Implementación de sistema de navegación autónomo en robot móvil experimental para reconstrucción y exploración de entornos desconocidos

Carlos A. Velásquez Hernández [1], John J. Chávez Chávez [2], Ernesto Córdoba Nieto [3]

(1) cvelasquez@unal.edu.co

Departamento de Eléctrica y Electrónica
Universidad Nacional de Colombia

(2) jchavezc@unal.edu.co
(3) ecordoban@unal.edu.co

Departamento de Mecánica y Mecatrónica
Universidad Nacional de Colombia

Implementación de sistema de navegación autónomo en robot móvil experimental para reconstrucción y exploración de entornos desconocidos

Palabras clave:
SLAM; robótica móvil; evasión obstáculos; navegación autónoma

RESUMEN

La implementación de un sistema de navegación autónomo en un robot móvil experimental de tracción diferencial permite dar autonomía a dicho agente para reconstruir y explorar su propio entono, sin necesidad de establecer condiciones iniciales para su operación autónoma. Para ello, se hace necesario resolver el problema de mapeo y localización simultánea (SLAM), tema de amplio estudio en la comunidad académica y científica relacionada con el área de la navegación de robots móviles. Además, resolver el problema de navegación, enmarcado en la generación autónoma de trayectorias y evasión de obstáculos dinámicos y estáticos, es un reto abordado y resuelto en este trabajo investigativo. Adicionalmente, este sistema de navegación desarrollado es implementado en un robot de tracción diferencial dada su complejidad en cuanto al control de movimiento respecto a un robot omnidireccional.

Basados en técnicas SLAM y de navegación existentes, el control de movimiento, la generación de trayectorias suavizadas y la evasión de obstáculos son los temas de mayor aporte al campo de la navegación autónoma de robots móviles dado su satisfactoria implementación en un robot móvil experimental de tracción diferencial.

I. INTRODUCCION

El problema de localización y posicionamiento de robots móviles en un ambiente desconocido y cambiante es tema de amplia investigación en la robótica actual. Estas investigaciones son impulsadas por las potenciales aplicaciones que puede tener un vehículo robotizado con la capacidad de navegar automáticamente en entornos desconocidos. Transporte automatizado de piezas y almacenaje en la industria, transporte de pacientes o medicamentos en medicina, recolección de cosechas y transporte de insumos en agroindustria, transporte de equipajes en aeropuertos, búsqueda en áreas de desastre, exploración terrestre y acuática [1], son sólo algunas de las aplicaciones que tienen este tipo de robots.

En la actualidad se han desarrollado varias técnicas aplicables a la navegación y posicionamiento de plataformas robóticas móviles en entornos dinámicos [2], estáticos, interiores [2] y exteriores [3]. Estas técnicas se basan en la localización y el mapeo simultáneo SLAM (Simultaneous Localization and Mapping), que es el proceso de obtener una representación del entorno a la vez que se usa dicha representación para localizar al vehículo. La mayoría de métodos de SLAM se centran en el problema de mapeo y localización para un único robot, sin embargo las técnicas aplicables a grupos de vehículos robotizados están empezando a ser formuladas [4-5].

Es por esto que en este artículo se expone el desarrollo, programación y validación de un sistema de navegación autónomo para un robot móvil de tracción diferencial usando de la técnica HectorSLAM [6] como algoritmo base de localización y mapeo de entornos, y un sistema de navegación basado en desarrollos open-source ampliamente conocidos en la plataforma robótica ROS. La selección de estas técnicas se realizó basado en estudios previos que ratifican el buen desempeño de estos. En [7], por ejemplo, se hace un análisis detallado de la técnica SLAM seleccionada sobre otras técnicas SLAM existentes como Gmapping, RGB-D SLAM, KartoSLAM, entre otras. Dada la poca precisión de la odometría estimada para robots móviles de transmisión diferencial, usar una técnica dependiente de la sensoria interna del robot puede conllevar a estimaciones pocos precisas y consistentes con su entorno. Adicionalmente, el sistema de navegación basa el cálculo de sus trayectorias y la evasión de obstáculos estáticos de acuerdo a un mapa de costos que evalúa el riesgo de transitar por zonas circundantes a los obstáculos percibidos por el robot, teniendo en cuenta el área que este (el robot) ocupa y la manera como se desplaza en el entorno (control de movimiento).

El presente documento está organizado como sigue: en la Sección 2 se contextualiza y presenta los fundamentos básicos de la técnica de localización y mapeo simultáneo (SLAM). En la Sección 3 se expone el Sistema de Navegación Implementado en los robots móviles de prueba. Seguido a esto se expone en la Sección 4 una sección de Desempeño y Eficiencia del sistema de navegación, en esta sección se exponen cálculos y las características más relevantes del sistema.

Finalmente, en la Sección 5 se discuten y exponen los resultados y conclusiones más relevantes de lo desarrollado hasta el momento en este proyecto investigativo.

II. LOCALIZACIÓN Y MAPEO SIMULTÁNEO (SLAM)

El mapeo y localización simultánea o SLAM es el problema de obtener una estimación del entorno a la vez que se localiza en el mismo [8].

Fundamentalmente SLAM sigue una secuencia de pasos para garantizar un mapeo y un estimado de la posición lo más preciso posible. Primero, el algoritmo calcula su posición mediante odometría, esta es considerada una primera aproximación cruda o gruesa, debido a la gran incertidumbre asociada a dicha medida. Luego se procede a realizar la toma de una muestra del entorno para refinar esa primera estimación de la pose del robot y se estima el mapa alrededor del agente. Seguido a esto, se predice la posición del robot una vez se ha introducido un paso de control (movimiento). Se calcula por odometría donde se encuentra nuevamente el

robot y finalmente este debe poder re-observar la muestra o landmark que vio en el primer paso con el fin de actualizar y refinar la pose y mapa del robot en el entorno.

