

# Localización Simultánea y Mapeo para Control de un Robot Móvil Autónomo usando Escaneo de Nube de Puntos LiDAR y Métodos de Aprendizaje de Máquina

*Simultaneous location and mapping for control of an autonomous mobile robot using LiDAR point cloud scans and Machine Learning methods*

Ricardo Urvina Córdova <sup>1</sup>

Eduardo Aguilar Torres <sup>2</sup>

Álvaro Prado Romo <sup>3</sup>\*

<sup>1</sup>Universidad Católica del Norte. Departamento de Ingeniería de Sistemas y Computación. Antofagasta, Chile. Email: [ricardo.urvina@alumnos.ucn.cl](mailto:ricardo.urvina@alumnos.ucn.cl); [eduardo.aguilar@ucn.cl](mailto:eduardo.aguilar@ucn.cl); [alvaro.prado@ucn.cl](mailto:alvaro.prado@ucn.cl)

\* Autor de correspondencia: [alvaro.prado@ucn.cl](mailto:alvaro.prado@ucn.cl)

## RESUMEN:

Este artículo presenta técnicas alternativas de localización y mapeo simultaneo del ambiente, que permiten a los robots móviles auto referenciarse en un ambiente de navegación de reducida accesibilidad al posicionamiento mediante medios externos, como Sistemas de Posicionamiento Global (o GPS por sus siglas en inglés). La metodología consiste en implementar cuatro algoritmos de aprendizaje de máquina no-supervisado. Utilizando conjuntos de datos que se generan en base a una nube de puntos de rango entregados por las mediciones de un sensor LiDAR (o Detección de Luz y Rango por sus siglas en inglés). El enfoque propuesto identifica las características del mapa de navegabilidad y un método adicional, basado en el Filtro de Kalman Extendido (o EKF por sus siglas en inglés). EKF permite encontrar el posicionamiento del robot que se conjuga con cada uno de los algoritmos propuestos. El primer método propuesto consiste en la estimación de las características del ambiente mediante métodos heurísticos y la conformación del mapa mediante principios geométricos. El segundo método implementado se basa en K-Means para incorporar la incertidumbre en la medición del sensor, mientras que la tercera solución utiliza el modelo de mezcla de Gaussianas. El cuarto método se enfoca en el agrupamiento espacial de datos basado en densidad de datos con ruido (o DBSCAN por sus siglas en inglés). Para incluir incertidumbre dentro del ambiente de prueba, se induce error de odometría en el robot, la misma que se propaga hacia las lecturas del posicionamiento. Los resultados muestran que DBSCAN presenta un mejor tiempo de ejecución del sistema de localización propuesto frente a los otros métodos de comparativa. Además, la localización del robot es más precisa con este método, mostrando una reducción del 5% del error frente al resultado obtenido de los otros algoritmos propuestos. Finalmente, con los resultados alcanzados se prevé que con la disminución de error de localización y mapeo automático se pueda disminuir el consumo de los recursos del robot.

Palabras clave: Localización y mapeo simultaneo, K-medias, modelos de mezcla Gaussiana, agrupamiento espacial basado en densidad de aplicaciones con ruido, Filtro de Kalman Extendido, robot móvil autónomo

#### **ABSTRACT:**

This paper proposes several techniques for simultaneous localization and mapping of the environment, which allow mobile robots to self-reference themselves in a navigation environment with reduced accessibility by external means, such as Global Position System (GPS). The methodology consists of implementing four unsupervised machine learning algorithms. Using data sets that are generated based on a cloud of range points delivered by LiDAR sensor measurements. The proposed approach identifies characteristics from a navigability map, whereas an additional method based on Extended Kalman Filter (EKF), allows to find the robot positioning in conjunction. The proposed approach identifies navigability map features and an additional method, based on EKF. EKF allows find the robot positioning that is conjugated with each of the proposed algorithms. The first proposed method consists of estimating the features of the environment using heuristic methods and shaping the map using geometric principles. The second method is based on K-Means to incorporate the uncertainty in the sensor measurement, while the third solution uses the Gaussian Mixture Model (GMM). The fourth method focuses on Density-Based Spatial Clustering (DBSCAN). To include uncertainty within the test environment, odometry error is induced in the robot, which propagates to the positioning readings. The results show that DBSCAN presents a better execution time of the proposed localization system compared to the other comparative methods. In addition, the localization of the robot is more accurate with this method, showing a 5% reduction of the error compared to the result obtained from the other proposed algorithms. Finally, with the results achieved, it is expected that with the reduction of localization error and automatic mapping, the consumption of robot resources can be reduced.

Keywords: simultaneous localization and mapping, k-means, Gaussian Mixture Model, Density-based spatial clustering of applications with noise, Extended Kalman Filter, Autonomous ground vehicle.

## INTRODUCCIÓN

En las últimas décadas, la introducción de las tecnologías de información, el aprendizaje de máquina, y la robótica han sido herramientas clave para mejorar la eficiencia de las operaciones mineras tales como la exploración, perforación, manejo de explosivos, y transporte de material. Por ejemplo, maquinaria terrestre no tripulada provee una herramienta efectiva en faena para la inspección, extracción y acarreo de minerales; sin embargo, las condiciones hostiles del ambiente y superficies cerradas deniegan la localización del vehículo mediante GPS, lo cual es parte fundamental para la navegabilidad autónoma. Es así como se han planteado alternativamente los Sistemas de Localización y Mapeo Simultáneo (o SLAM)<sup>(1)</sup>, los cuales comprenden una estrategia clave para que robots móviles puedan navegar de manera autónoma a lo largo de trayectorias o caminos en ambientes desconocidos, donde la señal GPS puede estar ocluida por condiciones del robot o entorno <sup>(2-4)</sup>. La idea fundamental de SLAM radica en la estimación simultanea del trayecto que describe un robot y un mapa construido por sensores propioceptivos o exteroceptivos. Los datos que son extraídos habitualmente provienen de sensores de odometría, velocidad, o mayormente de percepción visual, los mismos que luego permiten determinar puntos navegables del espacio de trabajo <sup>(5)</sup>. De aquí que el objetivo de esta investigación es desarrollar un marco de trabajo para SLAM que asegure requerimientos de localización robusta para el control de movimiento autónomo en base a sensores de percepción.

Existen varios enfoques de SLAM que generalmente utilizan datos de rango obtenidos vía sensores LiDAR, los cuales pueden ser agrupados en: i) enfoques basados en emparejamiento y ii) técnicas por muestreo <sup>(6)</sup>. Los métodos basados en emparejamiento utilizan el cotejamiento geométrico para comparar datos de escaneos previos y actuales, y luego son empleados para calcular el desplazamiento y rotación del robot <sup>(7)</sup>. Por ejemplo, el Punto Iterativo más Cercano (o ICP por sus siglas en inglés) es un algoritmo heurístico que provee estimaciones precisas mediante la constatación de similitud entre datos asociados con una nube de puntos, usando generalmente la evaluación de un gradiente o mediante SVD <sup>(8)</sup>. Al igual que ICP, el algoritmo de Consenso de Muestras Aleatorias (RANSAC) presenta la solución al mapeo en base a registro de nubes de puntos, pero debido a la naturaleza de su metodología basada en optimización con gradiente. Estos dos últimos enfoques son dependientes de condiciones iniciales y están sujetos a mínimos locales <sup>(9)</sup>. Por otra parte, considerando incertidumbre de medición en la señal de rango, el método de Transformación de Distribución Normal <sup>(10)</sup> es utilizado para analizar la densidad de probabilidad de una nube de puntos y luego es aplicado para identificar el nivel de correlación que existe entre puntos cercanos, asumiendo que a cada medida de rango se le asocia una distribución de probabilidad Gaussiana <sup>(11)</sup>. En los dos casos, la eficiencia computacional no es suficiente para un importante número de puntos necesarios para una adecuada reconstrucción del mapa, por lo que los métodos basados en muestreo agilitan la extracción de características y complementan la construcción de primitivas en tiempo real.

