CEA Comité Español de Automática
XLVII JORNADAS DE AUTOMÁTICA CÓRDOBA 2026
UNIVERSIDADE DA CORUÑA

Jornadas de Automática

Sistema robótico de mapeo táctil 3D y generación de relieves para personas con discapacidad visual

Álvaro Fernández[1], Jorge Muñoz[a,*], José M. Sánchez Pena[b,]

[a] *RoboticsLab, Departamento de Ingeniería de Sistemas y Automática, Universidad Carlos III de Madrid, Avenida de la Universidad 30, 28911 Leganés, Madrid, España.*
[b] *Grupo de Displays y Aplicaciones Fotónicas, Departamento de Tecnología Electrónica, Universidad Carlos III de Madrid, Avenida de la Universidad 30, 28911 Leganés, Madrid, España.*

To cite this article: Fernández,Álvaro, Muñoz, Jorge, Sánchez Pena, José M. 2026. 3D tactile mapping robotic system and relief generation for visual impairment. Jornadas de Automática, 47.
https://doi.org/10.17979/ja-cea.2026.47.13877

Resumen

Este artículo presenta el desarrollo de un robot móvil autónomo diseñado para generar mapas 2D de interiores (Occupancy Grids) que serán extruidos para generar maquetas 3D (.STL) que sean imprimibles. El objetivo principal es proporcionar a las personas con discapacidad visual una herramienta de preorientación espacial háptica que les permita comprender la distribución arquitectónica de un entorno antes de transitarlo. El sistema implementa una arquitectura de hardware distribuida donde una NVIDIA Jetson Orin Nano procesa algoritmos de SLAM y visión artificial bajo ROS 2 (Jazzy Jalisco), mientras que un microcontrolador Raspberry Pi Pico gestiona el control de bajo nivel de un sensor LiDAR. Se detalla el diseño mecánico, la implementación de nodos de odometría simulada en C++ con acceso nativo a hardware mediante *libgpiod*, y la creación de una aplicación web offline que permite renderizar el mapa en vivo sin depender de software externo. Finalmente, se describe el algoritmo de extrusión 3D basado en dilatación morfológica (OpenCV) y el método *Marching Cubes*.

Palabras clave: Robótica de asistencia, SLAM, Accesibilidad, ROS 2, Modelado táctil 3D.

3D tactile mapping robotic system and relief generation for visual impairment

Abstract

This paper presents the development of an autonomous mobile robot designed to generate indoor maps (Occupancy Grids) and automatically convert them into three-dimensional models (.STL) with tactile relief. The primary goal is to provide visually impaired individuals with a haptic spatial pre-orientation tool, enabling them to understand the architectural layout of an environment prior to navigation. The system implements a distributed hardware architecture where an NVIDIA Jetson Orin Nano processes SLAM and computer vision algorithms under ROS 2 (Jazzy Jalisco), while a Raspberry Pi Pico microcontroller handles the low-level control of a LiDAR sensor. This paper details the mechanical design, the implementation of simulated odometry nodes in C++ with native hardware access via *libgpiod*, and the creation of an offline web application for live map rendering without external software dependencies. Finally, the 3D extrusion algorithm based on morphological dilation (OpenCV) and the *Marching Cubes* method is described, validating the design under Universal Design principles.

Keywords: Assistive robotics, SLAM, Accessibility, ROS 2, 3D tactile modeling.

*Autor para correspondencia: alvafergo@gmail.com

1. Introducción

La orientación espacial en entornos de interior desconocidos es uno de los mayores desafíos para las personas con discapacidad visual. Mientras que en exteriores el uso de sistemas GPS ha supuesto una revolución para la autonomía, los interiores arquitectónicos (edificios públicos, hospitales, centros educativos) presentan barreras cognitivas significativas al carecer de referencias estructurales globales accesibles de forma no visual (Golledge, 1993).