Es evidente, que el paso de re-observar las landmarks es fundamental para el algoritmo de SLAM, ya que de esta manera el robot puede estimar cuanto se movió respecto a la landmark reconocida y con esto refinar sus movimientos y disminuir la incertidumbre asociada a su posición. Si el robot detecta una nueva landmark en su muestreo y cumple con las condiciones para ser considera como referencia, es incluida dentro del mapa creado por el robot y adicionalmente es usada para refinar aún más la estimación de su pose durante su exploración.

Todos los cálculos realizados por SLAM, fundamentalmente se basan en métodos probabilísticos, es decir, calculan la probabilidad de la presencia de un objeto o no luego de observarlo una y otra vez para usarlo como landmark. Usan estos métodos también para estimar si su posición real y actual ha sido consistente con las medidas y cálculos hechos con posiciones anteriores.

A. Técnicas SLAM

Las técnicas en SLAM pueden ser clasificadas en dos grupos, los métodos basados en filtros y los métodos basados en correspondencias. Las aproximaciones basadas en filtros abordan el SLAM como un problema probabilístico y aplican filtros para estimar un vector de estado que contiene la pose (dupla de posición y orientación) del vehículo y la posición de los puntos de referencia (landmark). Los filtros más aplicados son el filtro extendido de Kalman (EKF), reportado en la literatura como el más antiguo e implementado [9], filtros de partículas (PF), UKF (Unscented Kalman Filter) y filtros de información extendida.

- ***Hector SLAM [6]***: técnica de SLAM basada en filtro extendido de Kalman para estimación 3D del mapa y posición del robot en cada instante. Hector SLAM basa su proceso de localización en tiempo real haciendo uso exclusivamente de la alta frecuencia de muestreo con los sensores de percepción para procesar y estimar la pose del robot. Esta técnica no tiene en cuenta la estimación por odometría, debido a que los autores consideran que la baja precisión de este proceso le resta exactitud y desempeño al algoritmo.

Al contar con esta novedad, este algoritmo es muy usado para robots UAV (aviones no tripulados en español).

- ***KartoSLAM:*** técnica de SLAM basada en grafos, desarrollada por la empresa SRI International's Karto Robotics y liberada su versión libre en ROS (Robot System Operating). Fundamentalmente difiere de las otras técnicas, dado que representa el mapa captado a través de nodos y arcos que los conectan. Cada nodo representa una Pose del robot incluyendo las medidas de los sensores y la incertidumbre asociada a dicha información, los arcos o conexiones entre nodos representan las posibles trayectorias de robot de un punto a otro, al igual que representan las restricciones de espacio por la cuales dicho robot puede moverse.

Esta aproximación permite desarrollar el mapeo y localización del robot en grandes superficies, ya que un nuevo nodo es almacenado cuando su información ha sido verificada y consistente con el entorno (ese nuevo nodo debe poseer conexiones con los nodos anteriormente procesados). Sin embargo, el consumo de memoria para almacenar su información y el tiempo de computación al tener un grafo tan grande, son sus principales limitantes.

KartoSLAM ha demostrado ser una aproximación de SLAM precisa y escalable dentro de las técnicas de SLAM conocidas.

- ***FastSLAM:*** la técnica más ampliamente conocida en SLAM basada en filtro de partículas es FastSLAM. Divide el problema de SLAM en dos partes, la primera relacionada con la posición del vehículo y la

segunda con la posición de marcas de referencia dentro del entorno. En esta aproximación se utiliza un filtro de partículas (Rao-Blackwellized) para estimar la pose del vehículo y EKF para estimar la posición de los puntos de referencia. Esta formulación hace al algoritmo más robusto y le da la capacidad de manejar modelos de movimiento del vehículo no lineales. A pesar de sus grandes ventajas FastSlam también tiene inconvenientes siendo el más significativo su degeneración con el tiempo (se relaciona con la pérdida de la diversidad de las partículas), sin embargo se han reportado soluciones eficientes a tales inconvenientes, lo que ha permitido implementaciones exitosas de FastSlam como el caso reportado en [10].

- ***Técnicas Basadas en Correspondencia:*** Los métodos basados en correspondencias toman dos conjuntos de información consecutivos, generalmente datos 3D, obtenida de los sensores e intentan estimar una transformación que permita asociar cada conjunto de datos para obtener el mapa del entorno junto con la pose del vehículo, estos métodos son ampliamente utilizadas en las aproximaciones basadas en visión. Se han reportado implementaciones con cámaras monoculares [11], estéreo [12] y cámaras infrarrojas de profundidad (RGB-D) [13]. Uno de los mecanismos de asociación de datos más utilizado en este tipo de métodos es ICP (Iterative Closest Point) que es una técnica que permite la alineación de conjuntos de datos [13].

  La técnica más representativa de este tipo de técnicas SLAM es conocida como RGB-DSLAM

Basados en los resultados expuestos y analizados en [7], las técnicas que presentan un mejor desempeño en términos de consistente y velocidad del algoritmo es Hector SLAM y Gmapping. Sin embargo, dada la limitante para el cálculo del paso de odometría necesario para la técnica Gmapping en los robots móviles disponibles del Laboratorio, se decide seleccionar la técnica Hector SLAM basado en el estudio analizado y en la limitante expuesta.

B. Hector Mapping

Hector Mapping (Hector SLAM) es un sistema flexible y escalable que aborda el problema de SLAM, realiza mapeo y localización 2D con alta precisión y bajo costo computacional. El sistema puede ser usado para SLAM en pequeños entornos. Hector Mapping utiliza un arreglo regular de cuadrados para representar el mapa, cada cuadrado tiene un estado que dependerá de la probabilidad de ocupación calculada por el algoritmo [6]. Para estimar la probabilidad de ocupación el sistema utiliza las siguientes definiciones y expresiones [6].

