MAPEANDO EL FUTURO: VSLAM Y SUS ENFOQUES PARA LA NAVEGACIÓN AUTÓNOMA

Santiago Arturo Díaz Maturano Yesenia Eleonor González Navarro, Dra.
Paola Nayeli Cortez Herrera, M. en C.

Instituto Politécnico Nacional
UPIITA

sdiazm1700@alumno.ipn.mx ygonzalezn@ipn.mx pcortez@ipn.mx

Resumen

El presente artículo expone una visión general del concepto de SLAM y su aplicación en VSLAM, destacando su relevancia en la navegación autónoma, el reconocimiento y la reconstrucción de entornos. Se describe el procedimiento general que sigue una solución básica de VSLAM y se analizan tres algoritmos, cada uno con enfoques distintos y potenciales aplicaciones futuras.

Palabras clave: SLAM, VSLAM, mapeo, localización, optimización, odometría visual.

Introducción

En la última década, el mundo tecnológico ha vivido una gran evolución en temas de autonomía de las máquinas; cada vez existen más y más dispositivos que no requieren de una intervención humana para poder realizar sus funciones. Este cambio se ha reflejado principalmente en la industria automotriz y en el ámbito de la robótica. Donde los vehículos autónomos se ven como el futuro de la conducción y del transporte en nuestra sociedad. Del mismo modo, los avances en este campo pueden aplicarse también a sistemas robóticos, ya que prácticamente un robot y un automóvil

comparten características similares que les permiten beneficiarse a ambos de estas nuevas tecnologías. Dentro de estas soluciones innovadoras se encuentra la tecnología de Mapeo y Localización Simultánea (Simultaneous Localization and Mapping o SLAM), una herramienta fundamental que se ha consolidado como el método principal para implementar la navegación autónoma en sistemas como robots o vehículos, entre otros [1].

El SLAM, como su nombre lo indica, se encarga de localizar un sistema mientras construye simultáneamente un mapa del entorno. Para ello, el sistema extrae características del espacio en el que se encuentra y, con base en esa información, actualiza su posición y genera un modelo del entorno. De esta manera, el sistema no solo puede determinar su ubicación actual, sino también reconocer las áreas previamente exploradas y comprender el espacio en el que se ha desplazado. Una solución común para abordar un problema de SLAM consiste en utilizar métodos de estimación basados en datos de odometría, los cuales permiten calcular y actualizar de forma continua la posición y orientación estimadas del sistema dentro del entorno [2]. Estos métodos aprovechan la información obtenida a partir de distintos sensores, como cámaras, escáneres láser, sensores de proximidad o unidades de medición inercial (IMU), entre otros. Al combinar las lecturas de estos sensores, el sistema puede estimar su movimiento y reconstruir un mapa del entorno, incluso en ausencia de referencias externas. Este proceso constituye la base de muchos algoritmos de SLAM, ya que proporciona una representación inicial sobre la cual se realizan posteriores etapas de optimización y corrección de errores.

Este artículo se centrará en la técnica de VSLAM, definiendo primero su significado y los procesos que conforman una solución de este tipo. En la segunda parte, se analizan tres enfoques diferentes de VSLAM, describiendo su funcionamiento y algunas de sus aplicaciones.

VSLAM: La tecnología que permite a los robots moverse sin perderse

Dentro de las distintas implementaciones del SLAM, existe una basada en la visión, conocida como Visual SLAM (VSLAM). Esta variante se ha vuelto muy común debido a las ventajas que ofrecen los sensores visuales, tanto por su bajo costo y tamaño compacto como por la calidad y computabilidad de la información que proporcionan [3]. En este tipo de sistemas, se pueden utilizar cámaras monoculares, estéreo o RGB-D para capturar imágenes del entorno, las cuales se procesan con el fin de estimar la trayectoria del dispositivo adjunto a la cámara y construir un mapa del entorno que lo rodea.

Fases principales de una implementación VSLAM

Una solución de VSLAM contempla varios procesos, los cuales pueden ser vistos de mejor manera en la Figura 1.