Hoy en día, existen diversas ayudas tecnológicas para orientarse y desplazarse, aunque presentan limitaciones en este contexto específico:

- **Bastones blancos electrónicos:** Estos dispositivos, como evolución del tradicional bastón blanco, incorporan sensores y conectividad. Dispositivos como WeWALK (WeWALK, 2024) o SmartCane (SmartCane, 2024) integran sensores ultrasónicos para detectar obstáculos aéreos (ramas, carteles) y alertar al usuario mediante vibraciones. Si bien mejoran la seguridad inmediata, no ofrecen una visión global del recinto.

- **Navegación GPS y Apps accesibles:** Aplicaciones como Lazarillo permiten explorar el entorno mediante mensajes de audio y rutas por voz. No obstante, la señal satelital se degrada o desaparece en interiores, imposibilitando una orientación fiable en edificios complejos (Zandbergen, 2009).

- **Balizas Bluetooth:** Para abordar el problema *indoor*, se utilizan redes de *beacons* de corto alcance que estiman la ubicación y brindan instrucciones locales (p. ej., "El ascensor está a 5 metros") (Faragher and Harle, 2015). Sin embargo, requieren la instalación costosa de infraestructura y su información sigue siendo secuencial y puntual, sin generar un mapa cognitivo global del espacio.

Por otro lado, la robótica móvil cuenta con tecnologías maduras para la digitalización de estos espacios. Las técnicas de SLAM (*Simultaneous Localization and Mapping*) más extendidas para generar mapas 2D en interiores son GMapping, un algoritmo clásico basado en filtros de partículas (*Rao-Blackwellized*) robusto en entornos estáticos (Grisetti et al., 2007), y Hector SLAM, que prescinde de la odometría basándose exclusivamente en el láser mediante *scan matching*, ideal para terrenos irregulares (Kohlbrecher et al., 2011). Ambos algoritmos generan un *Occupancy Grid* (Rejilla de Ocupación), un croquis cartográfico muy preciso. Sin embargo, esta representación es puramente visual, lo que supone una barrera de accesibilidad crítica para los usuarios con ceguera.

Para cubrir este vacío tecnológico, este proyecto propone el diseño e implementación de un robot móvil capaz de cartografiar entornos interiores y materializar esos datos en mapas táctiles impresos en 3D. El objetivo es ofrecer una "preorientación háptica": permitir que el usuario explore mediante el tacto la geometría del entorno (muros, accesos, distribución) antes de transitarlo, transformando la navegación reactiva en un conocimiento estructural proactivo.

1.1. Umbrales de detección háptica

Para que esta transferencia de información visual a táctil sea efectiva, es imperativo considerar los umbrales de detección háptica. La sensibilidad táctil humana es menos aguda que la visual (Klatzky and Lederman, 2002), por lo que los diseños deben exagerar el relieve. Estudios demuestran que, aunque en condiciones de laboratorio se detectan diferencias de 0.04-0.08 mm, en aplicaciones prácticas se recomiendan relieves mayores a 0.5 mm, situándose la altura óptima de los símbolos táctiles entre 0.3 y 1.5 mm (Loomis, 1981).

Asimismo, la separación espacial es crítica: un valor estándar es dejar al menos 2.5 mm de distancia mínima entre elementos; por debajo de esto, dos puntos tienden a percibirse como uno solo (Appelle et al., 1997). Estos umbrales justifican la necesidad de aplicar un procesamiento morfológico posterior a los mapas generados por el robot antes de su impresión en 3D.

2. Diseño e Implementación

2.1. Concepto General

El sistema se compone de un robot móvil de bajo coste que se sitúa en el espacio a cartografiar. Se conecta vía WiFi local a un smartphone desde donde se controla mediante una App Web. Al finalizar el mapeo, el robot procesa el mapa 2D, extruye los muros basándose en umbrales de percepción háptica y genera un archivo STL que se envía al teléfono para su posterior impresión en 3D. En la Figura 1 se puede observar una imagen del montaje del prototipo completo.