Hector Mapping realiza correspondencia de datos (alineación de mediciones realizadas en el instante t con mediciones realizadas en instantes anteriores), usando una aproximación Gauss-Newton donde se busca la transformación rígida:

Que minimiza:

$$M = (p_x, p_y, \psi)^T \tag{1}$$

$$\xi = argmin = \sum_{i=1}^{n} [1 - M(S_i(M))]^2 \tag{2}$$

Las expresiones (1) y (2) permiten encontrar la mejor transformación que lleva a la mejor alineación de mediciones láser con el mapa actual. En estas expresiones $S_i(M)$ representa las coordenadas del mundo de un punto escaneado que es función de la pose del robot (en las coordenadas del mundo) como se indica en la expresión (3).

$$S_i(M) = (\cos(\psi) - \sin(\psi)\ \sin(\psi)\ \cos(\psi))(s_{i,x}\ s_{i,y}) + (p_x\ p_y) \tag{3}$$

La función $M(S_i(M))$ representa el valor del mapa en la coordenada dada por $S_i(M)$.

Hector Mapping está disponible como paquete de librerías Open Source para el framework ROS y fue utilizado para la implementación del sistema descrito en la sección 3.

III. SISTEMA DE NAVEGACIÓN AUTÓNOMA IMPLEMENTADO

El sistema de navegación se concibe con la idea de dotar al robot con la capacidad para ir a puntos específicos del entorno o si se desea, a cualquier punto del entorno de trabajo. Para esto, el robot debe conocer el entorno en el que se mueve, evadir obstáculos estáticos mediante la generación de una trayectoria autónoma y segura al punto de navegación establecido. Adicionalmente, debe ser capaz de detectar cualquier obstáculo dinámico no representado en su mapa de navegación.

Con toda esta información recopilada, el algoritmo de navegación debe generar los comandos de movimientos idóneos para que el robot pueda navegar segura y eficazmente en el entorno donde se encuentra. De esta manera se conforma y desarrolla el sistema de navegación programado para los robots móviles del Laboratorio LabFabEx de la Universidad Nacional de Colombia, basados en el algoritmo de navegación Hector Navegation de los mismos desarrolladores de la técnica de localización y mapeo simultaneo Hector SLAM.

A. Mapas para Navegación y Exploración de Entornos

Los mapas o representaciones del entorno son importantes en la robótica móvil debido a su obligatorio uso para tareas de navegación, actualización y exploración del entorno mismo.

En la navegación de robots móviles puede ser muy útil el uso de planos de los espacios donde se supone navegará y actuará el robot, sin embargo estos mapas o planos tienen la gran limitante de que muchas veces las edificaciones no tienen las dimensiones ni la disposición final que se encuentra en los planos, además, los mapas o planos de las edificaciones no tienen presente ni en cuenta el mobiliario o disposición interna de los mismos dentro de los espacios [14].

B. Algoritmo de Navegación Implementado

El algoritmo de navegación desarrollado se basa en el stack de ROS Hector Navigation. Este stack es un conjunto de algoritmos capaz de permitir a un robot móvil navegar y explorar autónomamente un entorno desconocido mediante la asignación de un punto de exploración de frontera del mapa, generación y asignación de una trayectoria, evasión de obstáculos, y escaneo y reconstrucción del entorno para robots móviles estándar. Sin embargo, su implementación en un robot de tracción diferencial conllevó a una reprogramación del algoritmo de generación y seguimiento de trayectorias, así como el algoritmo de evasión de obstáculos dinámicos y estáticos, dado que el robot AGV de tracción diferencial utilizado posee las restricciones mecánicas presentes en este tipo de robots (radio de giro diferente de cero, deslizamiento en rotación y traslación, esquema de control complejo, entre otros).

El sistema de navegación desarrollado se compone de diferentes procedimientos, los cuales individualmente se encargan de tareas específicas pero que en su conjunto permiten que el robot móvil AGV-UN navegue en su entorno. El algoritmo de navegación inicia con una Exploración de Fronteras de su entorno (Algoritmo basado en el stack Hector Navigation [15]). Seguido a este procedimiento, el algoritmo Generación de Trayectoria obtiene un camino el cual es suavizado y publicado para que el algoritmo de Seguimiento de Trayectoria genere a través del esquema de Control de Movimiento los comandos u órdenes para que el robot se mueva a través de la trayectoria dada. Adicionalmente, un algoritmo de Evasión de Obstáculos verifica que el robot evada y evite choques con obstáculos presentes en su entorno o con obstáculos dinámicos.

Como desarrollo y aporte de este trabajo de investigación se resalta el algoritmo de generación de trayectorias que genera y refina una trayectoria de navegación válida y segura para el robot.

Una vez el algoritmo de exploración de fronteras obtiene un vector de posibles objetivos a explorar, toma la posición actual del robot (Pose) respecto al mapa que ha venido reconstruyendo y calcula las trayectorias más viables priorizando la distancia. Para todos los posibles objetivos de exploración el algoritmo le asigna un peso a cada uno de

acuerdo a la distancia recorrida y de acuerdo al riesgo de transitar por zonas cercanas a objetivos. Este último costo se conoce como mapa de costos y establece un radio de seguridad para que el robot evada eficazmente los obstáculos detectados dentro de su entorno. A medida que va cumpliendo con un objetivo inicia con el siguiente, sin embargo durante el seguimiento de una trayectoria, el robot mapea el entorno a su alrededor y actualiza el vector de poses objetivos, ya que una o varias poses objetivos pueden haber sido descubiertas.