Cámaras de profundidad o sensores de rango son extensivamente usados en aplicaciones de visión por computador, tales como reconstrucción 3D <sup>(12)</sup>, reconocimiento de objetos <sup>(13)</sup>, caracterización de obstáculos <sup>(14)</sup>, y SLAM <sup>(15)</sup>, por nombrar unos cuantos. Sin embargo, el uso de sensores de rango multicanal ha tenido aplicabilidad en SLAM por su menor complejidad de uso, aplicabilidad en entornos de iluminación variable, interiores y exteriores, utilidad en superficies

de rugosidad variada, y costo computacional reducido por el manejo de volúmenes de datos necesarios para la construcción de mapas densos <sup>(16)</sup>. Con la medición de rango, los objetos puntuales en escena pueden ser localizados y los objetos volumétricos pueden ser caracterizados geoméricamente; sin embargo, los conglomerados de datos inciertos y puntos dispersos requieren ser disociados o agrupados para identificar a cada objeto estructural de forma precisa y robusta dentro del mapa <sup>(17)</sup>. En particular, nubes de puntos adquiridas por sensores de profundidad son generalmente ruidosos, redundantes, y usualmente proveen solapamiento de la escena, por lo que se requiere de técnicas de filtrado y estimación para encontrar puntos vecinos afines que permitan alimentar de datos con menor variabilidad a la localización en SLAM <sup>(18, 19)</sup>. Entonces, la principal motivación de este trabajo radica en estimar en tiempo real la región más probable de las características de un entorno para robustecer un sistema de localización. Es así como se plantea el uso de diversas técnicas probabilísticas basadas en ML para la clusterización de datos y la posterior construcción de un mapa navegable para el robot.

La novedad de este trabajo radica en implementar métodos de aprendizaje no supervisado como: K-means <sup>(20)</sup>, GMM <sup>(21)</sup>, y DBSCAN <sup>(22)</sup>, para obtener de forma simple un mapa probabilístico del entorno navegable del robot en base a lecturas de rango de un sensor LiDAR. Este trabajo contribuye con una estrategia integral de SLAM que fusiona los métodos de aprendizaje no supervisado propuestos con EKF para la localización simultanea de un robot autónomo que navega en tiempo real <sup>(23)</sup>. El rol que desempeñan los métodos de clustering es el agrupamiento y manejo de las características identificadas por el sensor LiDAR. También se plantea una comparativa entre los métodos de SLAM propuestos en base a su desempeño, propiciando ventajas y desventajas de cada una de las estrategias implementadas en pruebas experimentales. Además, con el marco de SLAM propuesto, se genera una herramienta de localización robusta que puede ser utilizada en sistemas de control de movimiento de robots terrestres autónomos <sup>(24)</sup>.

Lo que continua del trabajo está organizado de la siguiente manera. La segunda sección presenta los trabajos relacionados con este estudio; la subsiguiente sección presenta la metodología para la construcción de un mapa probabilístico de características en base al agrupamiento de puntos de rango y la técnica utilizada para la localización del robot. Luego, en la siguiente Sección se presenta la implementación de las estrategias propuestas, mientras que los resultados alcanzados y su respectivo análisis son detallados en la subsiguiente Sección. Finalmente, se exponen las conclusiones más relevantes del trabajo realizado.

## **TRABAJOS RELACIONADOS**

Una estrategia de control de movimiento autónomo fundamentalmente requiere de sistemas de localización y mapeo precisos que permitan realimentar el posicionamiento de un robot dentro del ambiente de navegación <sup>(3)</sup>. En la práctica, es necesario que tanto la localización del robot como el mapeo del ambiente se produzcan de forma simultánea y en tiempo real para la referencia continua de movimiento de un vehículo robótico autónomo. Dado que, los sistemas de estimación de posición del robot durante un escenario de navegación autónoma suelen exhibirse ocluidos en interiores por la inhibición de señales satelitales en estructuras cerradas o

por la interferencia de los objetos en escena, es necesario optar por mecanismos auxiliares como SLAM <sup>(25-27)</sup>. En este caso, no trivial para minería profunda, SLAM se ha convertido en una tecnología capaz de estimar la propia posición de un robot y a su vez construir un mapa incremental de la escena en base a la lectura y fusión de datos, los mismos que son usualmente entregados por diversos tipos de sensores y sistemas de censado como unidad de medición inercial (IMU), cámaras, LiDARs <sup>(11-14, 17)</sup>. Por ejemplo, en <sup>(13)</sup>, se utiliza un sensor IMU para alinear escaneos de un LiDAR que permiten corregir la correspondencia entre mapas y mejorar consecuentemente la precisión de la localización de un vehículo robótico con SLAM.

El manejo de grandes cantidades de características entregadas por el sensor requiere que se estudie el tratamiento de estas nubes de puntos con diferentes métodos para optimizar los recursos de un robot, en este sentido existen varios trabajos relacionados. El trabajo en <sup>(28)</sup>, ofrece una descripción general de aplicabilidad de los sensores LiDAR, tomando como punto de referencia la perspectiva del procesamiento de datos para la conducción autónoma. Aquí se describen las principales funciones de LiDAR en la conducción y transporte autónomo. Además, se brindan amplia información acerca del estado del arte que se cuenta en la actualidad en términos de tecnologías y métodos de implementación con estos sensores.

Jonghwi kim y otros <sup>(6)</sup>, muestran cómo localizar un vehículo en zonas urbanas sin la necesidad de GPS. Esto se logra mediante la detección de los límites de los edificios extraídos de un mapa de referencia, que se comparan con los datos de nubes de puntos proporcionados por un sensor LiDAR. Con el mapa de referencia y los datos del sensor hacen un emparejamiento donde la información mutua se normaliza. El resultado de este emparejamiento se combina con las medidas de un sensor de inercia y la odometría mediante un EKF. La posición del vehículo se determina con los parámetros de rotación y traslación que maximizaron la información mutua entre la imagen de la nube de puntos y el mapa de referencia. Este resultado de combinación junto con la señal de odometría y las mediciones de un sensor inercial utilizando EKF permiten implementar un sistema de localización preciso y consistente.

Rubinstein y Erna Viterbi <sup>(17)</sup>, exhiben un sistema de localización y mapeo para robots autónomos dentro de túneles, donde proponen un sistema para el mapeo denominado LiTANK. El sistema depende completamente de las lecturas del sensor LiDAR, sin el uso de otros sensores como GPS u odometría. Mediante el registro de nubes de puntos se estiman la pose y el mapa de navegabilidad. Ryan Wolcott y Ryan Eustice <sup>(18)</sup>, presentan un sistema el cual empareja el escaneo de multi-resolución para la localización de vehículos mediante un método probabilístico. A un vehículo se le equipa con un escáner tridimensional y mediante un modelado de mezcla de Gaussianas, que caracterizan la distribución de altura y reflectividad del entorno logran rasterizar para facilitar la rápida inferencia y exacta de la multi resolución. En este trabajo se muestra un conjunto de datos que permiten localizar a un vehículo con sensores exteroceptivos sobre caminos encontrados en ambientes reales. En <sup>(23)</sup>, se implementa un sistema de localización de un robot en tiempo real con puntos de referencia artificiales, utilizando datos LiDAR para compensar la odometría en caso de giros del robot a alta velocidad. También implementan algoritmos para encontrar y rastrear las características del entorno con el método de localización de reflectores, cuya técnica radica en compensar el movimiento bajo condiciones óptimas del camino.

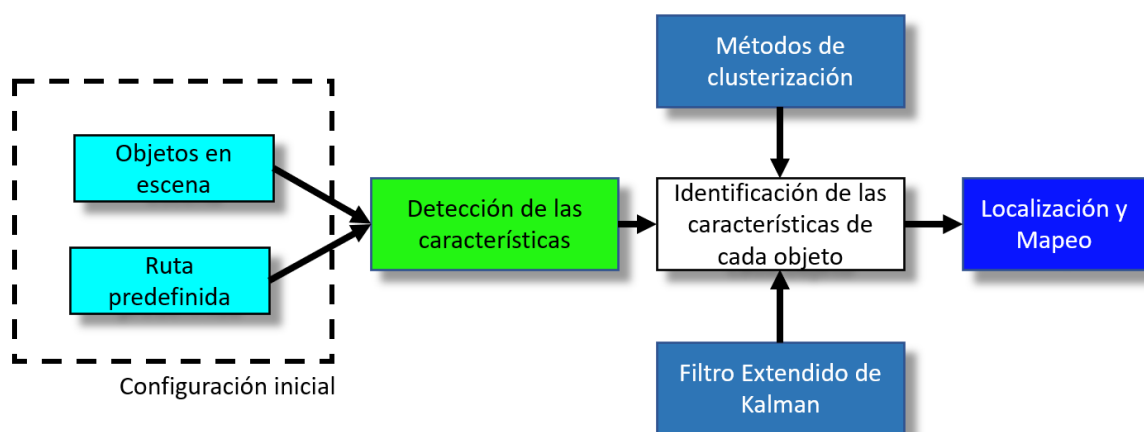
Pawel Slowak y otros <sup>(24)</sup> proponen un sistema SLAM utilizando EKF, que junto con los datos obtenidos por un sensor LiDAR 2D, permite a un robot móvil realizar tareas de forma autónoma. Aplicando el algoritmo ICP para estimar la localización del robot y mediante el empleo de métodos de clusterización, permiten agrupar formas que se obtienen mediante el reconocimiento y reconstrucción del mapa del entorno.

