma del objeto. Los trabajos presentados en (Liu et al., 2019; Lin et al., 2021; Grinvald et al., 2019; Qi et al., 2020) generan mapas semánticos con objetos basados en SLAM. En todos estos casos, se aplica un filtro después de recortar la nube de puntos. En (Liu et al., 2019), se propone un algoritmo de crecimiento de regiones para eliminar los puntos del contorno de la máscara que no pertenecen al objeto. En (Lin et al., 2021), DBSCAN y un análisis de componentes conectados se combinan con el mismo propósito. En (Grinvald et al., 2019), la nube de puntos se segmenta según las normales estimadas. Es necesario analizar el rendimiento de estos filtros, ya que un pequeño error en los bordes de la máscara puede causar grandes errores en la estimación de la forma del objeto.

Los modelos 3D permiten también a los robots agarrar objetos con destreza. Sin esta información, un robot puede tener dificultades para determinar la mejor aproximación para agarrar un objeto. Muchas de las técnicas aplicadas son similares a las descritas anteriormente, aunque existen algunas otras que aportan una nueva visión del problema. Por ejemplo, los autores de (Li et al., 2021) utilizan GrabCut para seleccionar el objeto en primer plano de un recorte de imagen procedente de una detección por bounding box, ajustándose mejor a la forma del objeto y aproximándose a la forma en que funciona la segmentación por instancias.

Esta visión general sobre la extracción de modelos 3D pone de manifiesto que la variedad de aplicaciones se ha traducido en una variedad de algoritmos. Para realizar una comparación práctica de varios enfoques, analizamos el funcionamiento de cuatro métodos: crecimiento de regiones, LCCP, GrabCut y segmentación por instancias.

3. Técnicas de segmentación desarrolladas

De todos los métodos revisados, se han seleccionado cuatro para compararlos entre sí. Todos ellos parten de la obtención de una imagen y una nube de puntos a partir de una cámara RGB-D. Tres de los métodos se basan en la detección de objetos mediante bounding box, mientras que el cuarto se basa en la máscara obtenida de segmentación por instancias. En todos estos casos, es necesario establecer una relación entre la nube de puntos 3D y la imagen 2D. A continuación se explica el procedimiento de detección de objetos en la imagen, así como la proyección de los puntos 3D sobre el plano de la imagen y las técnicas de segmentación aplicadas. El código se ha desarrollado utilizando la librería PCL en C++ (Rusu and Cousins, 2011) y Open3D en Python (Zhou et al., 2018).

3.1. *Detección de objetos*

La detección de objetos se realiza en la imagen 2D después de capturar datos. Para conseguir este objetivo, en este trabajo se utiliza una CNN que trabaja en tiempo real: *YOLOv5* (Jocher et al., 2021). Esta detección se puede realizar de dos formas diferentes: detección por bounding box y segmentación por instancias. Un ejemplo de resultado de ambos métodos se muestra en la Fig. 1. La principal diferencia es que las cajas delimitan los objetos utilizando un rectángulo, lo que puede conducir a una localización imprecisa de los objetos. Mientras tanto, la segmentación de instancias delimita los objetos píxel a píxel, manteniendo las formas de los objetos Hafiz and Bhat (2020). Esto tiene un impacto significativo durante el entrenamiento de la CNN. La segmentación de instancias requiere conjuntos de datos más grandes y más tiempo de entrenamiento. Además, el etiquetado de objetos es más difícil y requiere más tiempo por instancia de objeto. En este trabajo, se pretende probar diferentes algoritmos de extracción de modelos de objetos 3D utilizando los dos métodos de detección de objetos para analizar su impacto en la segmentación 3D y determinar cuándo es recomendable utilizar la detección por bounding box o la segmentación por instancias.

(a) (b)

Figura 1: Métodos de detección de objetos: (a) bounding box, (b) segmentación por instancias.