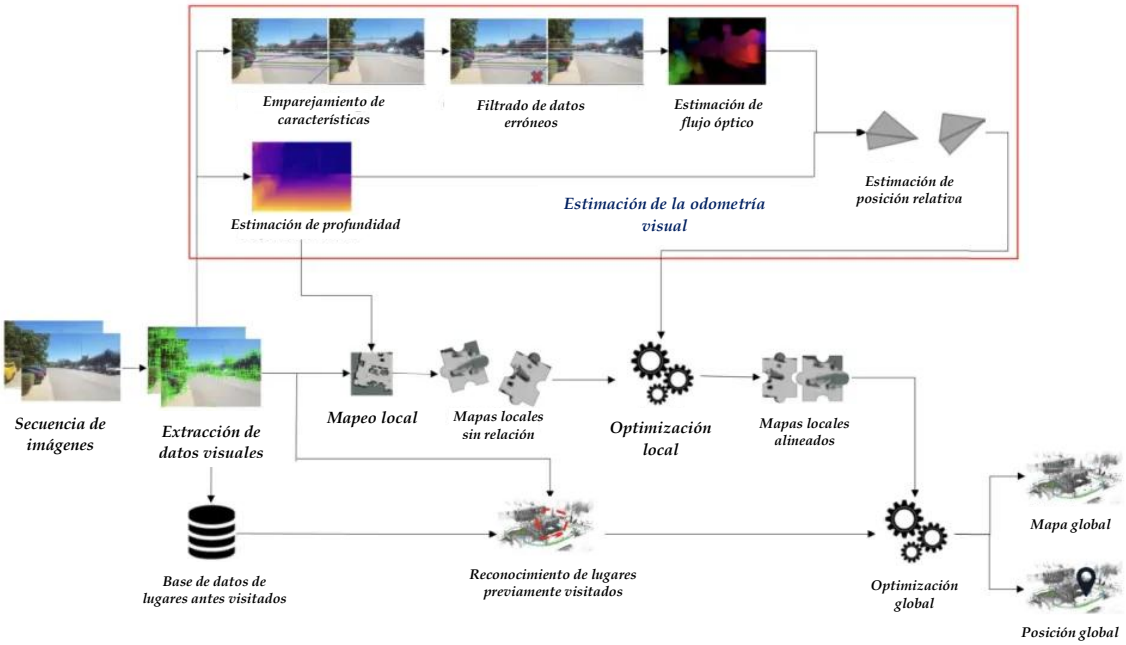

Figura 1. Flujo de trabajo de VSLAM [4]

La Figura 1 muestra varios procesos; sin embargo, estos se pueden clasificar en tres partes principales [5]: extracción de datos visuales y odometría visual, optimización y mapeo local, loop closure (reconocimiento de lugares previamente visitados) y optimización global (ver Figura 2). No obstante, antes de realizar cualquiera de estos pasos, el sensor seleccionado para este proceso debe pasar por una fase previa de calibración, con el fin de garantizar que los datos obtenidos sean lo más precisos y fiables posible.