Xuebin Sun y otros <sup>(29)</sup>, presentan un esquema para comprender los datos de nube de puntos proporcionados por un sensor LiDAR. Proponen un método para evitar la distorsión y la pérdida de información mediante un esquema de compresión sin pérdidas basado en el agrupamiento de nubes de puntos. Este método se integra con técnicas de predicción, que tiene en cuenta la correlación de la información de distancia de los puntos para eliminar las redundancias espaciales en lugar de utilizar las predicciones de las imágenes directamente. La metodología propuesta se basa en convertir los datos de la nube de puntos estructurados en una imagen de rango. Luego usando un método de agrupamiento elimina la redundancia espacial. Finalmente, con los esquemas tradicionales se codifican los datos residuales y se obtiene un rendimiento de alta compresión de datos.

Los trabajos mencionados previamente indican el uso e implementación de SLAM con LiDAR en distintas técnicas, abordando así varios problemas de localización de robots y mapeo de ambientes dentro de áreas ocluidas o de difícil acceso.

## METODOLOGÍA

El estudio utiliza un algoritmo que permite la localización y mapeo de un robot móvil mediante el procesamiento de datos obtenidos a través del sensor y rayo láser y cómo estos datos permiten a un robot autónomo encontrar la pose adecuada para seguir con una ruta específica. Para esto se propone la metodología que es presentada en la Figura 1.



**Figura 1.** Breve esquema que representa los pasos incluidos en la metodología para la implementación del sistema de localización y mapeo simultáneo.

## Detección de características del mapa

El método propuesto asume que existe una trayectoria predefinida y un robot equipado con un sensor LiDAR. Inicialmente se fija la pose del robot móvil dependiendo la trayectoria predefinida para empezar la navegación. Después con el sensor LiDAR se ejecuta la detección de características de un mapa para cada instante de tiempo. La detección de características del mapa comprende la acción de identificar los objetos en escena que están dentro del alcance del sensor LiDAR, de tal manera que identifica las características de los objetos en escena alcanzados dentro un rango establecido. Estas características generan una nube de puntos la cual se trata con diferentes métodos de aprendizaje de máquina propuestos para el fin de la localización y mapeo simultaneo.

### Identificación de características de cada objeto

En esta etapa se toma la nube de puntos generada por la detección de características del mapa. Al procesar esta nube de puntos mediante los métodos de clusterización y el método heurístico se puede clasificar la nube de puntos generada por el LiDAR. Es así como se logra identificar las características de cada objeto. Para llevar a cabo esta etapa se consideran cuatro tipos de algoritmos, descritos como sigue:

1. **Algoritmo heurístico:** Este algoritmo hace el uso de medias y distancias para identificar las características de cada objeto. Haciendo un etiquetamiento mediante la obtención de la distancia euclidiana entre dos puntos consecutivos como se indica en la ecuación (1). Si dicha distancia es menor a un umbral predefinido significa que la característica corresponde al mismo objeto. Debido a que, el barrido del sensor de rango gira en sentido antihorario, se comparan consecutivamente las distancias hasta la última medición. Formalmente, se define como sigue:

$$d_j = \sqrt{(X_p^{(i)} - X_v)^2 + (Y_p^{(i)} - Y_v)^2} \quad (1)$$

donde,  $(x_p^{(i)}, Y_p^{(i)})$  es la i-ésima posición correspondiente a cada objeto y  $(X_v, Y_v)$  es la posición del vehículo en cada instante de tiempo.

2. **Algoritmo K-Means:** Es un algoritmo de aprendizaje no supervisado que permite agrupar conjuntos de datos en un número predeterminado de grupos de acuerdo con la cercanía de los datos con respecto al centroide. Una vez obtenido el grupo de medias por cada objeto se aplica este método de aprendizaje automático, donde se asigna un número de medias para toda la nube de puntos y el algoritmo se encarga de encontrar las medias que corresponden a cada grupo de información minimizando la distancia. Para el funcionamiento de este método se inicializan las posiciones de los centroides  $p_1, p_2, \dots, p_k$  y esto se repite hasta que la función de costo en la ecuación 2 converja. El conjunto de ecuaciones que soportan éste método pueden verse en la siguiente formulación (ver ecuaciones (2-4)):

$$c^{(i)} = \arg \min ||p_1 - up_j||_2^2 \quad (2)$$

$$up_j = \frac{\sum_{i=1}^m 1\{c^{(i)} = j\}p_i}{\sum_{i=1}^m 1\{c^{(i)} = j\}} \quad (3)$$

$$J(c, up) = \sum_{i=1}^m ||p_1 - up_j||_2^2 \quad (4)$$

donde,  $c^{(i)}$  es el  $i$ -ésima centroide,  $m$  es la cantidad de objetos detectados en ese instante de tiempo,  $up_j$  es la  $j$ -ésima estimación (por objeto),  $J(c, up)$  es la función de probabilidad a evaluar para determinar la convergencia de las medias. Además, en las ecuaciones anteriores, se observa la manera de calcular de forma iterativa los centroides  $up_j$  de cada grupo, con el objetivo de tener el mejor centroide asociado a cada objeto. Este método de clusterización usa la nube de puntos generada en la detección de características de mapa y después de aplicar K-Means se obtiene cada objeto con sus características y el centroide más representativo de estas características.

3. **Algoritmo GMM:** Este algoritmo se centra en un enfoque para agrupar el conjunto de datos y su matriz de covarianza asociada al error de propagación. En cada iteración se calcula el mejor estimado de cada punto, en este caso de cada objeto, y las covarianzas se determinan en base a dichos pesos y medias. El modelo hace uso del algoritmo de expectativas (EM), este itera sobre los pasos de expectativa (paso E) y maximización (paso M) hasta que las estimaciones de los parámetros convergen, es decir, para todos los parámetros en cada iteración. El paso de expectativa o paso E, consiste en calcular la expectativa de las asignaciones de los componentes  $C_k$  para cada punto de los datos  $X_i \in X$  dados los parámetros del modelo  $\phi_k, u_k$  y  $\sigma_k$ . El segundo paso de maximización consiste en maximizar las expectativas calculadas en el paso de expectativa con respecto a los parámetros del modelo. Este paso consiste en actualizar los valores de  $\phi_k, u_k$  y  $\sigma_k$ . Todo el proceso iterativo se repite hasta que el algoritmo converge, dando una estimación de máxima verosimilitud. Intuitivamente el algoritmo de maximización de expectativas comienza con un paso de inicialización, que asigna los parámetros del modelo a valores razonables en función de los datos. El algoritmo está descrito en las ecuaciones (5-12).

#### Paso de inicialización.

Se asigna aleatoriamente muestras sin reemplazo del conjunto de datos  $X = \{x_1, \dots, x_N\}$  a las estimaciones medias de los componentes  $\hat{\mu}, \dots, \hat{\mu}_k$ . Establecer todas las estimaciones de la varianza de los componentes en la varianza de la muestra:

$$\hat{\sigma}_1^2, \dots, \hat{\sigma}_k^2 = \frac{1}{N} \sum_{i=1}^N (x_i - \bar{x})^2 \quad (5)$$



donde,  $\bar{x}$  es la media de la muestra

$$\bar{x} = \frac{1}{N} \sum_{i=1}^N x_i \quad (6)$$

después de establecer todas las estimaciones previas de distribución de componentes a la distribución uniforme

$$\hat{\phi}, \dots, \hat{\phi}_k = \frac{1}{K} \quad (7)$$

#### Paso de expectativas:

Calcular  $\forall_i, k$

$$\hat{Y}_{ik} = \frac{\hat{\phi}_k N(x_i | \hat{\mu}_k, \hat{\sigma}_k)}{\sum_{j=1}^k \hat{\phi}_j N(x_i | \hat{\mu}_j, \hat{\sigma}_j)} \quad (8)$$

donde,  $\hat{Y}_{ik}$  es la probabilidad de que  $x_i$  sea generada por el componente  $C_k$ , de este modo

$$\hat{Y}_{ik} = p(C_k | x_i, \hat{\phi}, \hat{\mu}, \hat{\sigma}) \quad (9)$$

#### Paso de maximización:

$$\hat{\phi}_k = \sum_{i=1}^N \frac{\hat{Y}_{ik}}{N} \quad (10)$$

$$\mu_k = \frac{\sum_{i=1}^N \hat{Y}_{ik} x_i}{\sum_{i=1}^N \hat{Y}_{ik}} \quad (11)$$

$$\sigma_k^2 = \frac{\sum_{i=1}^N \hat{Y}_{ik} (x_i - \hat{\mu}_k)^2}{\sum_{i=1}^N \hat{Y}_{ik}} \quad (12)$$

donde el número de componentes  $K$  no se conoce a priori. Generalmente en este caso se estima el número de componentes y se ajusta el modelo a los datos usando el criterio del codo. Esto se hace para muchos valores diferentes de  $K$ . Por lo general, se mantiene el modelo con el mejor compromiso entre ajuste y número de componentes, donde los modelos más simples tienen menos componentes.