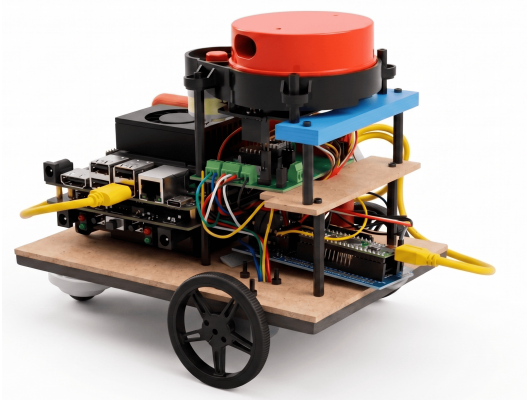

Figura 1: Imagen del montaje completo del robot.

Cabe destacar que la visión a largo plazo para el prototipo final trasciende la fabricación de maquetas físicas estáticas; se contempla la posibilidad de transmitir la información cartográfica directamente a un dispositivo de visualización háptica dinámica capaz de modificar y actualizar sus relieves en tiempo real para el usuario. Aunque esta tecnología de refresco interactivo no se encuentra desarrollada en la presente iteración del sistema, establece el ecosistema conceptual definitivo del proyecto y su viabilidad técnica se describirá en detalle en el apartado de trabajos futuros.

2.2. Hardware

2.2.1. Diseño mecánico y chasis

El chasis combina piezas impresas en 3D (PLA) con una plataforma principal de *foamboard* (cartón pluma reforzado con tablero de partículas) para reducir peso sin perder rigidez.

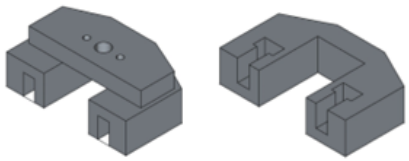

Figura 2: Modelos 3D de los soportes de motor impresos en PLA (Interior y Exterior).

2.2.2. Sensores

El sensor principal es el LiDAR LDS02RR. Proporciona un escaneo 2D de 360 grados, un rango de detección de 0.15 m a 6.0 m, una frecuencia de muestreo de 1800 Hz y una frecuencia de rotación de 5 Hz, logrando una resolución angular de 1 grado. Este es un LiDAR de bajo coste, habitualmente empleado en dispositivos como aspiradoras automáticas. Dado que este sensor únicamente es capaz de capturar una sección bidimensional estática del entorno desde su punto de vista, resulta estrictamente necesario integrarlo sobre una plataforma móvil provista de un sistema de tracción por ruedas. El desplazamiento físico del robot a través de la estancia es el mecanismo indispensable que permite cambiar el punto de referencia del escáner, superar las zonas de oclusión generadas por el mobiliario y alimentar de forma continua al algoritmo SLAM, posibilitando así la construcción de un mapa global y completo del recinto.

2.2.3. Actuadores y Potencia

Se utilizan micromotores metálicos DC Pololu (Pololu Corporation, 2024) con relación de transmisión 75:1 (290 RPM a 6 V) que integran encoders magnéticos de cuadratura. Su eje en ”D”se acopla directamente con ruedas de 60 mm de diámetro. La etapa de potencia se maneja con un driver dual TB6612FNG (Toshiba Corporation, 2008). La alimentación global se extrae de una batería INIU de 10000 mAh.

En la Figura 3 se puede observar cómo encajan las ruedas y los motores con sus soportes en la plataforma principal.

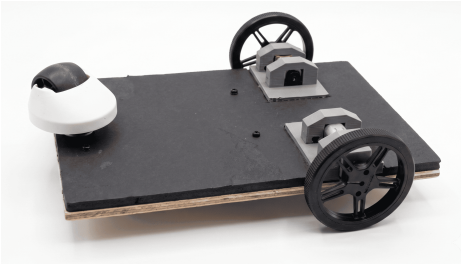

Figura 3: Imagen del chasis junto con las ruedas, los motores y sus soportes.

2.2.4. Arquitectura Computacional Distribuida

El cerebro del sistema es una NVIDIA Jetson Orin Nano (NVIDIA, 2024) (40 TOPS, 1024 núcleos CUDA, CPU ARM de 6 núcleos). Ejecuta redes neuronales, el middleware ROS 2, los algoritmos SLAM y el renderizado 3D.