Por otro lado, este algoritmo de trayectorias generadas en una primera etapa crea la trayectoria más segura y corta entre la pose del robot y la pose objetivo, sin embargo, dado lo variable del entorno, el algoritmo genera trayectorias que pueden ir cercanas al radio de seguridad del mapa de costos y por ende generar trayectorias con segmentos muy cortos que no son válidos para nuestro robot AGV-UN dado que el esquema de control de movimiento concebido, podría generar comandos de movimiento muy bruscos para el robot, y este, por sus limitantes mecánicas, no puede responder de manera adecuada. Para solucionar este inconveniente, el algoritmo de trayectorias tiene una etapa de refinamiento que elimina segmentos inferiores a 28 cms (largo del robot) y ángulos entre segmentos superiores a 70°. Con estas condiciones implementadas, los resultados en cuanto a generación de trayectorias suavizadas para el robot AGV-UN muestran grandes diferencias entre la trayectoria original y la trayectoria final enviada al algoritmo de seguimiento de trayectorias y control de movimiento del robot.

De las Fig. 1 y Fig. 2, las trayectorias cumplen con los perfiles de desplazamiento requeridos para este tipo de robots. Los resultados en cuanto a suavizado, control y seguimiento de la trayectoria mostraron un cambio radical en el comportamiento del vehículo pues las incidencias de pérdida de la trayectoria bajaron de 25% a un 5%, lo que quiere decir que el robot puede seguir trayectorias de manera autónoma y segura hacia objetivos trazados.

Los resultados obtenidos demuestran que el algoritmo de generación de trayectorias desarrollado es una aproximación válida para esta clase de robots móviles.

Fig 1. Corrección de trayectoria obtenida con refinamiento de algoritmo (camino Rojo). El camino original se denota por el color Naranja. Tal como se ve, la primera corrección (camino Verde) reduce significativamente los tramos cortos de la trayectoria original. En esta figura se denota un tramo en S de la trayectoria la cual es suavizada en el camino rojo, camino refinado por el algoritmo

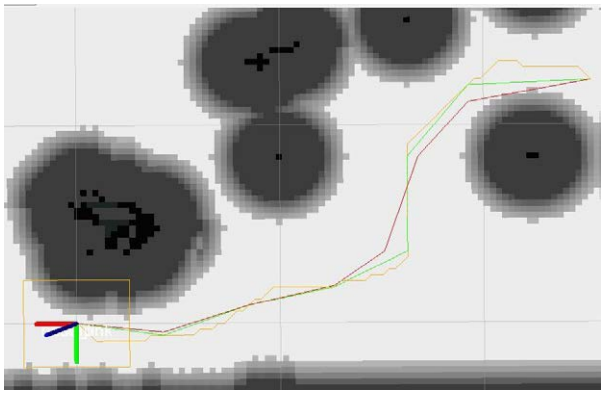
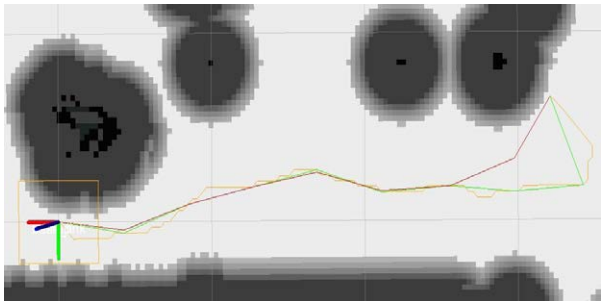


Fig 2. Ángulos producidos por el algoritmo de generación de trayectorias inicial (Camino Verde) pueden ser muy grandes y por ende provocar que el esquema de control no pueda generar los comandos de movimientos apropiados para un robot móvil de tracción diferencial. En esta figura, el final del tramo verde presenta un cambio abrupto en su ángulo de giro, el cual es suavizado por el refinamiento hecho al algoritmo de generación de trayectorias (Camino Rojo).



C. Control de Movimiento

El control de movimiento implementado es un control de posición del robot sobre toda la trayectoria trazada y generada. Una vez es obtenida del módulo de generación de trayectorias, el módulo de seguimiento de trayectorias y control de movimiento recibe la información y comienza a generar los perfiles de velocidad trapezoidal del robot a través de comandos PWM para cada motor.

El presente esquema de control es implementación y desarrollo propio de los investigadores de este trabajo. Su concepción y planteamiento se produjo luego de realizar pruebas de movimientos experimentales con el robot móvil AGV-UN.

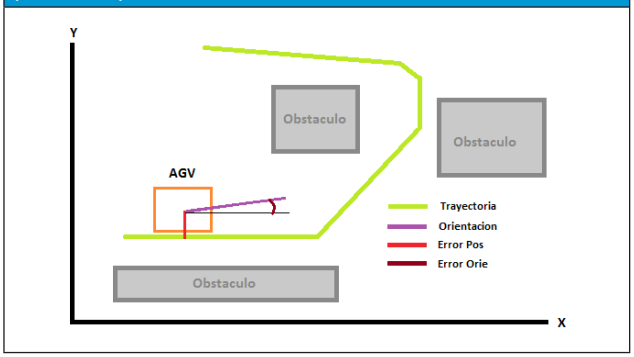
A. Esquema de Control

La estrategia utilizada se basa en el cálculo de una función de error que depende del desfase en orientación y translación del robot, Fig. 3. Dicha función de error en todo instante busca ser minimizada con el fin de garantizar el posicionamiento adecuado del robot durante toda la trayectoria.

$$Ferr = \alpha e_{tras} + \beta e_{rot} \quad (4)$$

Dicha función de error cuenta con dos pesos (α,β) que experimental fueron calculados para el buen desempeño del robot AGV-UN durante sus movimiento dirigidos: dado que los tramos de la trayectorias son segmentos en línea recta, el peso es levemente mayor al peso β, dado que para este esquema de control es de mayor importancia eliminar el error en traslación pero compensar los desvíos generados por esta corrección con un control en orientación del robot. De esta manera se determinaron los valores de dichos pesos los cuales para este robot móvil experimental están programados en α = 0.57 y β = 0.43. Su sumatoria es igual a 1.