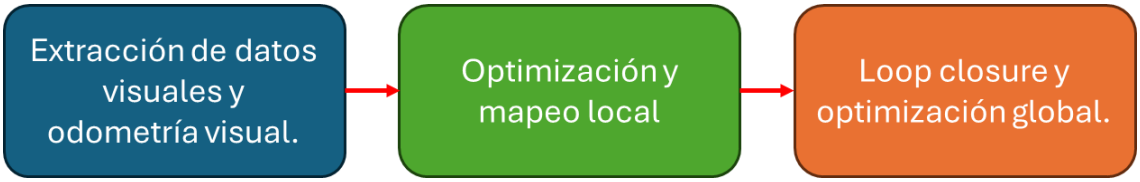

Figura 2. Diagrama de las etapas de una solución VSLAM

En la primera parte, se encuentran la extracción de los datos visuales y la odometría visual. Después de haber calibrado el sensor que recolecta los datos visuales (imágenes), estos ya están aptos para la extracción de características. Dichas características son elementos de la imagen que permiten al algoritmo reconocer, seguir y comparar partes del entorno de fotograma a fotograma. Este tipo de características puede variar según el tipo de algoritmo seleccionado, pero en la mayoría de los casos se utilizan puntos que identifiquen esquinas, bordes o texturas contrastantes. Con la odometría visual, los puntos obtenidos de las imágenes permiten estimar el movimiento de la cámara. El algoritmo analiza el cambio de posición de estos puntos entre fotogramas para calcular la traslación y rotación relativas. Así se reconstruye la trayectoria de la cámara sin necesidad de sensores externos. Sin embargo, al ser un método acumulativo, está sujeto a errores que se propagan en el

tiempo. Para reducir esta situación, suelen emplearse técnicas como loop closure, de la cual hablaremos más adelante.

Para la segunda parte, entran en acción la optimización local y el mapeo. La optimización local se encarga de refinar el mapa del entorno utilizando la información obtenida en la extracción de características y la odometría visual. En este proceso se ajustan las posiciones de la cámara y de los puntos en el espacio para reducir errores acumulados. Para ello se emplean métodos como la triangulación o la generación de mapas de profundidad, que permiten estimar con mayor precisión la estructura tridimensional del entorno.

Para la tercera parte, tenemos la identificación de lugares previamente visitados. Este proceso, conocido como loop closure detection, consiste en comparar las características visuales extraídas del entorno actual con aquellas almacenadas en el historial de características obtenidas durante el recorrido anterior. Cuando el sistema detecta similitudes características entre ambas, reconoce que se encuentra en una zona ya explorada. Este mecanismo permite corregir errores acumulados de estimación de posición y mejorar la precisión global del mapa, garantizando una reconstrucción más coherente del entorno. Ahora el paso final consiste en alinear el mapa completo y generar el resultado final, que incluye tanto el mapa optimizado del entorno como la pose estimada del sistema. En esta etapa, se integran todas las observaciones y trayectorias registradas durante el proceso de mapeo, ajustando posibles errores de alineación entre los diferentes fotogramas o mapas boceto. Mediante algoritmos de optimización global, se obtiene una representación más precisa del espacio, junto con la posición y orientación final del sensor o robot dentro de dicho entorno.

Tres enfoques que están transformando las soluciones VSLAM.

ORB-SLAM

Comenzamos describiendo uno de los algoritmos más reconocidos y utilizados en aplicaciones de VSLAM: ORB-SLAM. Diversos autores han desarrollado variantes basadas en las distintas versiones de este algoritmo [6], destacando principalmente la segunda versión, ORB-SLAM2. Debido a que esta versión es compatible con enfoques tanto monoculares como estéreo y RGB-D, además de que incorpora las técnicas de optimización global y reconocimiento de lugares previamente visitados (loop closure) [7], lo que permite obtener resultados en las aplicaciones de VSLAM más precisos y reales posibles. Su tercera versión, lanzada en 2021, incorporó ORB-SLAM Atlas, el primer sistema SLAM multimapa capaz de crear, reconocer, relocalizar y fusionar múltiples mapas generados en distintas sesiones de mapeo [7]. Esta característica permite al sistema mantener una precisión en la localización y reutilizar información previa, lo que mejora significativamente el desempeño en entornos amplios o exploraciones prolongadas.

Este algoritmo emplea el método de detección de características ORB (Oriented FAST and Rotated BRIEF), el cual combina dos métodos principales. EL primero es el detector FAST (Features from Accelerated Segment Test), que identifica esquinas y bordes distintivos en las imágenes. El segundo es el descriptor BRIEF (Binary Robust Independent Elementary Features), que permite comparar y encontrar coincidencias entre los puntos clave detectados en imágenes consecutivas [8]. Ahora que

se entiende cómo opera el algoritmo ORB, vale la pena ver cómo esta tecnología se ha perfeccionado para enfrentar desafíos reales.