4. **Algoritmo DBSCAN:** Es un algoritmo de agrupamiento de datos que se basa en densidad debido a que encuentra un número de clústeres. El algoritmo inicia estimando la distribución de densidad de los nodos correspondientes que se resume en encontrar muestras del núcleo de alta densidad y se expande a grupos a partir de ella. Una vez identificado las características correspondientes a cada objeto se calcula la media y la covarianza de cada uno de estos grupos en cada instante de tiempo para la clusterización

de los objetos en el mapa. El funcionamiento de DBSCAN requiere de al menos 2 parámetros los cuales son: i)  $\varepsilon$  el cual especifica lo cerca que deben estar los puntos entre sí para ser considerados parte de un clúster, ii) puntos mínimos, los cuales especifican el número mínimo de puntos para formar una región densa. Como resultado de este método se tienen 3 tipos de puntos de datos: a) punto núcleo el cual es un punto que tiene más de un número especificado de puntos mínimos dentro de un radio de  $\varepsilon$  a su alrededor, b) punto de borde el cual contiene menos puntos mínimos dentro de la región de análisis, pero se encuentra en la vecindad de un punto de núcleo, y c) punto de ruido, el cual no es un punto de borde o de núcleo. A continuación, se indica la ecuación (13) que detalla la función de densidad:

$$N_{\varepsilon}(p): \{q | d(p, q) \leq \varepsilon\} \quad (13)$$

donde,  $N_{\varepsilon}(p)$  es la densidad de datos cercanos;  $\varepsilon$ -vecindario son los objetos dentro de un radio de  $\varepsilon$  desde un objeto;  $d(p, q)$  es la densidad de un conjunto  $p$  o  $q$ . Se considera densidad alta cuando  $\varepsilon$ -vecindario de un objeto contiene al menos puntos mínimos de los objetos.

Finalmente, en esta etapa después de aplicar el método heurístico o los métodos de aprendizaje no supervisado se procede a calcular la matriz de covarianza de las características en base a las medias obtenidas por cada objeto. La elipse que es el resultado del cálculo de la covarianza se determina en base a los valores, y vectores propios de la matriz de covarianza de los puntos muestreados. El valor propio de mayor magnitud indica la mayor cantidad de covarianza de distribución de la información y, el segundo valor propio caracteriza el segundo eje de un elipsoide. Finalmente, el ciclo de identificación del mapa se repite hasta que el robot haya terminado de recorrer la trayectoria preconfigurada.

## Localización y Mapeo

Además de los métodos de aprendizaje no supervisado se utiliza el EKF para la estimación de la pose del robot considerando la cinemática no-lineal del sistema. EKF es un elemento del sistema SLAM, el cual permite la combinación indirecta de los datos registrados por el sensor LiDAR después de cada medición. EKF usa la predicción del vector de estado del sensor donde, se asume una velocidad constante de movimiento del escáner láser entre mediciones. Esto se puede caracterizar con las siguientes ecuaciones (14,15):

$$\hat{x}_{k+1|k} = \phi \hat{x}_{k|k} \quad (14)$$

$$P_{k+1|k} = \phi P_{k|k} \phi^T + Q_k \quad (15)$$



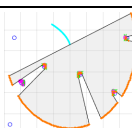
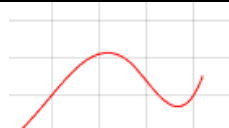




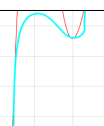
donde  $\hat{x}_{k+1|k}$  es el vector de estado, el cual contiene las estimaciones de los parámetros cinemáticos del sistema;  $\hat{x}_{k|k}$  es el vector estimado por el filtro en el paso de tiempo anterior;  $\phi$  es la matriz de transición de estados. La variable  $k + 1|k$  significa que la predicción se relaciona con el estado del instante  $k + 1$ . Las matrices  $P_{k+1|k}$  y  $P_{k|k}$  representan la covarianza de la matriz de error, mientras que  $Q_k$  es una matriz de covarianza asociada al ruido del proceso, la misma

que pondera el efecto de afectación del modelo sobre los efectos aleatorios del movimiento del robot. La matriz  $Q_k$  en este caso es dada previo conocimiento de la fidelidad del modelo, siendo así sus elementos de menor tamaño cuando no existe error de odometría o mayor cuando se considera este error.

## CONFIGURACIÓN EXPERIMENTAL

Los métodos desarrollados se evaluaron en un computador con un procesador Intel® Core™ i9-20980HK CPU @ 3.10GHz, 32GB de RAM en Windows 11. El rango del sensor LiDAR es de 8 metros. Los elementos del entorno considerados en la experimentación se muestran en la Tabla 1.

**Tabla 1.** Ilustraciones del entorno de simulación

Nombre	Descripción	Representación
Robot sin error de odometría	Elemento que sigue una trayectoria dentro del mapa y sobre este se encuentra el sensor LiDAR.	
Robot con error de odometría	Elemento que sigue una trayectoria dentro del mapa sumado a un error de odometría.	
Sensor LiDAR	Elemento que permite identificar las características del mapa en cada instante de tiempo	
Trayectoria de referencia	Elemento que está definido por puntos en el eje x e y dentro del entorno de la simulación por donde se mueve el robot sin odometría.	
Objeto en escena	Elemento que está definido por un punto en el eje x e y dentro del entorno de simulación.	
Características del objeto	Elemento que está definido por puntos verdes dentro del entorno de simulación.	
Valores medios del rango del objeto	Elemento que está representado por puntos de color lila dentro del entorno de simulación.	
Región más probable de localización del objeto.	Elemento que está representado por una elipse dentro del entorno de simulación	
Ruta estimada	Elemento que está representado por una secuencia de puntos de color celeste.	

La Figura 2, detalla el flujo de la ejecución de la metodología propuesta en donde, tenemos un robot móvil, una trayectoria predefinida que debe seguir el robot dentro del mapa probabilístico, y objetos en escena definidos aleatoriamente dentro del mapa. En este experimento, se evalúa la precisión del robot para seguir la trayectoria definida y que el robot no se pierda dentro del mapa en el caso de no lograr detectar los objetos. Ya que mediante la detección de los objetos esto permite construir el mapa de navegabilidad. A continuación, se describen los pasos realizados durante la ejecución del sistema SLAM.