Fig 3. Error de Traslación y Orientación del Robot AGV-UN respecto a la trayectoria a seguir. Como se puede apreciar durante el control de la posición, el error de traslación es el de mayor peso debido a que el robot puede chocarse con un obstáculo o simplemente perder su referencia de posición óptima.



El error de posición máximo permitido es de +/-10mm, teniendo en cuenta que la precisión en posición del sistema de localización y mapeo (SLAM) es +/-6mm, por ende, un error en posición no puede ser menor a este último parámetro; sin embargo, el esquema de control, ayudado por la sensorica interna del robot (diferente al laser de navegación), intentará disminuir este error una vez el robot haya finalizado su movimiento. Por otro lado, el error en orientación se estableció en +/-15 grados, debido a que la orientación del robot también presenta una incertidumbre de +/- 10 grados.

Finalmente, el control de movimiento es capaz de controlar los cambios de orientación por cambio de tramo o segmento de la trayectoria que sigue. Para esta parte del control, se establece un radio de cercanía al punto donde se presenta el cambio de orientación (nuevo tramo de la trayectoria). Una vez el robot entra en este radio, el control cambia su referencia en orientación para seguir con el nuevo tramo de la trayectoria. Para esto, la función de error depende del error en orientación del robot.

$$Ferr = \beta e_{rot} \quad (5)$$

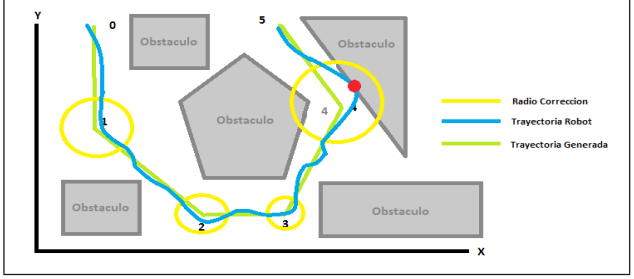
En este caso, el peso β depende de la distancia del robot hacia el nuevo tramo, es decir, cuando el robot entra en el radio de corrección establecido este peso es 0 y crece a medida que se acerca al nuevo tramo, siendo el peso 1 cuando el robot está sobre el nuevo tramo. Una vez el robot logra corregir su orientación con respecto a su nueva trayectoria, la función de error o de control se establece como en (4).

Este radio de corrección, que se establece para saber cuándo iniciar con el cambio de referencia de orientación del nuevo tramo, se establece de tal manera que la corrección sea la suficiente como para cambiar la orientación del robot sin chocarlo con los obstáculos presentes en el entorno. Por ende, un radio de corrección muy grande puede desviar al robot de tal manera que este puede chocarse con un obstáculo cercano, pero un radio de corrección muy pequeño no le da al control el tiempo suficiente como para llevarlo a la nueva referencia sin salirse de los rangos máximos de error permitidos para orientación y translación. Por lo tanto, el cálculo de este parámetro se estableció de manera experimental, debido a la diversidad de trayectorias generadas

a lo largo del entorno de prueba que constantemente cambiaba.

En la Fig. 4 se muestran distintos casos donde dependiendo del radio de corrección establecido (color amarillo), el control de movimiento genera los comandos de velocidad para cambiar la orientación del robot respecto al siguiente tramo de la trayectoria (línea azul es la ruta real seguida por el robot, mientras que la verde es la trayectoria generada por el sistema de navegación). Como se puede notar en el punto 1 y 2, para distintos radios de corrección, el robot tiende a cambiar su orientación sin problema. Para el punto 3, el radio de corrección pequeño tiende a corregir agresivamente la orientación del robot muy cercano al punto de cambio de sección, por lo que radios tan pequeños no son apreciables, debido a que el control cambiaría rápidamente la orientación del robot y este a su vez, por su inercia puede girar más de lo esperado o en su defecto deslizarse sobre el piso. Para el punto 4, se estableció un radio de corrección muy grande, como se puede notar este radio se intercepta con obstáculos del ambiente, esto, tal y como está señalado en el punto rojo, puede provocar que el robot se choque al tratar de cambiar su orientación de manera más lenta de lo normal.

Fig 4. Afectación del esquema del control bajo diferentes radios de corrección a través de diferentes secciones o tramos de la trayectoria.



Los radios de corrección, experimentalmente para este robot AGV-UN se calcularon entre 25 y 30 cm, dando como resultados trayectorias como las mostradas en los puntos 1 y 2 de la anterior figura. El parámetro fijado en 28cm corresponde a un poco más de la mitad del carro (50cm de largo). Este parámetro ayuda a expresar que el robot no tiene un radio de giro 0cm.

B. Algoritmo de Control de Movimiento

Con el procedimiento y explicación anteriormente establecido, se procede a bosquejar el algoritmo de control de movimiento implementado para el robot AGV-UN. Cabe anotar que este algoritmo es desarrollo propio de los investigadores del proyecto.

Sea PT(x_0, y_0, θ θ_0..... x_m, y_m, θ_m) el conjunto de puntos m de la trayectoria generada por el sistema de navegación.

- Sea PT_t el vector del tramo t de la trayectoria.
- Sea PR (x_r, y_r, θ_r) la posición del robot en todo instante.
- Sea $\vec{Pr}$ el vector de la posición del robot en todo instante.
- Sea r_{cor} el radio de corrección establecido.