A medida que la tecnología avanza, las implementaciones de sistemas automatizados se vuelven más comunes en distintas industrias. En el ámbito médico, por ejemplo, se han desarrollado robots que apoyan la logística interna de los hospitales, donde sus capacidades de navegación resultan esenciales para garantizar un servicio eficiente y preciso. En este contexto, Xiao et al. [9] presentaron una mejora del algoritmo ORB-SLAM3, optimizando su rendimiento para la navegación de robots logísticos en pasillos hospitalarios bajo condiciones de iluminación variables. Su propuesta logró un posicionamiento del robot más preciso, una planificación en las trayectorias más eficiente y una navegación más rápida y estable, superando el desempeño de las versiones tradicionales de ORB y SLAM.

DROID-SLAM

Actualmente, con el auge de la inteligencia artificial, las soluciones tecnológicas se orientan cada vez más hacia el aprendizaje automático basado en datos. En este contexto, Teed y Deng [10] desarrollaron un algoritmo que utiliza deep learning (aprendizaje profundo). Destacando por su alta precisión, al reducir el error en un 43 % en aplicaciones monoculares y un 71 % en estéreo en comparación con ORB-SLAM3. Además, el algoritmo puede trabajar con implementaciones monoculares, estéreo y RGB-D, pese a haber sido entrenado únicamente con entradas monoculares.

Su alto rendimiento y capacidad de generalización se deben a DROID (Diseño Recurrente Inspirado en Optimización Diferenciable) [10], un método basado en RAFT (Transformadas Recurrentes de Campo para todos los Pares), una arquitectura de redes profundas utilizada para la estimación de movimiento por píxel entre fotogramas de video [11]. Este método actualiza las poses de la cámara y los mapas de profundidad frecuentemente para refinar las estimaciones, minimizar desviaciones en trayectorias largas y permitir que un sistema monocular maneje entradas estéreo o RGB-D sin necesidad de reentrenamiento [10]. La Figura 3 muestra el resultado de una aplicación de este método de VSLAM.

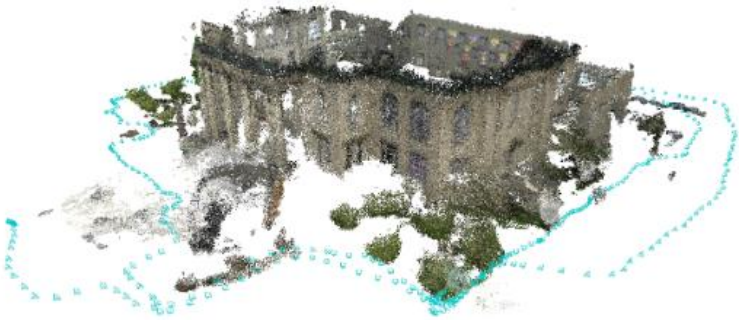

Figura 3. Resultado de una implementación de DROID-SLAM [10]

Otra aplicación de este algoritmo fue desarrollada por Tan et al., quienes emplearon DROID-SLAM para abordar la localización de vehículos autónomos en condiciones de lluvia [12]. Aunque existen sensores especializados que pueden manejar este tipo de entornos, suelen ser costosos y requieren mayores recursos computacionales para procesar la información. Por ello, los autores aprovecharon

las ventajas de los algoritmos visuales de SLAM para mejorar la precisión del posicionamiento y anticipar posibles dificultades derivadas de las condiciones climáticas. Esto resultó en una mejora de la precisión de posicionamiento del 50.83% en condiciones de lluvia y del 34.32% en condiciones normales, en comparación con otros algoritmos [12].