3.2. *Correspondencia 2D - 3D*

En los cuatro métodos propuestos se requiere una correspondencia entre los píxeles de la imagen 2D y los puntos de la nube de puntos 3D. Más concretamente, es necesario proyectar los puntos 3D en el plano de la imagen para ver si corresponden o no a la región del objeto obtenida a partir del detector de objetos. Para ello, se utiliza el modelo de cámara pinhole. Para establecer la relación entre un punto 3D $P = (X_b, Y_b, Z_b)$ y un punto 2D $p = (X_b, Y_b)$, se utilizan las siguientes ecuaciones:

$$x_b = \frac{f_x \cdot X_b}{Z_b} + c_x, \qquad y_b = \frac{f_y \cdot Y_b}{Z_b} + c_y \tag{1}$$

donde x_b e y_b son las coordenadas del punto en la imagen, X_b, Y_b y Z_b son las coordenadas en la nube de puntos, f_x y f_y son las distancias focales de la cámara en las direcciones x e y respectivamente, y c_x y c_y son las coordenadas del centro de la imagen.

3.3. *Segmentación de la nube de puntos*

En esta sección se explican los cuatro métodos de segmentación seleccionados, tres de ellos basados en el resultado de la detección por bounding box y un cuarto basado en la máscara obtenida de la segmentación por instancias.

3.3.1. *Algoritmo de crecimiento de regiones*

El método basado en crecimiento de regiones se inspira en el trabajo presentado en (Nakajima and Saito, 2018). En este trabajo, la nube se segmenta utilizando un algoritmo incremental y, a continuación, se establece un criterio para ver qué clusters corresponden al objeto comprobando si se encuentran dentro del bounding box del objeto. El primer paso consiste en aplicar el algoritmo de crecimiento de regiones a la nube de puntos global. El resultado es un conjunto de clusters que corresponden a objetos o a partes de objetos. Este último caso se da sobre todo en objetos no convexos, como las sillas, que suelen dividirse en respaldo y asiento. En la Fig. 2 se muestra un ejemplo en el que los parámetros del algoritmo de crecimiento de regiones se han modificado para forzar el incremento de número de clusters, de manera que pueda visualizarse mejor su comportamiento. A continuación, cada grupo se proyecta en el plano de la imagen utilizando pinhole. Para cada uno, se calcula la proporción de puntos dentro del bounding box sobre el número total de puntos en el cluster. Si este valor es superior a 0.9, el cluster se selecciona como parte del objeto. De este modo, se filtran los objetos de fondo y oclusiones.

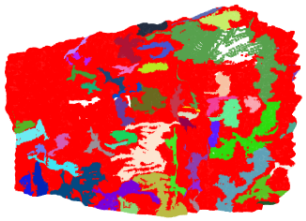


Figura 2: Segmentación de crecimiento de regiones aplicada a una nube de puntos. Cada color indica un cluster diferente. El color rojo indica outliers. En este ejemplo, los parámetros del algoritmo se han forzado para aumentar el número de clusters.

3.3.2. *LCCP*

LCCP se inspira en los trabajos presentados en (Wang et al., 2019b,a). En su investigación, los autores propusieron recortar primero la nube de puntos seleccionando los puntos 3D dentro de la bounding box del objeto. A continuación, aplicaron el algoritmo LCCP a la nube de puntos local. Este algoritmo se basa en dos etapas principales: la división en vóxeles pequeños mediante la segmentación de supervoxels y la fusión de voxels mediante el cálculo de un grafo de adyacencia. Un ejemplo se muestra en la Fig. 3. Para seleccionar el cluster correspondiente al objeto, los autores suponen que éste se encuentra en el centro de la bounding box y que ocupa la mayor parte del área de la caja. Sin embargo, esta suposición no puede realizarse, especialmente para objetos no convexos. LCCP tiende a sobresegmentar objetos como las sillas, ya que presta especial atención a las formas convexas. Por lo tanto, se necesita una estrategia para seleccionar más de un cluster en caso de que el objeto esté dividido en dos o más partes. Por ello, proponemos seleccionar clusters suponiendo que se encuentran en el centro de la nube de puntos (en lugar del centro de la caja) y que son mayores que otros clusters. De este modo, los clusters pequeños correspondientes a oclusiones se eliminan debido a su tamaño y los clusters de fondo grandes también se filtran porque están demasiado lejos del centro.