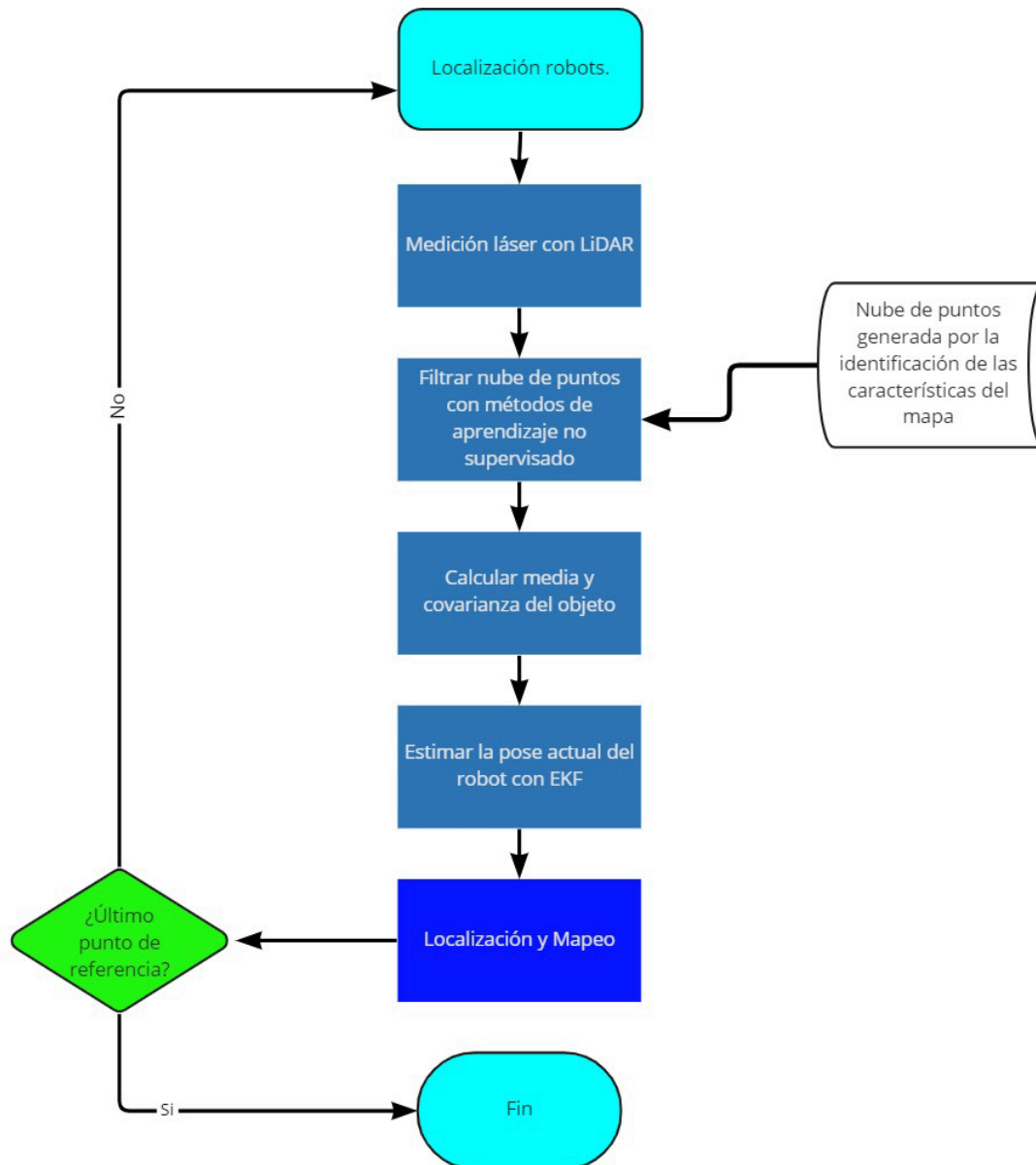
### **Paso 1: Localización de los robots para el tiempo $t_0$**

Considere que tenemos 2 robots, que se diferencian en que uno no asume un error en la odometría (robot 1) y el otro sí (robot 2). Además, contamos con una trayectoria predefinida que es representada por una serie de puntos dentro del mapa. Con el camino de referencia que se cuenta, el robot 1 se localiza en el primer punto de esa trayectoria. La pose inicial del robot se calcula con el primer punto en  $x$  e  $y$  de referencia y el ángulo con el cual empieza el robot está definido en la ecuación (16).

$$\tan^{-1} \frac{Y_2 - Y_1}{X_2 - X_1} \quad (16)$$

donde  $Y_1$  es el primer punto de referencia de la trayectoria en la coordenada  $y$ ,  $Y_2$  es el segundo punto en la coordenada  $y$  mientras que,  $X_1$  es el primer punto de referencia de la trayectoria en la coordenada  $x$  y  $X_2$  es el segundo punto de referencia en  $x$ . Los robots van cambiando de posición a medida que se mueve por la trayectoria. El ángulo que toma el robot en cada instante de tiempo se calcula con el punto actual donde se encuentra localizado el robot y el siguiente punto dentro de la trayectoria, el *robot 2* va avanzando a la misma velocidad que el *robot 1*, pero sumado el error de odometría según la ecuación (17).

$$\sum Total = \sum objetos + \sum odometría \quad (17)$$



**Figura 2.** Diagrama de Flujo del sistema propuesto para la localización y mapeo del robot.

### **Paso 2: Medición de láser con LiDAR.**

El sensor LiDAR se encuentra sobre el robot móvil en el centro de gravedad con un ángulo de visión de  $180^\circ$  y un rango de 8 metros máximos establecido. Es por lo que el láser apunta en la misma dirección que el frente del robot. Con la medición del sensor se identifican las coordenadas globales del mapa para cada instante de tiempo. Logrando así tener un entorno real del mapa mientras el robot se moviliza por el mismo.

**Paso 3: Filtrar la nube de puntos usando el método heurístico o los métodos de aprendizaje de máquina.**

Después de la medición de láser en cada instante de tiempo obtenemos una matriz de puntos en el mapa de coordenadas. Estos representan todas las características de los objetos detectados en cada medición de tiempo. Con la matriz de puntos generada, se determina a qué objeto pertenece cada característica. Después, se aplica el método heurístico o los métodos de ML para clusterizar esta nube de puntos y así identificar las características de cada objeto.

#### **Paso 4: Calcular la media y covarianza de cada clúster.**

Con la obtención de las características de los objetos es necesario calcular la media y la covarianza de cada uno de estos grupos de puntos. Esto se realiza para indicar donde posiblemente se encuentran estos datos y así a construir el mapa de navegabilidad. Este cálculo depende del método de ML que se esté usando para el filtrar la nube de puntos generada por la medición del láser ya que: i) Cuando usamos el método heurístico la media y la covarianza se calcula usando el grupo de datos que corresponde a cada objeto. ii) El método de K-Means nos devuelve la clusterización para cada objeto con sus medias correspondientes. En este caso se procede solo a calcular la covarianza del grupo de datos entregado por K-Means. iii) Al emplear GMM no calculamos individualmente la media y covarianza de cada clúster ya que, GMM entrega esta información por cada grupo de datos. iv) Finalmente, con DBSCAN se calcula la media y la covarianza de igual manera que en el método heurístico, debido a que este método devuelve la identificación de las características de cada objeto.

#### **Paso 5: Estimar la pose actual del robot mediante EKF.**

Finalmente, el algoritmo calcula la nueva pose mediante EKF, mediante 2 etapas:

1. Actualización de las mediciones: Esta etapa consiste en la adquisición de la información del sensor láser para la corrección de la matriz de covarianza y la estimación de la pose del robot. Primero se calcula la ganancia de Kalman mediante las siguientes ecuaciones (18-20):

$$k_i = \sum_{t|t+1} H^T (R + H \Sigma_{t|t+1} H^T)^{-1} \quad (18)$$

donde:

$$h = \hat{z} = \left( \frac{p}{\varphi} \right) = \left[ \begin{array}{c} \sqrt{(x_i - \hat{x}_v)^2 + (y_i - \hat{y}_v)^2} \\ \arctan \left( \frac{y_i - \hat{y}_v}{x_i - \hat{x}_v} \right) - \hat{\varphi}_v \end{array} \right] \quad (19)$$

$$H = \left( \frac{\partial h}{\partial x} \right) = \left[ \begin{array}{ccc} \frac{x_i - \hat{x}_v}{r} & \frac{y_i - \hat{y}_v}{r} & 0 \\ \frac{y_i - \hat{y}_v}{r^2} & \frac{x_i - \hat{x}_v}{r^2} & -1 \end{array} \right] \quad (20)$$

dado que el modelo de la medición  $h$  es no-lineal, se han calculado los Jacobianos. Entonces, se resta la estimación y se actualiza la matriz de covarianzas conforme se indica en las ecuaciones (21,22)

$$\hat{x}_{t|t} = \hat{x}_{t|t+1} + K_t(z_t - \hat{z}_{t|t+1}) \quad (21)$$

$$\Sigma_{t|t} = (I - K_t H_t) \Sigma_{t|t-1} \quad (22)$$

donde  $x_i$  e  $y_i$  corresponden a las coordenadas del  $i$ -ésimo objeto,  $\hat{\phi}_v$  es la estimación del ángulo de orientación del vehículo,  $\hat{x}_v = \hat{x}_{t|t}$  es la estimación de la pose del vehículo en el instante de tiempo,  $\Sigma_{t|t}$  corresponde a la actualización de la matriz de covarianza en cada instante de tiempo,  $K_t$  es la constante de Kalman,  $H, h$  corresponden al Jacobiano y al modelo no lineal del sistema de medición respectivamente. En el conjunto de ecuaciones anteriores se tiene que los dos lados de las ecuaciones corresponden a dos sistemas de coordenadas diferentes (rectangular y polar), por lo que se procede a cambiar a uno de los dos lados a un solo sistema de coordenadas. Para el caso de este trabajo se consideraron a todos los sistemas en el eje de coordenadas rectangulares alineados al marco de coordenadas global, con el fin de estimar directamente la localización del vehículo.