NICE-SLAM

Finalmente, encontramos una de las tecnologías de VSLAM con mayor proyección a futuro: los algoritmos basados en Campos de Radiancia Neurales (NeRF), considerados como la siguiente generación de sistemas SLAM. Una de sus características más llamativas es que implementa perceptrones multicapa (MLP) para la codificación de la geometría e iluminación como campos neuronales, lo que permite generar imágenes 2D realistas desde nuevos puntos de vista [13]. Sin embargo, al ser una tecnología relativamente reciente, todavía presenta áreas de mejora, principalmente en cuanto a su eficiencia y su capacidad para operar únicamente en entornos de pequeña escala. Uno de los algoritmos que abordó estas limitaciones es NICE-SLAM.

El funcionamiento del algoritmo comienza tomando un flujo de imágenes RGB-D como entrada. Divide el entorno en rejillas de características jerárquicas (Hierarchical Feature Grid), donde cada una almacena información local sobre la geometría y la apariencia, que luego se decodifica mediante redes neuronales para estimar profundidad y color [14]. Al comparar las imágenes renderizadas con las capturadas por la cámara, el sistema minimiza las pérdidas al volver a renderizar, optimizando de manera conjunta las poses de la cámara y la geometría del entorno. Esta solución mejora la eficiencia, precisión y escalabilidad, lo que permite reconstrucciones 3D más detalladas casi en tiempo real [14]. En la Figura 4 se muestra una reconstrucción del entorno obtenida al aplicar la técnica de NICE-SLAM.

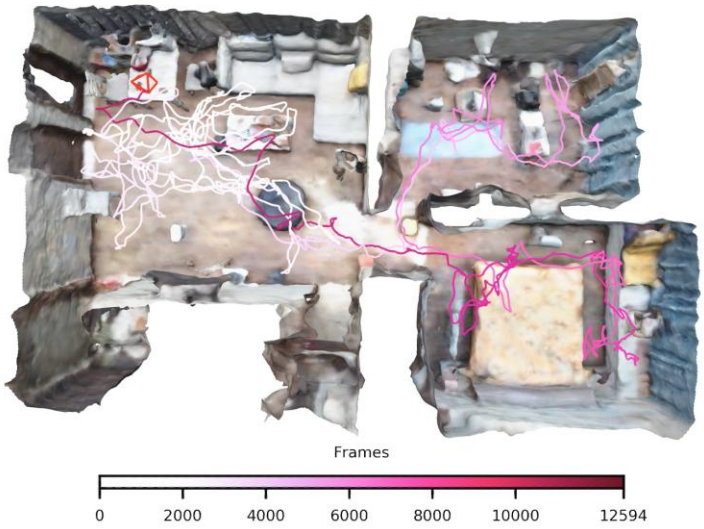

Figura 4. Reconstrucción 3D de un departamento usando NICE-SLAM [14]

Como se mencionó anteriormente, se trata de una solución nueva hasta cierto punto y, al tratarse de un algoritmo basado en NeRF, presenta las mismas ventajas y áreas de oportunidad que sus pares. Su capacidad de generar mapas 3D con gran detalle los hace principales candidatos para ser implementados en soluciones donde la información, el detalle y la precisión del entorno son clave,

como en navegación autónoma, realidad aumentada y robótica [15]. Sin embargo, una de sus grandes desventajas es la capacidad de procesamiento de la que normalmente carecen los sistemas embebidos por sus limitaciones tanto en tamaño como en peso [16].

Conclusiones

En conclusión, este artículo presenta el concepto general de SLAM y una de sus aplicaciones, VSLAM. Asimismo, se describió el procedimiento general que implica una solución básica de esta variante y, finalmente, se analizaron tres algoritmos con distintos enfoques que ofrecen diversas alternativas para su implementación futura. Esta es una tecnología en auge y, a medida que avanza, será implementada en más y más aplicaciones.