Por lo tanto

Para t=1 hasta t=m

$\Delta d_{t+1}=\vec{Pr}\cdot P\vec{T}r_{t+1}$

$\Delta\theta_{t+1}=\theta_{PT_{t+1}}-\theta_{Pr}$

Si $\Delta d_{t+1}\leq r_{cor}\wedge\Delta\theta_{t+1}>15°$

$e_{rot}=\Delta\theta_{t+1}$

$\beta=\dfrac{\Delta d_{t+1}}{r_{cor}}$

$Ferr=\beta e_{rot}$

$Vel=Vel_{ref}\pm(Vel_{ref}*Ferr)$

MoveMotors(Vel);

Sino

$\Delta d_t=\vec{Pr}\cdot P\vec{T}_t$

$\Delta\theta_t=\theta_{Pr}-\theta_{PT_t}$

Si $\Delta d_t>10mm$ v $\Delta\theta_t>15°$

StopMotors();

Sino

Si $\Delta d_{t+1}\leq r_{cor}\wedge\Delta\theta_{t+1}<15°$

$t=t+1;$

Sino

$e_{tras}=\Delta d_t$

$e_{rot}=\Delta\theta_t$

$Ferr=\alpha e_{tras}+\beta e_{rot}$

$Vel=Vel_{ref}\pm(Vel_{ref}*Ferr)$

MoveMotors(Vel);

Fin

IV. EFICIENCIA Y DESEMPEÑO DEL SISTEMA NAVEGACIÓN IMPLEMENTADO

Para medir el desempeño de los algoritmos implementados y desarrollados, se realizaron pruebas de navegación en un entorno determinado con el fin de estimar el rendimiento y consistencia de los algoritmos en ambientes dinámicos semi estructurados. Este tipo de ambiente se selecciona debido a la complejidad que acarrea para el algoritmo SLAM determinar regiones irregulares y de áreas pequeñas respecto al tamaño (área) del robot.

Con estas pruebas se precisa la eficiencia de los algoritmos y se puede caracterizar el Sistema de Navegación implementado.

A. Escenario de Pruebas

El escenario inicial para la realización de pruebas de mapeo y localización es el Laboratorio Fábrica Experimental de la Universidad Nacional de Colombia, este consiste de un entorno dividido en secciones que contienen máquinas, mesas de trabajo y otros elementos, el entorno es dinámico, sin embargo, este componente no es tenido en cuenta y se asume un entorno estático, además de que la técnica usada ayuda a omitir ciertos objetos móviles (cabe recordar que SLAM re mapea una y otra vez una misma zona antes de aceptarla parte de su mapa). La reconstrucción del entorno es un mapa 2D que se encuentra al nivel de la trama del sensor, ésta se encuentra a 10 cm del suelo del entorno por lo que elementos arriba y debajo de este no serán mapeados. El vehículo y sensor láser son colocados en posiciones aleatorias del laboratorio, posteriormente el vehículo es movido hasta obtener un mapa completo del entorno. En la Fig. 5 se presentan algunos resultados de los mapas obtenidos para diferentes resoluciones y puntos de inicio.

Con las pruebas de navegación y exploración autónoma se obtuvieron resultados notables y destacables. La navegación del robot en el entorno para reconstruirlo en su totalidad ronda alrededor de los 25 minutos, sin embargo se espera que con la implementación de la plataforma el tiempo de reconstrucción disminuya de acuerdo al número de robots navegando.

Fig 5. Mapa parcial obtenido con resolución de 1024 celdas.

Los círculos rojos en cada una de las figuras representan los puntos de inicio del robot móvil.

En la Fig. 6 se puede ver una vista general del robot en el ambiente real durante las pruebas de navegación.

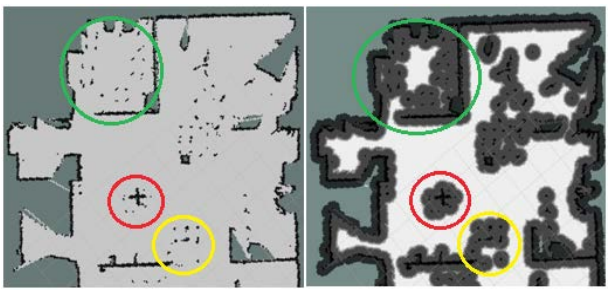
Fig 6. A) Pruebas de Navegación Robot AGV en Laboratorio Fábrica Experimental de la Universidad Nacional de Colombia. B) Montaje real en robot móvil AGV-UN del soporte y el sensor láser Sick LMS 102.

B. Desempeño del Sistema de Navegación

El sistema de navegación autónomo implementado para la red colaborativa de robots móviles en el Laboratorio LabFabEx es un sistema válido para entornos internos, dinámicos y semi estructurado. Es decir que, el sistema es para ambientes dentro de edificaciones, no para ambientes exteriores donde las características y condiciones ambientales son totalmente diferentes. Además, es válido para entornos dinámicos dado que el sistema es capaz de reaccionar ante obstáculos dinámicos presentes en el entorno (como personas u operarios del laboratorio LabFabEx).

Por último, es válido para ambientes semi estructurados, dado que la representación del mapa obtenido no solo capta zonas representativas del entorno como paredes, esquinas, figuras geométricas definidas (círculos, cuadrados, triángulos) sino que es capaz de determinar piezas del mobiliario deformes o simples puntos en el espacio sin representación gráfica en el mapa como las patas de una silla o una mesa las cuales el sistema sigue considerando como obstáculos estáticos y de acuerdo a ellos traza un radio de seguridad para evitar chocarse con ellos. De ahí que el robot nunca pase por debajo de sillas o mesas con un área no mayor a un 1 metro cuadrado.