(a) (b)

Figura 3: Pasos del LCCP: (a) segmentación por supervoxels, (b) unión de supervoxels basada en el grafo de adyacencia.

3.3.3. *GrabCut*

El método basado en GrabCut está inspirado en el trabajo presentado en Li et al. (2021). El objetivo es conseguir una máscara similar a la obtenida con segmentación por instancias pero utilizando una CNN que realiza la detección por bounding box. La propuesta consta de dos partes: extracción del objeto 2D y segmentación de la nube de puntos. La extracción del objeto se realiza utilizando GrabCut (Rother, 2004) de OpenCV. El algoritmo utiliza un Gaussian Mixture Model (GMM) para modelar la distribución de color de los píxeles en una imagen y, mediante un grafo basado en esta distribución, genera la máscara binaria que delimita el objeto. Una vez recibida una imagen de la cámara, se recorta utilizando la bounding box. Es en este recorte donde se aplica GrabCut para que la forma del objeto quede mejor definida. A continuación, al igual que en los métodos anteriores, se seleccionan los puntos de la nube correspondientes a esa región. Dado que la máscara GrabCut no es tan precisa como la de segmentación por instancias y que las oclusiones pueden no haber sido filtradas, se añade un paso adicional para eliminar los datos no deseados aplicando un clustering basado en la densidad de puntos. Estos pasos se muestran en la Fig. 4.

(a) (b) (c)

Figura 4: Pasos del GrabCut: (a) recorte de la imagen usando la bounding box, (b) extracción del objeto, (c) recorte de la nube de puntos.

3.3.4. *Segmentación por instancias*

El método basado en segmentación por instancias es la propuesta más directa. Se aplica de nuevo el modelo de cámara pinhole para ver qué puntos de la nube 3D están dentro de la máscara calculada, que tiene la forma del objeto. Sin embargo, es necesario filtrar el resultado en caso de que la máscara no se ajuste perfectamente a la forma del objeto. Podría ocurrir que el límite de la máscara estuviera fuera del objeto, por lo que se seleccionarían puntos que estuvieran lejos del objeto. Por este motivo, la máscara se erosiona primero utilizando un kernel en forma de disco de radio 3. A continuación, se aplica un filtro de eliminación de outliers para asegurar que sólo se seleccionan los puntos correspondientes al objeto.

4. Evaluación de los algoritmos

Para comprobar la precisión de los métodos propuestos, sus resultados se evalúan con varias métricas. En primer lugar, se capturan datos de un número variado de objetos característicos de entornos de interiores. La captura de datos se ha realizado utilizando una cámara RGB-D RealSense D-435i, que proporciona imágenes y nubes de puntos alineadas. En cuanto al software, se ha elegido ROS como enlace entre el hardware y los algoritmos. Seguidamente, tras aplicar los métodos propuestos, se evalúa la calidad de la segmentación comparando los resultados con datos etiquetados manualmente. A continuación se presentan los resultados, tanto cualitativa como cuantitativamente, además de un ejemplo de uso en el que se realiza un mapa de objetos.