El control de muy bajo nivel del LiDAR (motor del sensor y captura de bits del LiDAR) se delega a una Raspberry Pi Pico (Raspberry Pi, 2022). Esta actúa como nodo esclavo enviando datos telemétricos limpios por Serial USB a 115200 baudios.

La integración de una Raspberry Pi Pico como microcontrolador intermediario entre el sensor LiDAR y la placa base NVIDIA Jetson Orin Nano responde a dos restricciones de hardware críticas identificadas durante el desarrollo. En primer lugar, la orquestación de un tercer canal de Modulación por Ancho de Pulsos (PWM) en la Jetson, necesario para el control del motor del escáner, presentó problemas a la hora de su configuración, y por la limitación de temporizadores de hardware disponibles, se decidió usar uno de los pines PWM de la Raspberry Pi Pico para el control del motor del LiDAR. En segundo lugar, se detectó que las líneas de transmisión del puerto serie del LiDAR presentaban niveles de voltaje inestables y ruido inducido. Dado que las interfaces UART de la Jetson poseen tolerancias lógicas muy estrictas frente a transitorios eléctricos, una conexión directa generaba corrupción en los datos.

La inclusión del microcontrolador intermediario introduce una latencia de transmisión estimada entre 1 y 3 ms, deducida a partir del tiempo físico requerido para transferir un paquete de datos de 22 bytes (formato de los datos del LiDAR) a una tasa de 115200 baudios (1.9 ms) y el intervalo de muestreo síncrono del bus USB. Los mensajes del LiDAR están configurados para ser enviados por el programa cada 200 ms y la odometría operará con periodos de actualización de 100 ms (intervalo de integración temporal discretizado en $dt = 0{,}1$ s para la cinemática diferencial). En consecuencia, esta latencia a microescala es absorbida sin penalizar el rendimiento del algoritmo de SLAM Toolbox (Macenski, 2024), un conjunto de herramientas para SLAM 2D. Por el contrario, este esquema de aislamiento físico erradica la corrupción de datos por ruido inductivo, garantizando la integridad de la nube de puntos y maximizando la precisión topológica del mapa resultante.

En la Figura 4 se puede observar un diagrama de las conexiones del sistema completo.

2.3. Arquitectura de Software en ROS 2

El sistema se distribuye en tres paquetes lógicos implementados sobre ROS 2 Jazzy Jalisco (Open Source Robotics Foundation, 2024). Un paquete se encarga del control de los motores y de la odometría del robot (`motor_control_pkg`), otro paquete procesa la información del LiDAR (`lidar_pkg`) y un último paquete ejecuta el algoritmo para convertir el mapa de 2D a 3D (`map_generator_pkg`). Estos paquetes se apoyan además en una aplicación web desde la que el usuario controlará el movimiento del robot y el conjunto de herramientas SLAM Toolbox. Esta arquitectura junto con los tópicos mediante los que se comunica se puede observar en el diagrama de la Figura 5.

2.3.1. Control de Motores y Odometría (motor_control_pkg)

Escrito en C++ (`motor_driver_cpp_node.cpp`), este nodo traduce comandos de `/cmd_vel` a las señales eléctricas