2. Actualización de tiempo: en esta fase del ciclo EKF, se predice los estimadores de posición del vehículo y se determina la propagación de la covarianza. Para lo cual se utiliza el modelo del robot unicycle como se muestra en la ecuación (23)

$$\begin{pmatrix} x \\ y \\ \theta \end{pmatrix}_{k+1} = \begin{pmatrix} x \\ y \\ \theta \end{pmatrix}_{k+1} + \Delta t \begin{pmatrix} \cos \theta & 0 \\ \sin \theta & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} v \\ w \end{pmatrix} \quad (23)$$

donde, se observa que el modelo no corresponde a un modelo lineal en la variable  $\theta$ , por lo que para implementar EKF se debe evaluar el sistema en un punto de equilibrio, es así como se determina el Jacobiano de la matriz de transición como se muestra en la ecuación 23 y las ecuaciones (24,26) son de predicción,

$$F = \frac{\partial g}{\partial x} = \begin{pmatrix} 1 & 0 & -\Delta t \sin(\theta) \\ 0 & 1 & \Delta t \sin(\theta) \\ 0 & 0 & 1 \end{pmatrix}_{\hat{x}_{t|t}} \quad (24)$$

$$\hat{x}_{t|t+1} = g(x, y, \theta)|_{\hat{x}_{t|t}} \quad (25)$$

$$E_{t+1|t} = Q + F E_{t|t} F^T \quad (26)$$

teniendo en cuenta que las matrices  $Q$  y  $R$  son simétricas y son utilizadas para dar mayor o menor ponderación al proceso o a la medición. Considerando un estado inicial, el mismo que es igual a pose del robot. Igualmente, la matriz de covarianza del error asociada a la estimación a priori se consideró como una matriz simétrica con un valor alto en su diagonal.

El algoritmo de localización se basa en varios parámetros que se pueden manipular para mejorar la calidad del mapeo y la ubicación. Estos parámetros fueron estudiados durante este trabajo y los resultados se muestran en la siguiente Sección. Los parámetros son:

- Distancia mínima para el cálculo de los clústeres a partir de la nube de puntos.
- Rango del sensor láser.
- Número mínimo de vecinos para la obtención de clústeres según el método.

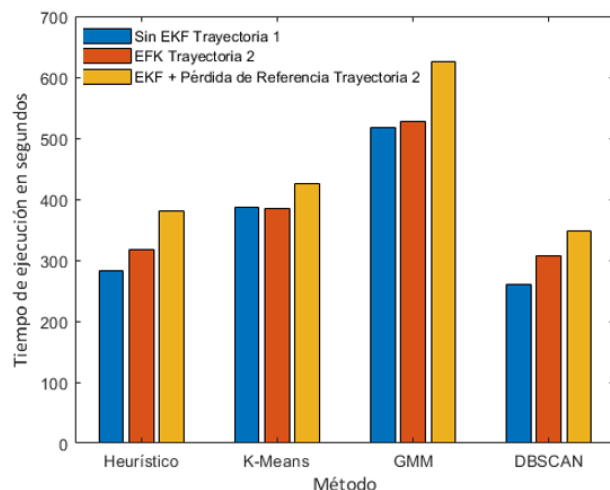
## **ANÁLISIS DE RESULTADOS**

Para el análisis de resultados en los cuatro métodos propuestos se utilizó la misma configuración inicial del ambiente en cuanto a los objetos y la trayectoria. A continuación, en la Figura 3 se presentan los tiempos de ejecución para cada método. En la Figura 4 se muestra el error acumulado por cada método. El error se calculó con la trayectoria estimada y la trayectoria predefinida. Calculando la distancia entre cada punto en el eje  $y$  cuando las trayectorias coinciden en el eje  $x$ .

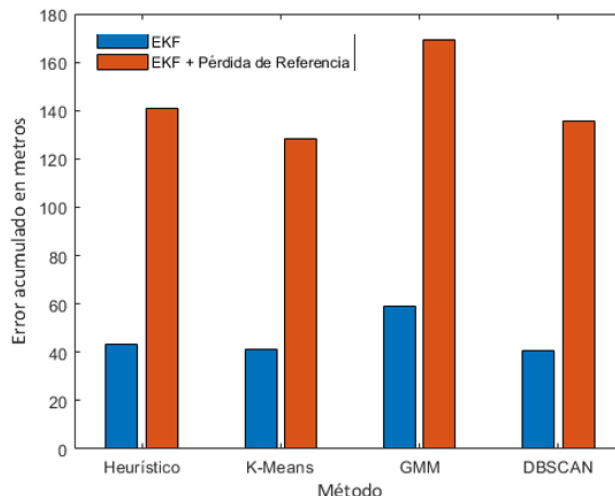
En la Figura 3, se observa que DBSCAN presenta menores tiempos de ejecución del experimento en los distintos ambientes propuestos frente a los otros métodos de clusterización. Esto se debe al algoritmo que emplea DBSCAN al realizar la clusterización. La clave está en que este algoritmo busca la vecindad de cada punto en un grupo que está dentro de un radio con un mínimo de puntos para el agrupamiento. Incluso, maneja de manera eficiente los valores atípicos y los conjuntos de datos ruidosos. DBSCAN funciona correctamente en el experimento realizado debido a que, los datos no tienen una alta dispersión. Por otro lado, GMM empleó mayor tiempo en ejecutar el experimento, esto se debe a que realiza más operaciones en conjunto para encontrar las medias y covarianzas.

En la Figura 4 se indica que al emplear DBSCAN con EKF estima de mejor manera la posición del robot debido a que en las pruebas realizadas a los cuatro métodos propuestos es el que menos error de posicionamiento acumuló.



