Comparativa de algoritmos de SLAM con ROS y Python

Josue Carlos Rodas Rosales[1]*, Ángel Israel Soto Marrufo[2], Francesco José García Luna[3]

Resumen

Este trabajo analiza la eficiencia computacional de tres algoritmos de SLAM (Simultaneous Localization and Mapping): Gmapping, Hector y Karto, aplicados en entornos simulados mediante ROS y Python. La navegación autónoma de robots móviles requiere mapear el entorno y determinar su posición y orientación, tarea que SLAM resuelve mediante la fusión de datos sensoriales. El estudio se realizó en tres escenarios virtuales (Hexágono, Living Room y Stage 4) utilizando el robot Turtlebot3 Burger en el simulador Gazebo. Se evaluaron dos métricas clave: uso de CPU y memoria RAM, aplicando análisis estadístico ANOVA para determinar diferencias significativas. Los resultados muestran que Gmapping consume más CPU en todos los entornos, aunque esta variable no es confiable por su baja explicación (<35 %). En contraste, el uso de memoria RAM está altamente explicado (>95 %), destacando a Hector como el más demandante. Finalmente, Karto se posiciona como el algoritmo más eficiente al requerir menor memoria y segundo menor uso de CPU, siendo recomendado para aplicaciones con recursos limitados.

Palabras Clave

SLAM – ROS – Algoritmos de navegación – Simulación robótica – Optimización computacional

[1,2,3]Departamento de Ingeniería Industrial y Manufactura, Universidad Autónoma de Ciudad Juárez, México.
***Autor de correspondencia:** al256012@alumnos.uacj.mx

Programa académico
Maestría en Ingeniería en Manufactura

Institución responsable del estudio
Universidad Autónoma de Ciudad Juárez

Fecha de presentación
27 de noviembre de 2025

Evento académico
1.er Congreso Internacional de las Fronteras de las Ingenierías

Financiamiento
SECIHTI (CVU 2085059)

Conflicto de interés
Sin conflicto de interés declarado

Referencias

1. Alsadik, B., & Karam, S. (2021). The Simultaneous Localization and Mapping (SLAM) - An Overview. *Journal of Applied Science and Technology Trends*, *2*(2), 147–158. https://doi.org/10.38094/jastt204117
2. Trejos, K., Rincón, L., Bolaños, M., Fallas, J., & Marín, L. (2022). 2D SLAM Algorithms Characterization, Calibration, and Comparison Considering Pose Error, Map Accuracy as Well as CPU and Memory Usage †. *Sensors*, *22*(18). https://doi.org/10.3390/s22186903
3. Herath, D., & St-Onge, D. (Eds.). (2022). *Foundations of Robotics: A Multidisciplinary Approach with Python and ROS*. https://doi.org/10.1007/978-981-19-1983-1_5
4. Cuevas Castañeda, C. C. (2016). Ros-gazebo. Una valiosa herramienta de vanguardia para el desarrollo de la robótica. *Publicaciones e Investigación*, *10*, 145. https://doi.org/10.22490/25394088.1593

CITACIÓN: Rodas Rosales, J.C., Soto Marrufo, A.I., & García Luna, F.J. (2026). Comparativa de algoritmos de SLAM con ROS y Python [Fronteras de la Ingeniería y Transformación Tecnológica]. *Memorias Científicas y Tecnológicas*, *5*(1), 58-59.

CONGRESO INTERNACIONAL DE LAS FRONTERAS DE LA INGENIERÍA
INVESTIGACIÓN PARA UN MUNDO CONECTADO
del 26 al 28 de Noviembre
2025

Universidad Autónoma de Ciudad Juárez
Instituto de Ingeniería y Tecnología
Departamento de Ingeniería Industrial y Manufactura
Maestría en Ingeniería en Manufactura

Comparativa de algoritmos de SLAM con ROS y Python

Josue Carlos Rodas Rosales
Dr. Ángel Israel Soto Marrufo, Dr. Israel Ulises Ponce Monárrez, Dr. Francesco José García Luna

Resumen

En la navegación de robots móviles se necesita mapear el entorno en que se encuentran y establecer su postura, posición y orientación. Una de las alternativas para resolver esto es la técnica llamada SLAM; la cual requiere la fusión de sensores que monitoreen el comportamiento del robot y entorno. Este trabajo presenta una comparación entre tres algoritmos de SLAM en tres entornos simulados mediante el uso de ROS y Python tomando en cuenta los recursos computacionales que estos algoritmos requieren para ejecutar su simulación; uso de CPU y memoria RAM. Los resultados obtenidos fueron evaluados mediante el análisis de varianza (ANOVA).