4.1. *Resultados cuantitativos*

Para la evaluación cuantitativa, se han aplicado cuatro métricas diferentes. La primera es el tiempo de ejecución, con el objetivo de comprobar si los métodos son válidos para aplicaciones en tiempo real. El código se ha ejecutado en una CPU Intel(R) Core(TM) i7-12700H de 12ª generación. La segunda métrica es intersection over Union (IoU), que cuantifica el solapamiento entre dos bounding boxes 3D, uno correspondiente a la nube de puntos del ground truth y el otro a la salida del método especificado. La tercera métrica es la distancia Chamfer (CD), otra métrica de similitud calculada como la suma de las distancias entre cada punto de una nube y su vecino más cercano en la otra nube. Por último, también se mide la distancia entre el centro de la nube de puntos real y la estimada.

Los resultados se recogen en la Tabla 1, donde se proporcionan los valores medios así como las desviaciones estándar. El mejor valor para cada métrica se ha marcado en negrita.

Si nos fijamos en el tiempo, se observa que GrabCut (GC) tarda más en ejecutarse que el resto de métodos. Éste es un factor clave, ya que algunas aplicaciones pueden requerir cálculos en tiempo real. Debido a esto, los métodos más apropiados

Tabla 1: Métricas de evaluación para los cuatro métodos propuestos

	Tiempo (s)	IoU	CD (m)	Distancia (m)
RG	0.4415 ± 0.2291	0.4512 ± 0.2243	997.7513 ± 841.9354	0.2085 ± 0.1225
LCCP	0.0383 ± 0.0217	0.4551 ± 0.2591	704.7949 ± 522.7520	0.2250 ± 0.1773
GC	2.3043 ± 1.3372	0.4525 ± 0.2566	600.1952 ± 490.6789	**0.1934 ± 0.1631**
INST	**0.0124 ± 0.0083**	**0.4763 ± 0.2038**	**459.2447 ± 425.6846**	0.2195 ± 0.1367

serían LCCP y la segmentación por instancias (INST), ya que en nuestro caso son los que mejor se aproximan a la frecuencia en la que la cámara captura datos (30 fps). En cuanto a las métricas de precisión, en general, la segmentación por instancias ofrece los mejores resultados. Sólo GC supera a INST en cuanto a la métrica de distancia. Sin embargo, las diferencias en esta métrica entre los cuatro métodos son muy pequeñas, al igual que en IoU. Sólo en CD se observa una diferencia significativa, estableciendo que el método INST es el más adecuado tanto por tiempos como por precisión.

4.2. Resultados cualitativos

Observando el resultado visual obtenido con cada método, podemos detectar las principales limitaciones cualitativas de los métodos (Fig. 5). En la Fig. 5(a) podemos apreciar las dificultades que tiene el crecimiento de regiones (RG) con objetos que están muy cerca del fondo, ya que todo se agrupa en un único cluster. Algo similar ocurre en la Fig. 5(b), en este caso LCCP tiene problemas para separar los distintos elementos de la escena. La Fig. 5(c) muestra el problema para GC, que en algunos casos la distancia de agrupamiento puede fallar debido al umbral seleccionado, por lo que las oclusiones no se filtran. También INST (Fig. 5(d)) comete errores, ya que la máscara no elimina los agujeros de los objetos, por lo que puntos del fondo se agrupan junto con el objeto. Estos hechos afectan directamente a la calidad del mapeado, ya que la ubicación y dimensión estimadas de los objetos serían incorrectas, llegando incluso a producirse conflictos entre varios objetos que podrían solaparse.

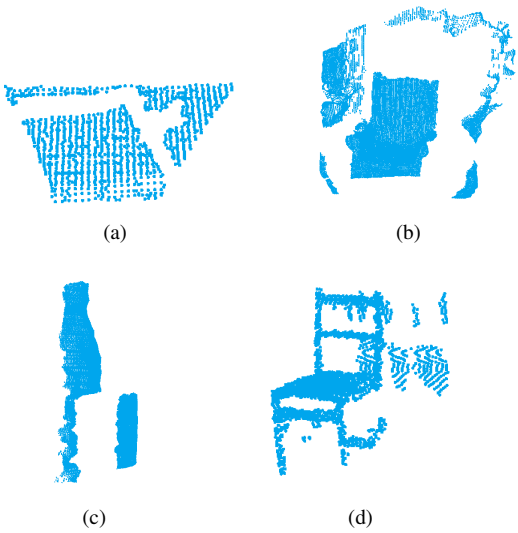


Figura 5: Principales limitaciones de los métodos cuando aparecen situaciones complejas: (a) crecimiento de regiones, (b) LCCP, (c) GrabCut, (d) segmentación por instancias.