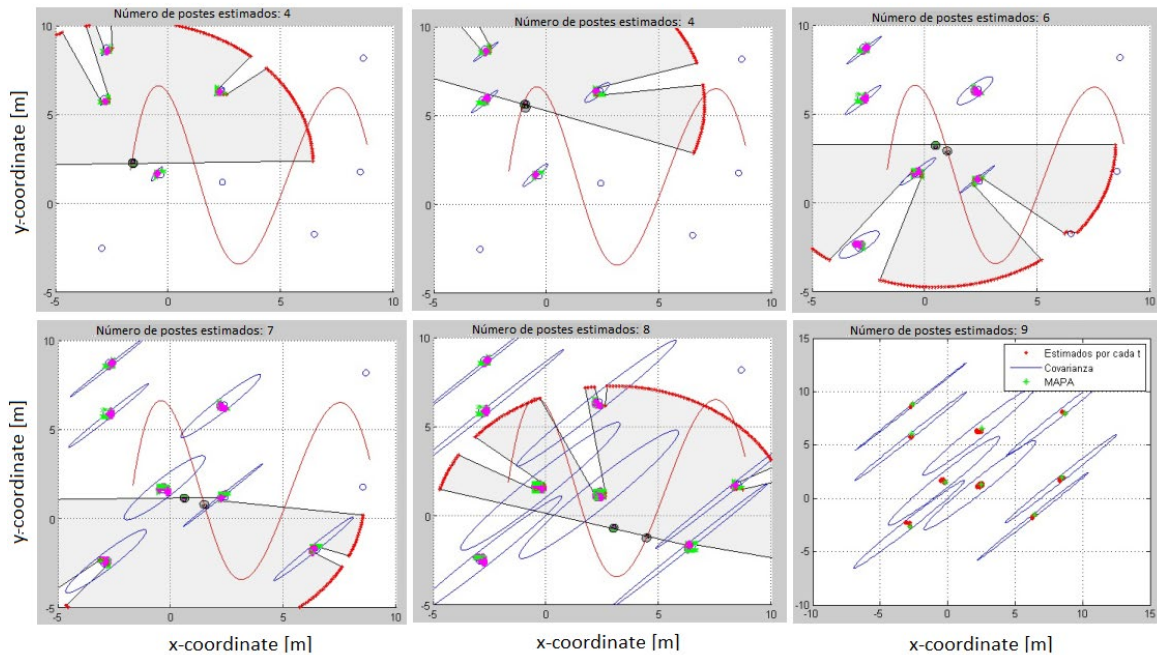


**Figura 3.** Tiempo de ejecución por cada método.



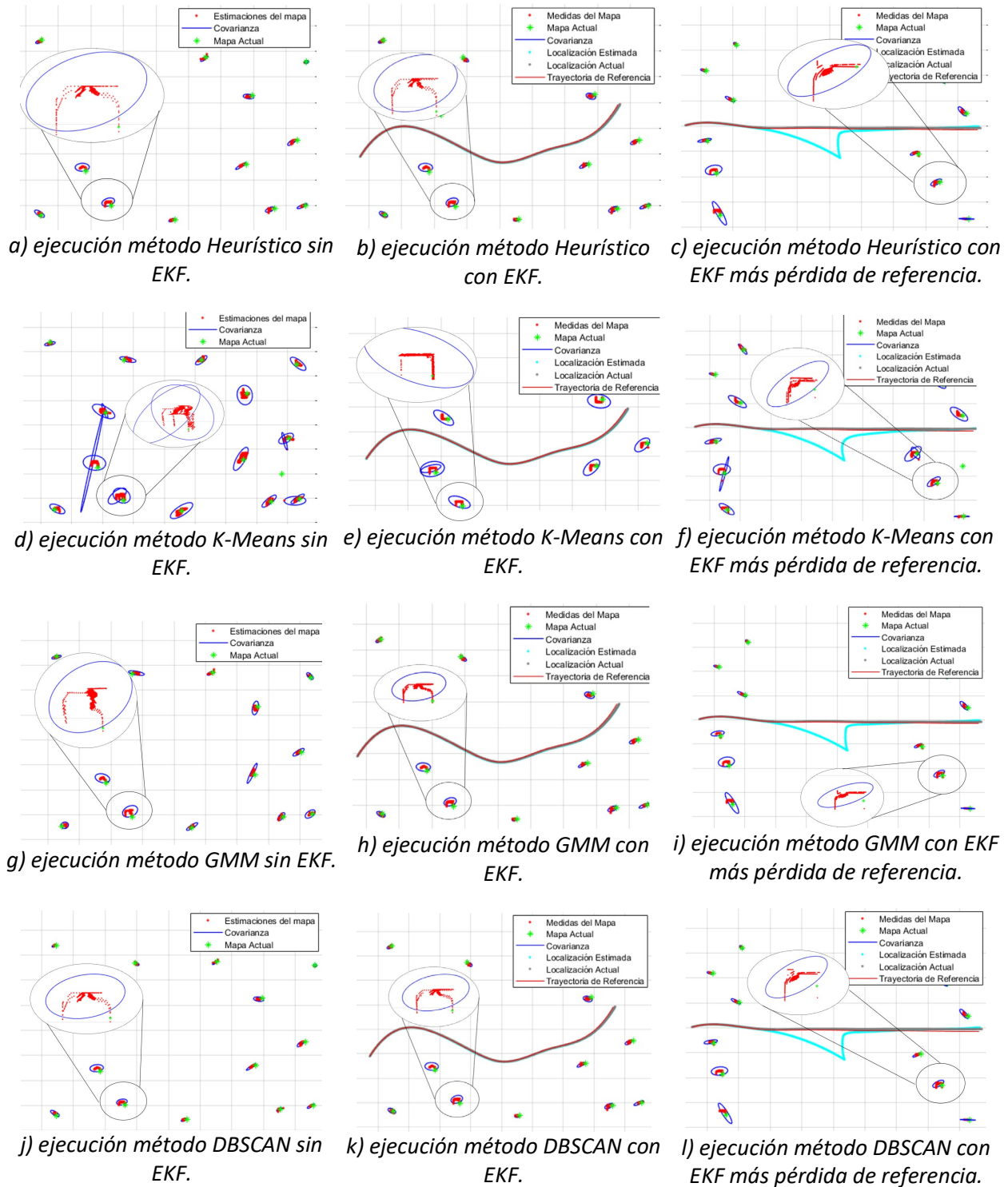
**Figura 4.** Error acumulado entre la trayectoria estimada y la trayectoria predefinida

Uno de los ejemplos utilizados en la prueba anterior puede visualizarse en Figura 5. En esta figura, los puntos de color lila corresponden a medidas del sensor LiDAR, los puntos verdes consisten en la región donde se caracteriza el objeto dada las mediciones, la región gris con bordes rojos es el área de detección efectiva del sensor de rango, y las elipses azules representan la forma geométrica de la matriz de covarianza asociada a la posición más probable en la que se encuentra un objeto. La prueba aquí consiste en utilizar solo un método de clusterización para la identificación del mapa, descuidando la localización simultánea. En específico, se utilizó el método de K-Means para la obtención de la región más probable donde se encuentren los puntos guía o a su vez obstáculos que están presentes en el espacio de trabajo del robot. Considerando la odometría del robot, se puede analizar por simple inspección que la incertidumbre del posicionamiento de los objetos en escena contempla mayor incertidumbre a medida que menor cantidad de estimaciones son posibles o postes visibles en la escena. Dado que el sensor LiDAR no detecta los objetos en la escena y a el error de odometría se incrementa, la región que describe la zona navegable del robot se incrementa durante la prueba, mostrando la efectividad del método aun cuando el error de odometría se incrementa. Sin embargo, para solucionar este problema, el método de actualización de forma iterativa de la matriz de covarianza es necesario para evitar la propagación del error asociado a la odometría del robot.



**Figura 5.** Resultados obtenidos de localización utilizando solo el método de clusterización K-Means. La representación geométrica de la matriz de covarianza puede graficarse en forma de elipses, cuya región se puede visualizar más grande a medida que objetos adicionales en la escena se encuentran en la escena. El incremento en la incertidumbre de la ubicación de los obstáculos en el mapa se debe a que el error asociado a la odometría se acumula mientras se sigue una trayectoria predefinida.

Con el objetivo de proveer una comparativa de los resultados obtenidos con y sin método de actualización de la matriz de covarianza con el enfoque propuesto EFK, la Figura 6 muestra los resultados obtenidos con los métodos propuestos de clusterización en combinación con el filtro EFK. En la segunda columna de la Figura 6 se muestra la ejecución de los métodos propuestos cuando se conjugan con EKF y el robot no pierde la referencia en ningún instante de tiempo. En la columna 3 de la Figura 6 indica la ejecución de los métodos propuestos cuando se conjugan con EKF incluyendo en el experimento la pérdida de localización del robot. Como se observa en las ilustraciones de la tercera columna de la Figura 6 el robot durante un periodo de tiempo no sigue la trayectoria predefinida. Esto se debe a que el sensor LiDAR no detecta objetos en el mapa de navegabilidad. Cuando el sensor LiDAR detecta objetos en el mapa nuevamente, el robot recupera la localización correcta.



**Figura 6.** Resultados obtenidos de SLAM al simular el movimiento autónomo de un vehículo terrestre. Cada fila representa la ejecución de cada método en tres dinámicas diferentes y cada columna representa la ejecución de cada dinámica empezando por la ejecución del sistema sin EKF, la segunda columna representa la ejecución con EKF y la última columna representa la ejecución del sistema con EKF y con pérdida de objetos de referencia.

## CONCLUSIONES

En este artículo se presenta el uso de técnicas de aprendizaje no supervisado más Filtro de Kalman Extendido para el diseño de un sistema de localización y mapeo simultaneo de robots móviles. Los enfoques utilizados emplean el método heurístico, K-means, GMM y DBSCAN para tratar nubes de puntos generada por un sensor de rango LiDAR. Además de utilizar dichos métodos, se complementa las técnicas propuestas con una metodología híbrida que permite a la localización automática actualizar la matriz e covarianza asociada a la localización de los objetos en escena mediante el Filtro de Kalman Extendido EKF. La estimación de la posición del robot en escena es precisa y muestra la capacidad de recuperar la mediada frente a pérdidas de objetos referencia dentro del mapa de navegación. Los métodos propuestos son válidos para generar realimentación a sistemas de control de navegación autónoma de vehículos terrestres que presentan errores de posicionamiento adicional, como en la odometría.

Las técnicas empleadas en la localización presentan, en particular, menor tiempo de ejecución en la tarea de localización y mapeo utilizando DBSCAN en comparación a los otros enfoques de clusterización. Esto debido a que este método de aprendizaje no supervisado presenta menos limitaciones de cálculo, como la cantidad de clusters y sus centroides que son prescritos antes de las pruebas. Asimismo, el trabajo presenta una alternativa diferente para fusionar los métodos de aprendizaje no supervisado y EKF para la estimación de la pose del robot en cada instante de tiempo, demostrando que es posible localizar el robot y mapear el mismo dentro del ambiente de navegación.

Como trabajo futuro se busca el implementar los enfoques propuestos sobre un sistema real dentro de la agricultura o minería, donde la señal de GPS puede llegar a ser inhibida, ocluida o distorsionada por factores externos o inherentes al modelo del proceso.

## RECONOCIMIENTO

Esta investigación fue soportada y financiada por el Departamento de Ingeniería de Sistemas y Computación de la Universidad Católica del Norte bajo el proyecto 202203010029-VRIDT-UCN.