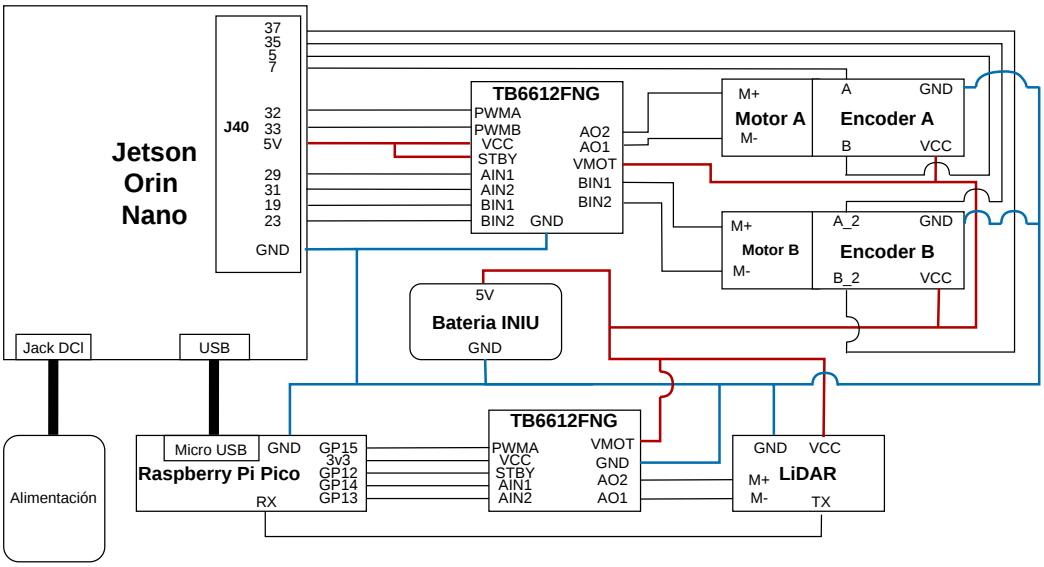

Figura 4: Esquemático del hardware del sistema.

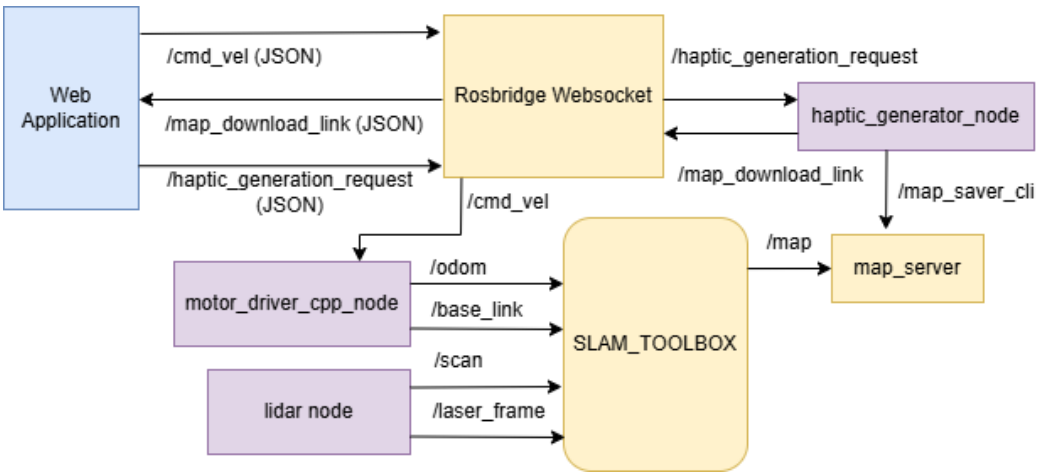

Figura 5: Diagrama de la arquitectura del sistema en ROS 2.

que serán enviadas al driver TB6612FNG para controlar los motores. El sistema opera de forma robusta en lazo cerrado implementando controladores PID. Estos controladores comparan continuamente las velocidades objetivo con la retroalimentación real proporcionada por los encoders magnéticos de cuadratura. El control de los distintos pines de la placa con los voltajes necesarios se hará mediante la librería *libgpiod* (libgpiod project, 2024).

Para la localización del robot y el SLAM, el nodo calcula la odometría matemática empleando la cinemática directa. Para ello, se basa en dos parámetros físicos fundamentales del chasis: el radio de las ruedas (R = 0,03 m) y la distancia entre ambas ruedas o ancho de vía (L = 0,118 m). Utilizando las velocidades reales medidas en cada ciclo, el algoritmo integra la posición espacial (X, Y, θ) con un diferencial temporal de dt = 0,1 s, publicando las coordenadas exactas en el tópico `/odom` y transmitiendo la Transformada (TF) hacia `base_link`.