4.3. Aplicación práctica: mapa de objetos

Como ejemplo de aplicación práctica, se busca realizar un mapa de objetos en el que se encuentra tanto la información espacial geométrica como la localización de los objetos detectados. Este tipo de mapas pueden utilizarse para aplicaciones como la mostrada en (Wu et al., 2019). Para esta prueba, hemos usado el robot móvil bimanipulador ADAM (Barber et al., 2022), que cuenta con un sensor láser integrado que será utilizado para tomar datos geométricos, y con una cámara RGB-D conectada para tomar datos de imágenes y profundidad. El mapa, correspondiente a un entorno real de oficinas, se ha realizado en dos pasos. Inicialmente, se aplica Gmapping, una técnica de SLAM disponible en ROS, que permite calcular un mapa de ocupación. Seguidamente, una vez el mapa ha sido completado, se utiliza el localizador AMCL, también de ROS, para localizar el robot en el mapa y poder detectar objetos en el eje de referencia global. Esto es mostrado en la Fig. 6, donde se muestra el mapa de ocupación (la zona gris clara es espacio libre y la negra ocupada), la localización estimada del robot (vector rojo) y la información del sensor láser (zonas azules-verdes). Será en este momento donde se capturará la información de los objetos, de manera que se localicen en el mapa geométrico.

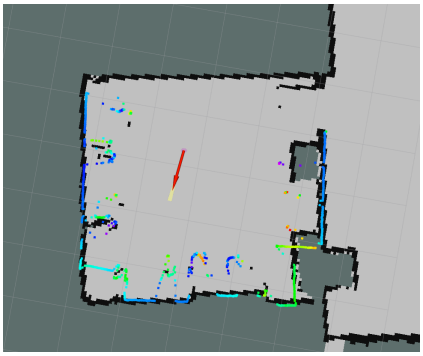
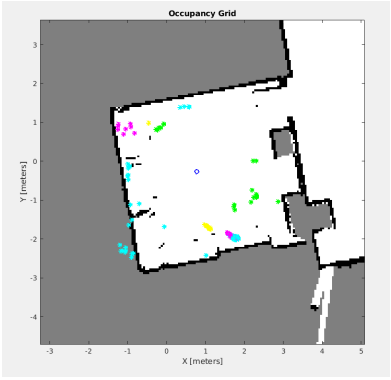


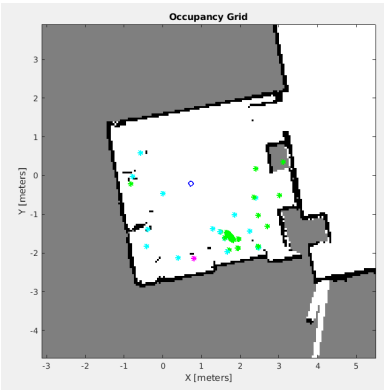
Figura 6: Escenario para tomar datos de objetos: mapa de ocupación, localización del robot (vector rojo) e información láser (tonos azules-verdes).

Para demostrar la necesidad de una detección precisa de los modelos 3D de los objetos, se han llevado a cabo dos pruebas de mapeado, una utilizando la selección del punto medio de la bounding box del objeto detectado y otra eligiendo el punto medio 3D del modelo extraído mediante segmentación por instancias. Se ha elegido este método como representativo de las técnicas desarrolladas en este trabajo ya que, cuantitativamente, ha proporcionado los mejores resultados.