## REFERENCIAS

- [1] Pragya Agrawal, Asif Iqbal, Brittney Russell, Mehrnaz Kh. Hazrati, Vinay Kashyap, and Farshad Akhbari, "PCE-SLAM: A Real-time Simultaneous Localization and Mapping using LiDAR data," IEEE Intelligent Vehicles Symposium (IV), Los Angeles, CA, USA: IEEE, 2017, pp. 1752–1757, doi: 10.1109/IVS.2017.7995960.
- [2] H. Yu, H. Zhu, and F. Huang, "Visual Simultaneous Localization and Mapping (SLAM) Based on Blurred Image Detection," Journal of Intelligent and Robotic Systems: Theory and Applications, vol. 103, no. 1, Sep. 2021, doi: 10.1007/s10846-021-01456-5.

- [3] W. Wei, X. Zhu, and Y. Wang, "Novel robust simultaneous localization and mapping for long-term autonomous robots," *Frontiers of Information Technology & Electronic Engineering*, vol. 23, no. 2, pp. 234–245, Feb. 2022, doi: 10.1631/FITEE.2000358.
- [4] W. F. A. Wan Aasim, M. Okasha, and W. F. Faris, "Real-Time Artificial Intelligence Based Visual Simultaneous Localization and Mapping in Dynamic Environments – a Review," *J Intell Robot Syst*, vol. 105, no. 1, p. 15, May 2022, doi: 10.1007/s10846-022-01643-y.
- [5] P. Karkus, S. Cai, and D. Hsu, "Differentiable SLAM-net: Learning Particle SLAM for Visual Navigation," May 2021, [Online]. Available: <http://arxiv.org/abs/2105.07593>
- [6] J. Kim, Y. Cho, and J. Kim, "Vehicle Localization in Urban Environment Using a 2D Online Map with Building Outlines," in *2018 15th International Conference on Ubiquitous Robots (UR)*, IEEE, Jun. 2018, pp. 586–590. doi: 10.1109/URAI.2018.8441821.
- [7] M. Yokozuka, K. Koide, S. Oishi, and A. Banno, "LiTAMIN: LiDAR-based Tracking And Mapping by Stabilized ICP for Geometry Approximation with Normal Distributions," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, Oct. 2020, pp. 5143–5150. doi: 10.1109/IROS45743.2020.9341341.
- [8] R. Tiar, M. Lakrouf, and O. Azouaoui, "Fast ICP-SLAM for a bi-steerable mobile robot in large environments," in *2015 International Conference on Advanced Robotics (ICAR)*, IEEE, Jul. 2015, pp. 611–616. doi: 10.1109/ICAR.2015.7251519.
- [9] C. Zhang, J. Tang, J. Zhou, and S. Wei, "Singular value decomposition compressed ghost imaging," *Applied Physics B*, vol. 128, no. 3, p. 47, Mar. 2022, doi: 10.1007/s00340-022-07768-0.
- [10] J. M. Arevalillo and H. Navarro, "A stochastic ordering based on the canonical transformation of skew-normal vectors," *TEST*, vol. 28, no. 2, pp. 475–498, Jun. 2019, doi: 10.1007/s11749-018-0583-5.
- [11] Ł. Sobczak, K. Filus, A. Domański, and J. Domańska, "LiDAR Point Cloud Generation for SLAM Algorithm Evaluation," *Sensors*, vol. 21, no. 10, p. 3313, May 2021, doi: 10.3390/s21103313.
- [12] J. Wen, C. Qian, J. Tang, H. Liu, W. Ye, and X. Fan, "2D LiDAR SLAM Back-End Optimization with Control Network Constraint for Mobile Mapping," *Sensors*, vol. 18, no. 11, p. 3668, Oct. 2018, doi: 10.3390/s18113668.
- [13] D. Matti, H. K. Ekenel, and J.-P. Thiran, "Combining LiDAR Space Clustering and Convolutional Neural Networks for Pedestrian Detection," Oct. 2017, [Online]. Available: <http://arxiv.org/abs/1710.06160>
- [14] M. U. Khan, S. A. A. Zaidi, A. Ishtiaq, S. U. R. Bukhari, S. Samer, and A. Farman, "A Comparative Survey of LiDAR-SLAM and LiDAR based Sensor Technologies," in *2021 Mohammad Ali Jinnah University International Conference on Computing (MAJICC)*, IEEE, Jul. 2021, pp. 1–8. doi: 10.1109/MAJICC53071.2021.9526266.

- [15] V. A. Rosas-Cervantes, Q.-D. Hoang, S.-G. Lee, and J.-H. Choi, "Multi-Robot 2.5D Localization and Mapping Using a Monte Carlo Algorithm on a Multi-Level Surface," *Sensors*, vol. 21, no. 13, p. 4588, Jul. 2021, doi: 10.3390/s21134588.
- [16] B. Sun and P. Mordohai, "Oriented Point Sampling for Plane Detection in Unorganized Point Clouds," May 2019, [Online]. Available: <http://arxiv.org/abs/1905.02553>
- [17] A. Rubinstein and T. Erez, "Autonomous robot for tunnel mapping," in 2016 IEEE International Conference on the Science of Electrical Engineering (ICSEE), IEEE, Nov. 2016, pp. 1–4. doi: 10.1109/ICSEE.2016.7806111.
- [18] R. W. Wolcott and R. M. Eustice, "Robust LIDAR localization using multiresolution Gaussian mixture maps for autonomous driving," *Int J Rob Res*, vol. 36, no. 3, pp. 292–319, Mar. 2017, doi: 10.1177/0278364917696568.
- [19] C. Pang, Y. Tan, S. Li, Y. Li, B. Ji, and R. Song, "Low-cost and High-accuracy LIDAR SLAM for Large Outdoor Scenarios," in 2019 IEEE International Conference on Real-time Computing and Robotics (RCAR), IEEE, Ago. 2019, pp. 868–873. doi: 10.1109/RCAR47638.2019.9044147.
- [20] Md. Zubair, MD. A. Iqbal, A. Shil, M. J. M. Chowdhury, M. A. Moni, and I. H. Sarker, "An Improved K-means Clustering Algorithm Towards an Efficient Data-Driven Modeling," *Annals of Data Science*, Jun. 2022, doi: 10.1007/s40745-022-00428-2.
- [21] G. Zickert and C. E. Yarman, "Gaussian mixture model decomposition of multivariate signals," *Signal Image Video Process*, vol. 16, no. 2, pp. 429–436, Mar. 2022, doi: 10.1007/s11760-021-01961-y.
- [22] J.-H. Kim, J.-H. Choi, K.-H. Yoo, and A. Nasridinov, "AA-DBSCAN: an approximate adaptive DBSCAN for finding clusters with varying densities," *J Supercomput*, vol. 75, no. 1, pp. 142–169, Ene. 2019, doi: 10.1007/s11227-018-2380-z.
- [23] S. Wang et al., "A Lightweight Localization Strategy for LiDAR-Guided Autonomous Robots with Artificial Landmarks," *Sensors*, vol. 21, no. 13, p. 4479, Jun. 2021, doi: 10.3390/s21134479.
- [24] P. Slowak and P. Kaniewski, "LIDAR-based SLAM implementation using Kalman filter," in *Radioelectronic Systems Conference 2019*, P. Kaniewski and J. Matuszewski, Eds., SPIE, Feb. 2020, p. 5. doi: 10.1117/12.2564818.
- [25] A. Iqbal and N. R. Gans, "Data Association and Localization of Classified Objects in Visual SLAM," *J Intell Robot Syst*, vol. 100, no. 1, pp. 113–130, Oct. 2020, doi: 10.1007/s10846-020-01189-x.
- [26] R. Martins, D. Bersan, M. F. M. Campos, and E. R. Nascimento, "Extending Maps with Semantic and Contextual Object Information for Robot Navigation: a Learning-Based Framework Using Visual and Depth Cues," *J Intell Robot Syst*, vol. 99, no. 3–4, pp. 555–569, Sep. 2020, doi: 10.1007/s10846-019-01136-5.

[27] T. Pire, J. Corti, and G. Grinblat, "Online Object Detection and Localization on Stereo Visual SLAM System," *J Intell Robot Syst*, vol. 98, no. 2, pp. 377–386, May 2020, doi: 10.1007/s10846-019-01074-2.

[28] C. Benedek, A. Majdik, B. Nagy, Z. Rozsa, and T. Sziranyi, "Positioning and perception in LIDAR point clouds," *Digit Signal Process*, vol. 119, p. 103193, Dic. 2021, doi: 10.1016/j.dsp.2021.103193.

[29] X. Sun, H. Ma, Y. Sun, and M. Liu, "A Novel Point Cloud Compression Algorithm Based on Clustering," *IEEE Robot Autom Lett*, vol. 4, no. 2, pp. 2132–2139, Abr. 2019, doi: 10.1109/LRA.2019.2900747.