2.3.2. *Procesamiento LiDAR (lidar_pkg)*

El nodo `lidar_node.py` (Python) gestiona los servicios para el encendido y apagado seguro del motor láser. Una vez encendido el LiDAR, el nodo se encarga de leer el flujo de bytes de la Pico. Identifica la cabecera hexadecimal (`0xFA`), decodifica distancias y ángulos y los almacena en un vector de 360 posiciones. Al detectar un ciclo completo, publica un mensaje `LaserScan` en el tópico `/scan`. Adicionalmente, se encarga de publicar la transformada estática desde `base_link` a `laser_frame`, refiriéndose a la altura a la que está el plano de captura del LiDAR respecto del plano del eje de los motores. En el prototipo desarrollado, esta altura es de Z = 0,15 m.

2.3.3. *SLAM y Aplicación Web de Control*

Se emplea SLAM Toolbox en modo `online_async` para fusionar `/odom` y `/scan`. Para el control manual sin Internet, se aloja una App Web servida localmente. La pasarela `rosbridge_websocket` (RobotWebTools, 2012) (puerto 9090) comunica el frontend HTML/JS con ROS 2. En lu-

gar de RViz, el `/map` (OccupancyGrid) se renderiza en vivo usando una técnica de doble canvas HTML5: un canvas oculto procesa iterativamente los arrays de datos (asignando negro a áreas exploradas y cian a muros), y luego se dibuja escalado en el canvas visible, evitando sobrecargar el procesador del smartphone.

2.3.4. Conversión a Modelo Háptico (map_generator_pkg)

Cuando el operador solicita la transformación tridimensional del entorno desde la interfaz de control, el nodo `haptic_generator_node.py` entra en ejecución estructurando el proceso de traducción háptica en las siguientes etapas secuenciales:

1. **Captura del mapa 2D desde la aplicación web:** Al recibir el estímulo de activación a través de un tópico dedicado, el nodo ejecuta de forma síncrona un subproceso del sistema operativo para invocar la utilidad `map_saver_cli` de la pila de navegación. Esto genera una captura de la rejilla de ocupación (*Occupancy Grid*) actual (.pgm) y un archivo con la configuración usada (.yaml) etiquetados con un prefijo temporal único.
2. **Filtrado del ruido y extracción de contornos:** La imagen en escala de grises es cargada en memoria y procesada mediante la librería OpenCV (OpenCV, 2026). Se aplica una umbralización binaria inversa para discriminar con nitidez los obstáculos del espacio libre. Posteriormente, se ejecuta un algoritmo de extracción de contornos donde se descartan por software aquellas áreas geométricas menores a un umbral crítico de píxeles; este filtrado selectivo erradica el ruido inductivo y las imperfecciones puntuales del escaneo láser, reteniendo exclusivamente los componentes estructurales válidos del entorno.
3. **Optimización morfológica del espesor háptico:** Para satisfacer de manera estricta los umbrales de percepción analizados en la introducción, la máscara binaria de muros aislados se somete a una operación de dilatación morfológica utilizando un elemento estructurante simétrico (kernel plano de 5×5). Este engrosamiento por computación digital garantiza que cualquier tabique o contorno estructural extraído presente un espesor físico mínimo que supere el umbral de discriminación táctil de las yemas de los dedos (Appelle et al., 1997), evitando que las paredes se impriman con un grosor frágil o imperceptible.
4. **Discretización volumétrica y extrusión sobre base sólida:** La imagen bidimensional procesada se proyecta hacia un espacio euclídeo tridimensional mediante la instanciación de un tensor de vóxeles booleanos de profundidad definida (una matriz de 35 capas discretas en el eje Z). El algoritmo inicializa de forma estricta las primeras capas de este volumen (capas 0 a 15) en estado lógico verdadero, lo que materializa automáticamente una base sólida y rígida de geometría rectangular coincidente con los límites del plano. Sobre esta plataforma de soporte común, las capas superiores (15 a 35) replican secuencialmente el perfil de los muros dilatados, delimitando geométricamente la altura del relieve háptico de la habitación.
5. **Generación de la malla geométrica mediante *Marching Cubes*:** Una vez construido el volumen discretizado, se aplica el algoritmo de cubos de marcha (Kastner and Melani, 2000) provisto por la librería de procesamiento científico `skimage.measure` (scikit-image developers, 2026) utilizando un valor de corte de la isosuperficie de 0.5. Este método numérico calcula de forma exacta la frontera del tensor booleano, triangulando el volumen sólido para extraer el conjunto indexado de vértices, vectores normales y caras poligonales que delimitan y estructuran la superficie de la maqueta en la memoria del sistema.
6. **Escalado métrico, exportación y notificación de recurso:** Con el fin de adaptar las coordenadas nativas de la matriz a una escala ergonómica y manejable para el usuario final, la malla de vectores resultantes se somete a una transformación lineal de escalado algebraico. Finalmente, la estructura se almacena físicamente como un archivo de geometría binaria estándar de tipo STL en el almacenamiento de la placa base, y el nodo publica la URL de descarga local a través de una cadena de texto en el tópico `/map_download_link`, notificando al instante al frontend de la aplicación móvil para su disponibilidad inmediata.