Al introducir la localización de los objetos en el mapa utilizando los modelos 3D de cada uno, se puede observar una mejora al utilizar el método de segmentación por instancias, como se puede apreciar en la figura 7. La figura 7(a) muestra el resultado obtenido al introducir la localización de los objetos utilizando únicamente como referencia el punto medio de la bounding box del objeto detectado. En algunos casos, toma puntos ajenos a la nube de puntos del objeto dado que ésta no se ajusta a la forma del objeto, dando lugar a una mala localización del objeto en el mapa. Esto puede observarse sobre todo en la esquina inferior izquierda del mapa, donde hay objetos que se localizan fuera de la estancia, en zonas sin explorar. Mientras tanto, en la Fig. 7(b) al utilizar la máscara del objeto segmentado para la obtención del punto medio, se toman únicamente los valores de la nube que forman parte del objeto, corrigiendo ese error de localización. De esta forma, los puntos que antes salían del mapa, ahora se encuentran en su posición correcta. Cabe destacar que entre ambos métodos hay discrepancias en el número de objetos detectados. Esto se debe a que son métodos de detección de objetos diferentes.



(a)



(b)

Figura 7: Mapa obtenido con los distintos métodos de extracción de la información 3D de los objetos. a) Punto medio de la bounding box. b) Punto medio de la segmentación por instancias.

5. Conclusiones

En este trabajo se han presentado varios métodos de extracción de modelos 3D de objetos. Según nuestros resultados, la segmentación por instancias proporciona los resultados más rápidos y precisos. Debido a sus limitaciones en cuanto a la elaboración de datasets y el tiempo de entrenamiento, podrían elegirse métodos basados en la detección de bounding boxes. Para aplicaciones en las que el tiempo de ejecución no es relevante, se ha comprobado que el método GrabCut es el más preciso. Sin embargo, en los casos en los que el tiempo es especialmente relevante, como mapeado en tiempo real usando SLAM, LCCP resulta la mejor opción. Adicionalmente, se ha realizado una prueba de mapeado de objetos, demostrando la importancia de mapear los objetos correctamente. En general, puede afirmarse que fusionar la detección de objetos 2D y la segmentación de nubes de puntos es un enfoque prometedor para obtener modelos precisos.

Agradecimientos

Este trabajo ha sido realizado parcialmente gracias al apoyo del proyecto RoboCity2030 DIH-CM (S2018/NMT-4331, RoboCity2030 Madrid Robotics Digital Innovation Hub).

Referencias

Alliegro, A., Rudorfer, M., Frattin, F., Leonardis, A., Tommasi, T., 2022. End-to-end learning to grasp via sampling from object point clouds. IEEE Robotics and Automation Letters 7 (4), 9865–9872.

Barber, R., Ortiz, F. J., Garrido, S., Calatrava-Nicolás, F. M., Mora, A., Prados, A., Vera-Repullo, J. A., Roca-González, J., Méndez, I., Mozos, Ó. M., 2022. A multirobot system in an assisted home environment to support the elderly in their daily lives. Sensors 22 (20), 7983.

Bavle, H., De La Puente, P., How, J. P., Campoy, P., 2020. Vps-slam: Visual planar semantic slam for aerial robotic systems. IEEE Access 8, 60704–60718.

Chen, X., Li, S., Mersch, B., Wiesmann, L., Gall, J., Behley, J., Stachniss, C., 2021. Moving object segmentation in 3d lidar data: A learning-based approach exploiting sequential data. IEEE Robotics and Automation Letters 6 (4), 6529–6536.

Grinvald, M., Furrer, F., Novkovic, T., Chung, J. J., Cadena, C., Siegwart, R., Nieto, J., 2019. Volumetric instance-aware semantic mapping and 3d object discovery. IEEE Robotics and Automation Letters 4 (3), 3037–3044.