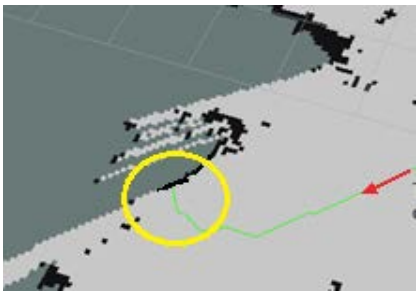
Fig 7. Mapa de costos (figura derecha) aplicado a mapa de navegación reconstruido (figura izquierda). En las aéreas demarcadas por círculos de color verde, rojo y amarillo se representan las zonas semi estructuradas que representan sillas o escritorios dentro del entorno.



Tal y como se muestra en la Fig. 7, una silla se representa como 4 puntos en el espacio reconstruido, donde cada uno cuenta con un radio de seguridad el cual se superponen con el fin de generar en el mapa de costos una zona de alto riesgo para el cruce del robot (zona de color gris oscuro). Con esto, se garantiza que el robot no intente pasar por debajo de este tipo de objetos.

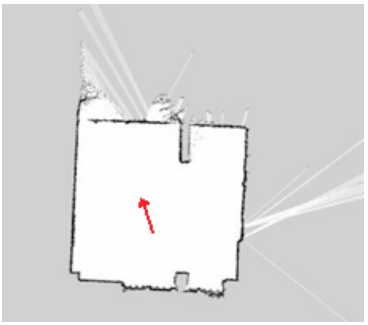
El sistema de navegación autónomo implementado en el robot móvil AGV-UN permite que el robot explore de manera autónoma un entorno trazando trayectorias calculadas hacia áreas no mapeadas. A medida que el robot actualiza su mapa y sigue una trayectoria, va verificando si la posición final de la trayectoria está ocupada o está dentro del radio de seguridad especificado para los obstáculos, es decir, que el robot verifica si su trayectoria al final está sobre un área donde hay un obstáculo. De esta manera se evita que el robot se dirija hacia un obstáculo y se optimiza el tiempo de exploración. Esta situación es mostrada en Fig. 8.

Fig 8. Robot, demarcado con la flecha de color rojo que demarca su pose, detecta un cambio en el objetivo de la trayectoria trazada. De acuerdo a esta situación, el robot corrige su trayectoria para evitar un choque y explorar zonas que ya fueron reconstruidas con muestreos previos durante el seguimiento de su trayectoria.



Ahora bien, de acuerdo al algoritmo de navegación Hector Navigation, el robot calcula sus trayectorias de acuerdo a fronteras de exploración libres; sin embargo, este debe tener una condición de parada cuando haya finalizado la exploración de su entorno. La condición de parada para esto, básicamente se determina cuando el robot después de haber calculado en 3 oportunidades el trazado de su trayectoria no puede obtener una trayectoria válida y segura. En Fig. 9 se expone esta situación.

Fig 9. Mapa de Navegación terminado. El robot, debido a su huella no puede trazar nuevas trayectorias, debido a que considera que ya culminó con el mapeo de su entorno. Este evento es capturado por el algoritmo de navegación y le notifica al usuario de la ocurrencia de dicho evento.



Una vez esta condición de parada ha sido alcanzada, el robot puede navegar dirigidamente hacia puntos del mapa reconstruido mediante el envío de

objetivos o realizar tareas mediante la recepción de órdenes por parte del usuario.

C. Característica del Sistema de Navegación Implementado

El sistema de navegación aplicado a la red colaborativa de robots móviles para el Laboratorio LabFabEx cuenta con las siguientes características:

Características robot:

- Exactitud en posición: +- 15mm
- Velocidad Crucero del Robot: 21 cm/s
- Autonomía: 3.5 horas
- Tamaño del Robot: 50cm x 45cm x 52cm
- Radio de Seguridad para evasión de obstáculos estáticos: 40 centímetros.
- Distancia Detección de Obstáculos: 20 centímetros.

Características Mapa Generado:

- Alcance Máximo de Mapeo: 18 metros.
- Tamaño de Mapa Reconstruido: 50 metros cuadrados.
- Tamaño de Mapa en Celdas: 1024x1024.
- Resolución de Mapa: 5 centímetros por cada celda del mapa.
- Reconstrucción de Entorno/Mapa en 2D.

Sistema de Navegación Robot:

- Navegación Autónoma: Exploración, Generación de Trayectorias, Evasión de Obstáculos, Localización Y Mapeo Autónomo.
- Navegación Dirigida: Envío y Seguimiento de Objetivos Específicos, Evasión de Obstáculos y Generación de Trayectorias.

Todos estos módulos fueron programados específicamente para el robot móvil AGV UN con tracción diferencial. Dichos módulos fueron validos a lo largo de pruebas de funcionamiento en las instalaciones del Laboratorio LabFabEx. Estos módulos fueron programados en lenguaje C++ para ROS versión Indigo en el sistema operativo Linux distribución Ubuntu Trusty 14.04.