Referencias bibliográficas

[1] Stachniss, C. (2022). *Multi-Cue Direct SLAM*. Medium. https://medium.com/stachnisslab/multi-cue-direct-slam-a02cd054e13a

[2] Stachniss, C. (2013). *SLAM Course* [YouTube playlist]. YouTube. https://youtu.be/U6vr3iNrwRA?si=sjceQQeHtGXeKX5a

[3] Salih, O. M., & Vásárhelyi, J. (2024). Visual Data Compression Approaches for Edge-based ORB-VSLAM Systems. *Proceedings of the 2024 25th International Carpathian Control Conference, ICCC 2024*. https://doi-org.bibliotecaipn.idm.oclc.org/10.1109/ICCC62069.2024.10569696

[4] S. Mokssit, D. B. Licea, B. Guermah and M. Ghogho, "Deep Learning Techniques for Visual SLAM: A Survey," in IEEE Access, vol. 11, pp. 20026-20050, 2023, doi: 10.1109/ACCESS.2023.3249661.

[5] Cohen, J. (2024, 8 de marzo). The 6 Components of a Visual SLAM Algorithm [Blog]. Think Autonomous. https://www.thinkautonomous.ai/blog/visual-slam/

[6] Tourani, A., Bavle, H., Sanchez-Lopez, J. L., & Voos, H. (2022). Visual SLAM: What Are the Current Trends and What to Expect? Sensors, 22(23), 9297. https://doi.org/10.3390/s22239297

[7] Brasiliano Campos, C. A. R. L. O. S. A. L. B. E. R. T. O. (2021). ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual–Inertial, and Multimap SLAM. IEEE Transactions on Robotics. https://doi.org/10.1109/TRO.2021.3075644

[8] Rublee, E., Rabaud, V., Konolige, K., & Bradski, G.R. (2011). ORB: An efficient alternative to SIFT or SURF. 2011 International Conference on Computer Vision, 2564-2571.

[9] Xiao, F., Zhang, Y., Fang, J., Guo, X., & Huang, R. (2025). Improving the navigation optimization of hospital logistics robots under complex lighting changes by using improved ORB-SLAM3 and deep learning visual SLAM algorithm. Discover Applied Sciences, 7(4). https://doi.org/10.1007/s42452-025-06775-y

[10] Teed, Z., & Deng, J. (2021). DROID-SLAM: Deep Visual SLAM for Monocular, Stereo, and RGB-D Cameras. Neural Information Processing Systems.

[11] Teed, Z., & Deng, J. (2020). RAFT: Recurrent All-Pairs Field Transforms for Optical Flow. In A. Vedaldi, H. Bischof, T. Brox, & J.-M. Frahm (Eds.), Computer Vision – ECCV 2020 - 16th European Conference, 2020, Proceedings (pp. 402-419). (Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics); Vol. 12347 LNCS). Springer Science and Business Media Deutschland GmbH. https://doi.org/10.1007/978-3-030-58536-5_24

[12] Tan, Y. X., Meghjani, M., & Prasetyo, M. B. (2023). Localization with anticipation for autonomous urban driving in rain. arXiv preprint arXiv:2306.09134.

[13] Gao, K., Gao, Y., He, H., Lu, D., Xu, L., & Li, J. (2022). Nerf: Neural radiance field in 3d vision, a comprehensive review. arXiv preprint arXiv:2210.00379.

[14] Zhu, Z., Peng, S., Larsson, V., Xu, W., Bao, H., Cui, Z., Oswald, M.R., & Pollefeys, M. (2021). NICE-SLAM: Neural Implicit Scalable Encoding for SLAM. 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 12776-12786.

[15] Ghadimzadeh Alamdari, A., Zade, F. A., & Ebrahimkhanlou, A. (2025). A Review of Simultaneous Localization and Mapping for the Robotic-Based Nondestructive Evaluation of Infrastructures. Sensors, 25(3), 712. https://doi.org/10.3390/s25030712

[16] Zhuang, L., Zhong, X., Xu, L., Tian, C., & Yu, W. (2024). Visual SLAM for Unmanned Aerial Vehicles: Localization and Perception. Sensors, 24(10), 2980. https://doi.org/10.3390/s24102980

Referencia del artículo

Díaz, S., González, Y. & Cortez, P. **(mayo - junio, 2026).** Mapeando el futuro: VSLAM y sus enfoques para la navegación autónoma. *Boletín UPIITA. año 21, (114) 2026*