Hafiz, A. M., Bhat, G. M., 2020. A survey on instance segmentation: state of the art. International journal of multimedia information retrieval 9 (3), 171–189.

Jocher, G., Stoken, A., Borovec, J., Christopher, S., Laughing, L. C., 2021. ultralytics/yolov5: v4. 0-nn. silu () activations, weights & biases logging, pytorch hub integration. Zenodo.

Li, Z., Xu, B., Wu, D., Zhao, K., Lu, M., Cong, J., 2021. A mobile robotic arm grasping system with autonomous navigation and object detection. In: 2021 International Conference on Control, Automation and Information Sciences (ICCAIS). IEEE, pp. 543–548.

Liao, Z., Wang, W., Qi, X., Zhang, X., 2020. Rgb-d object slam using quadrics for indoor environments. Sensors 20 (18), 5150.

Lin, S., Wang, J., Xu, M., Zhao, H., Chen, Z., 2021. Topology aware object-level semantic mapping towards more robust loop closure. IEEE Robotics and Automation Letters 6 (4), 7041–7048.

Liu, K., Fan, Z., Liu, M., Zhang, S., 2019. Object-aware semantic mapping of indoor scenes using octomap. In: 2019 Chinese Control Conference (CCC). IEEE, pp. 8671–8676.

Mascaro, R., Teixeira, L., Chli, M., 2022. Volumetric instance-level semantic mapping via multi-view 2d-to-3d label diffusion. IEEE Robotics and Automation Letters 7 (2), 3531–3538.

Nakajima, Y., Saito, H., 2018. Efficient object-oriented semantic mapping with object detector. IEEE Access 7, 3206–3213.

Qi, X., Wang, W., Liao, Z., Zhang, X., Yang, D., Wei, R., 2020. Object semantic grid mapping with 2d lidar and rgb-d camera for domestic robot navigation. Applied Sciences 10 (17), 5782.

Rother, C., 2004. Interactive foreground extraction using iterated graph cuts. ACM Trans. Graphics 23 (3), 309–314.

Rusu, R. B., Cousins, S., 2011. 3d is here: Point cloud library (pcl). In: 2011 IEEE international conference on robotics and automation. IEEE, pp. 1–4.

Stein, S. C., Wörgötter, F., Schoeler, M., Papon, J., Kulvicius, T., 2014. Convexity based object partitioning for robot applications. In: 2014 IEEE International Conference on Robotics and Automation (ICRA). IEEE, pp. 3213–3220.

Wang, L., Li, R., Sun, J., Liu, X., Zhao, L., Seah, H. S., Quah, C. K., Tandianus, B., 2019a. Multi-view fusion-based 3d object detection for robot indoor scene perception. Sensors 19 (19), 4092.

Wang, L., Li, R., Sun, J., Zhao, L., Shi, H., Seah, H. S., Tandianus, B., 2019b. Object-aware hybrid map for indoor robot visual semantic navigation. In: 2019 IEEE International Conference on Robotics and Biomimetics (ROBIO). IEEE, pp. 1166–1172.

Wu, P.-T., Yu, C.-A., Chan, S.-H., Chiang, M.-L., Fu, L.-C., 2019. Multi-layer environmental affordance map for robust indoor localization, event detection and social friendly navigation. In: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, pp. 2945–2950.

Xu, X., Zhang, L., Yang, J., Cao, C., Tan, Z., Luo, M., 2021. Object detection based on fusion of sparse point cloud and image information. IEEE Transactions on Instrumentation and Measurement 70, 1–12.

Zhang, Y., Tian, G., Shao, X., Liu, S., Zhang, M., Duan, P., 2021. Building metric-topological map to efficient object search for mobile robot. IEEE Transactions on Industrial Electronics 69 (7), 7076–7087.

Zhou, Q.-Y., Park, J., Koltun, V., 2018. Open3d: A modern library for 3d data processing. arXiv preprint arXiv:1801.09847.