Introducción

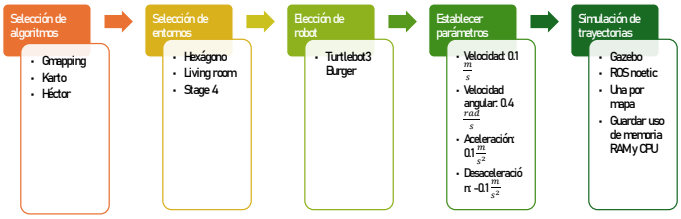
SLAM es una tecnología que permite a un robot móvil mapear un entorno desconocido tomando en cuenta su posición de manera simultánea sin necesitar sistemas de posicionamiento externos [1]. Esta técnica tiene algunas variantes, entre las que destacan los algoritmos que utiliza: Gmapping, Hector SLAM, Karto SLAM, entre otras [2].

Para usarla es necesario tener acceso y capacidad de operar múltiples sensores que incorpora el robot que se usará. Aquí es donde se utiliza ROS debido a que tiene por objeto facilitar el control, planeación, simulación y desarrollo de los robots mediante paquetes de programación [3], [4].

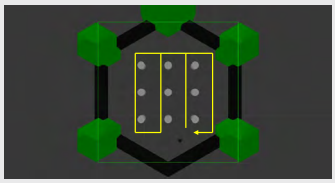
Objetivo general

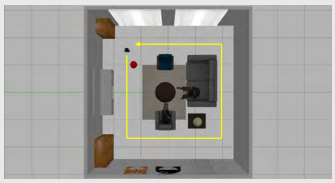
Comparar el rendimiento computacional; uso de CPU y memoria RAM, para tres tipos de algoritmos de SLAM; Gmapping, Karto y Hector, en tres mundos distintos simulados en el programa Gazebo utilizando ROS y Python para controlar la trayectoria seguida y almacenar la información necesaria.

Metodología



Resultados





Algoritmo	Entorno Hexágono				
	N	Media uso de CPU	Agrupación uso de CPU	Media uso de memoria RAM	Agrupación uso de memoria RAM
Gmapping	242	24.290	A	1097.22	B
Hector	242	18.972	B	1183.67	A
Karto	242	18.388	B	1065.80	C

Algoritmo	Entorno Living Room				
	N	Media uso de CPU	Agrupación uso de CPU	Media uso de memoria RAM	Agrupación uso de memoria RAM
Gmapping	157	24.276	A	1146.90	B
Hector	157	19.321	B	1231.13	A
Karto	157	19.431	B	1116.36	C

Algoritmo	Entorno Stage 4				
	N	Media uso de CPU	Agrupación uso de CPU	Media uso de memoria RAM	Agrupación uso de memoria RAM
Gmapping	331	23.323	A	1108.27	B
Hector	331	17.109	C	1192.22	A
Karto	331	20.067	B	1084.05	C

Conclusiones

Los ANOVA realizados a los resultados obtenidos muestran al algoritmo Gmaping-SLAM haciendo un mayor uso del CPU en los tres escenarios. Sin embargo, el análisis para esta variable arrojó que menos del 35% de esta es explicada por los factores estudiados por lo que no es una variable confiable para tomarla como referencia.
El uso de la memoria RAM es explicado en un porcentaje mayor al 95% por los factores utilizados mostrando al algoritmo Hector-SLAM como el que requiere mayor uso de esta.
Como conclusión, se escoge el algoritmo Karto como el más eficiente ya que es el que menos memoria RAM utiliza y es el segundo que menos CPU requiere.

Referencias:

[1].- B. Alsadik and S. Karam, "The Simultaneous Localization and Mapping (SLAM)-An Overview," Journal of Applied Science and Technology Trends, vol. 2, no. 2, pp. 147–158, Jul. 2021, doi: 10.38094/jastt204117.
[2].- K. Trejos, L. Rincón, M. Bolaños, J. Fallas, and L. Marín, "2D SLAM Algorithms Characterization, Calibration, and Comparison Considering Pose Error, Map Accuracy as Well as CPU and Memory Usage †," Sensors, vol. 22, no. 18, Sep. 2022, doi: 10.3390/s22186903.
[3].- D. Herath and D. St-Onge Eds, "Foundations of Robotics A Multidisciplinary Approach with Python and ROS," Sep. 2022. doi: 10.1007/978-981-19-1983-1_5.
[4].- C. C. Cuevas Castañeda, "Ros-gazebo. una valiosa Herramienta de Vanguardia para el Desarrollo de la Robótica," Publicaciones e Investigación, vol. 10, p. 145, Mar. 2016, doi: 10.22490/25394088.1593.

Cartel Académico: Comparativa de algoritmos de SLAM con ROS y Python.