3. Conclusiones y Trabajo Futuro

Se ha completado con éxito el desarrollo de un robot móvil capaz de generar modelos táctiles 3D de bajo coste y procesados directamente a bordo gracias a la potencia de la Jetson Orin Nano. Actualmente, el sistema se encuentra en un proceso activo de pruebas y validación experimental en escenarios reales estructurados, con el fin de evaluar la precisión métrica de los planos obtenidos y la fidelidad de las mallas tridimensionales resultantes antes de su despliegue final.

Sin embargo, el sistema presenta limitaciones técnicas en su estado actual. El sensor LiDAR LDS02RR mapea en un único plano 2D, ignorando obstáculos en voladizo (estanterías altas) o desniveles negativos (escalones), lo que puede comprometer la seguridad estructural del modelo impreso si no es supervisado. Además, al generar mapas estáticos, cualquier reconfiguración del mobiliario obliga a remapear el recinto.

Para paliar estas limitaciones y escalar el proyecto, se proponen cinco líneas de trabajo futuro:

1. **Exploración y Navegación Autónoma:** Desarrollar e implementar un algoritmo de exploración basada en fronteras (*Frontier Exploration*) en la pila de navegación del robot. Esto permitirá que la plataforma recorra y cartografíe las estancias de manera completamente independiente y eficiente, automatizando la recolección de datos espaciales y eliminando la necesidad de teleoperación mediante el joystick virtual.
2. **Control Accesible:** Integrar comandos de voz en la App Web para que operarios con baja visión puedan manejar el robot de forma autónoma en caso de requerir supervisión o guiado manual puntual.
3. **Percepción Volumétrica:** Sustituir el LiDAR 2D por cámaras RGB-D (*Depth*) que permitan capturar una nube de puntos tridimensional completa, solucionando por

completo los puntos ciegos en voladizo y los riesgos de desnivel.

4. **Inteligencia Artificial Semántica:** Aprovechar los *Tensor Cores* de la Jetson para ejecutar algoritmos de *Deep Learning* que clasifiquen automáticamente la naturaleza de los obstáculos detectados (p. ej., puertas, mesas, extintores) para poder asignarles texturas o patrones táctiles normalizados y diferenciados en el archivo STL final.
5. **Visualización Háptica Dinámica en Tiempo Real:** Trascender la fabricación de maquetas físicas estáticas mediante el diseño de una interfaz capaz de recibir la información cartográfica y transmitirla directamente a un dispositivo de visualización háptica dinámica (como una matriz activa de pines electromecánicos). Aunque esta tecnología de refresco interactivo no se encuentra implementada en la presente iteración, constituirá el ecosistema conceptual definitivo del proyecto al permitir modificar y actualizar los relieves en tiempo real para el usuario a medida que el robot explora el entorno.