V. DISCUSIÓN Y CONCLUSIONES

Los mapas obtenidos del entorno luego de la navegación y exploración autónoma son coherentes y consistentes con el laboratorio explorado, estos incluyen con gran precisión la definición del contorno, paredes y elementos divisorios que definen las secciones del espacio. Debido a la posición del sensor, el mapa contiene secciones de algunas máquinas, escritorios, sillas y puertas, por ejemplo. En algunas secciones del mapa es posible observar la posición de las 4 columnas soporte de bancos de trabajo. La forma del mapa obtenido no depende de la posición de inicio del sensor, sin embargo la orientación del mismo si se ve afectada por esta posición inicial. Al comparar las mediciones reales del entorno con mediciones sobre los mapas se observó una diferencia de 10mm cuyo valor corresponde, de igual manera, a la estimación de la posición del robot en el mapa. La resolución del mapa afecta únicamente la cantidad de información que puede ser reconstruida, por lo que esta deberá ser tenida en cuenta de acuerdo al tamaño del entorno a mapear. Los resultados obtenidos, en términos de la calidad y precisión del mapa, así como la estimación de la posición del robot son bastante buenos teniendo en cuenta que el sistema funciona en tiempo real y no requiere de información de odometría, además que la sensorica usada, Laser Sick LMS102, tiene un gran desempeño a nivel de fiabilidad en las medidas percibidas en cada muestreo.

Los módulos de navegación implementados presentaron un desempeño destacable y acorde a lo esperado, el algoritmo de generación de trayectoria mostró grandes virtudes ya que en las pruebas nunca se generaron trayectorias inseguras o indebidas. La precisión del sistema en cuanto a posición del robot en navegación se sintonizó en ± 15mm, siendo un resultado muy notable, dado que, como se expresó anteriormente, la precisión del sensor en la reconstrucción del mapa es cercana a este valor. Esto, fundamentalmente, se debe al control de movimiento implementado en el robot y al refinamiento y depuración de cada uno de los módulos programados en el sistema de navegación del robot.

El control de movimiento implementado para robots móviles de tracción diferencial, tuvo un comportamiento destacado dados los notables resultados expuestos en este documento.

Como se pudo evidenciar este artículo el Sistema de navegación autónomo implementado en la plataforma de robots móviles dispuestas en el laboratorio LabFabEx de la Universidad Nacional de Colombia sin lugar a dudas puede ser usado como estrategia de navegación en otras áreas y proyectos. Sin embargo, queda como línea de trabajo para futuros proyectos investigativos el mejoramiento de la precisión del sistema en cuanto a posicionamiento en el entorno. Adicionalmente, este trabajo puede servir como base para la extensión del sistema de navegación a sistemas multi-agentes heterogéneos o a ambientes externos, dado que los resultados consignados en este artículo se limitan a ambientes internos.

REFERENCIAS

[1]. Andra Aditya. "Implementation of a 4D fast SLAM including volumetric sum of the UAV". Sixth International Conference on Sensing Technology (ICST) 2012.

[2]. Jung-Suk Lee, Sang Yep Nam, Wan Kyun Chung. "Robust RBPF-SALM for Indoor Mobile Robots Using Sonar Sensors in Non-Static Environments". Robotics and Automation (ICRA), 2010 IEEE International Conference. pp 250-256.

[3]. Bo He, Shujing Zhang, Tianhong Yan, Tao Zhang, Yan Liang, Hongjin Zahng. "A Novel Combined SLAM Based in RBPF-SLAM and EIF-SLAM for Mobile System Sensing in a Large Scale Environment". Sensors 2011.

[4]. Diluka Moratuwage, Ba-Ngu Vo, Danwei Wang, Han Wang. "Extending Bayesian RFS SLAM to Multi-Vehicle SLAM". International Conference on Control, Automation, Robotics & Vision 2012.

[5]. Jeremy Thorpe, Robert McEliece. "Data Fusion Algorithms for Collaborative Robotic Exploration". Jet Propulsion Laboratory, California Institute of Technology. Mayo, 2002.

[6]. S. Kohlbrecher, J. Meyer, O. Von Stryk, U. Klingauf. "A Flexible and Scalable SLAM System with Full 3D Motion Estimation". In the Int. Symp. on Safety, Security and Rescue Robotics (SSRR), Nov. 2011.

[7]. Joao Machado Santos, David Portugal and Rui P. Rocha. "An Evaluation of 2D SLAM Techniques Available in Robot Operating System". Safety, Security, and Rescue Robotics (SSRR), 2013 IEEE International Symposium. ISBN:978-1-4799-0879-0. Octubre, 2013.

[8]. Thrun, S. Leonard, J. Simultaneous Localization and Mapping. 2011. Mobile and Distributed Robotics.

[9]. Kumar Pakki, Da-Wei Gu, Ian Postlethwaite. "SLAM Using EKF, EH and Mixed EH2/H Filter". IEEE International Symposium on Intelligent Control Part of 2010 IEEE Multi-Conference on Systems and Control Yokohama, Japan , September 8-10, 2010.

[10]. Ramazan HAVANGI, Mohammad NEKOUI, Mohammad TESHNEHHLAB. "An improved FastSLAM framework using soft computing". Turkish Journal of Electrical Engineering & Computer Sciences, Vol.20, No.1, 2012.

[11]. Solá Joan, Calleja Vidal, Civera Javier, Montiel José. "Impact of landmark Parametrization on Monocular EKF-SLAM with Points and Lines". International Journal of Computation Vision, 2012.

[12]. Thomas Lemaire, Cyrille Berger, Il-Kyun Jung, Simon Lacroix. Vision-Based SLAM: "Stereo and Monocular Approaches". International Journal of Computer Vision 74(3), pg. 343-364, 2007.

[13]. Engelhard Nikolas, Endres Felix, Hess Lürgen, Sturm Jürgen, Burgard Wolfram. "Real-time 3D visual Slam with a hand-held RGB-D camera". European Robotics Forum, 2011.

[14]. Thrun Sebastian , Dieter Fox Wolfram Burgard. "Occupancy Grid Mapping". Capitulo 9, Probabilistic Robotics. ISBN-10 0262201623. Octubre, 2005.

[15]. Stephan Wirth, Johannes Pellenz. "Exploration Transform: A stable exploring algorithm for robots in rescue environments". University of Koblenz and Landau, Koblenz, Germany.