Agradecimientos

Este trabajo ha sido realizado para fomentar la investigación en tecnologías de apoyo, con la visión de que edificios públicos y privados puedan ofrecer asistencia háptica automatizada y de bajo coste a sus usuarios. Estos resultados de investigación se enmarcan dentro del Proyecto del programa de Ayudas para la Actividad Investigadora de los Jóvenes Doctores del Programa Propio de Investigación de la UC3M, con referencia 2025/00764/001, concedido por la Universidad Carlos III de Madrid.

Referencias

Appelle, S., et al., 1997. Tactile spatial resolution and two-point discrimination. Perception & Psychophysics 59, 131–138.

Faragher, R., Harle, R., Nov 2015. Location fingerprinting with Bluetooth low energy beacons. IEEE Journal on Selected Areas in Communications 33 (11), 2418–2426.
DOI: 10.1109/JSAC.2015.2430281

Golledge, R. G., 1993. Geography and the blind. Transactions of the Institute of British Geographers 18 (1), 63–70.
DOI: 10.2307/623069

Grisetti, G., Stachniss, C., Burgard, W., Feb 2007. Improved techniques for grid mapping with Rao-Blackwellized particle filters. IEEE Transactions on Robotics 23 (1), 34–46.
DOI: 10.1109/TRO.2006.889486

Kastner, A., Melani, C., 2000. Implementación del algoritmo Marching Cubes. Tech. rep., Departamento de Computación, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires, trabajo de cátedra / informe académico.

Klatzky, R. L., Lederman, S. J., 2002. Touch. Encyclopedia of Cognitive Science / or related reviews on tactile perception.

Kohlbrecher, S., Meyer, J., Graber, T., Petersen, K., Klingauf, U., von Stryk, O., Nov 2011. A flexible and scalable SLAM system with full 3D motion estimation. In: 2011 IEEE International Symposium on Safety, Security, and Rescue Robotics. pp. 130–136.
DOI: 10.1109/SSRR.2011.6106777

libgpiod project, 2024. libgpiod: C library and tools for interacting with the Linux GPIO character device. Documentación / paquete de distribución.
URL: https://packages.fedoraproject.org/pkgs/libgpiod/libgpiod/index.html

Loomis, J. M., 1981. Tactile perception and Braille design standards. Proceedings / review literature on tactile symbols.

Macenski, S., 2024. SLAM Toolbox. Repositorio GitHub del proyecto ROS 2.
URL: https://github.com/SteveMacenski/slam_toolbox

NVIDIA, 2024. Jetson Orin Nano Series Data Sheet. Datasheet / product documentation.
URL: https://connecttech.com/ftp/pdf/nvidia_jetson_orin_datasheet.pdf

Open Source Robotics Foundation, 2024. ROS 2 Jazzy Jalisco. Release / changelog de ROS 2.
URL: https://docs.ros.org/en/jazzy/Releases/Jazzy-Jalisco-Complete-Changelog.html

OpenCV, 2026. opencv-python. PyPI package.
URL: https://pypi.org/project/opencv-python/

Pololu Corporation, 2024. Micro Metal Gearmotors. Rev. 6.1, April 2024.
URL: https://www.pololu.com

Raspberry Pi, 2022. Raspberry Pi Pico 1 Datasheet. Datasheet del Raspberry Pi Pico / RP2040.
URL: https://www.raspberrypi.com/products/raspberry-pi-pico/

RobotWebTools, 2012. rosbridge_suite. Repositorio GitHub; WebSocket bridge for ROS.
URL: https://github.com/robotwebtools/rosbridge_suite

scikit-image developers, 2026. skimage.measure — skimage 0.26.0 documentation. Documentation page.
URL: https://scikit-image.org/docs/stable/api/skimage.measure.html

SmartCane, 2024. SmartCane.
URL: https://smartcane.org/

Toshiba Corporation, May 2008. TB6612FNG: Driver IC for Dual DC Motor. Document revision dated 2008-05-09.

WeWALK, 2024. WeWALK smart cane.
URL: https://wewalk.io/

Zandbergen, P. A., 2009. Accuracy of iPhone locations: A comparison of assisted GPS, WiFi and cellular positioning. Transactions in GIS 13 (s1), 5–28.
DOI: 10.1111/j.1467-9671.2009.01152.x