UNIVERSIDAD
DE LA REPUBLICA
URUGUAY

Localización activa en el corto plazo utilizando solapamiento de hipótesis aplicada a robots de servicio

Federico Andrade

Programa de Posgrado en Informática
PEDECIBA Informática, Instituto de Computación, Facultad de Ingeniería
Universidad de la República

Montevideo – Uruguay
Mayo de 2020

UNIVERSIDAD
DE LA REPUBLICA
URUGUAY

Localización activa en el corto plazo utilizando solapamiento de hipótesis aplicada a robots de servicio

Federico Andrade

Tesis de Maestría presentada al Programa de Posgrado en Informática, PEDECIBA Informática de la Universidad de la República, como parte de los requisitos necesarios para la obtención del título de Magister en Informática.

Director:
Dr. Prof. Gonzalo Tejera

Director académico:
Dr. Prof. Héctor Cancela

Montevideo – Uruguay
Mayo de 2020

Andrade, Federico

Localización activa en el corto plazo utilizando solapamiento de hipótesis aplicada a robots de servicio / Federico Andrade. - Montevideo: Universidad de la República, PEDECIBA Informática, Instituto de Computación, Facultad de Ingeniería, 2020.

XXI, 76 p.: il.; 29, 7cm.

Director:

Gonzalo Tejera

Director académico:

Héctor Cancela

Tesis de Maestría – Universidad de la República, Programa en Informática, 2020.

Referencias bibliográficas: p. 71 – 74.

1. Robótica móvil, 2. Navegación, 3. Robots de servicio, 4. Localización activa, 5. Filtros de partículas. I. Tejera, Gonzalo, . II. Universidad de la República, Programa de Posgrado en Informática. III. Título.

INTEGRANTES DEL TRIBUNAL DE DEFENSA DE TESIS

Montevideo – Uruguay
Mayo de 2020

Agradecimientos

A Sabri por ser mi primer referencia y ejemplo de investigadora que ama lo que hace. A Aíma por haberme hecho hacer una pausa y tomar perspectiva. A Gonzalo Tejera por guía, aporte de ideas y principalmente hacerme pensar. A Héctor Cancela por estar siempre disponible para escucharme y apoyarme.

A Jesús Savage y todo el laboratorio de Bio-robótica de la UNAM por haberme tratado como un miembro más del equipo y por haberme permitido vivir una experiencia con la que llevaba años soñando.

A Meche por haber estado siempre atenta. A Eduardo Grampín por el aporte de ideas y por estar siempre pensando uno o dos años hacia adelante. Y finalmente a la CAP por el sostén económico.

RESUMEN

Uno de los problemas fundamentales de la robótica móvil es la localización. En la gran mayoría de las tareas que debe realizar un robot móvil, es necesario mantener una estimación precisa de la posición del robot. El problema de la localización se puede ver como un problema de correspondencia entre el sistema de coordenadas local del robot y el sistema de coordenadas global del mapa. Dentro de las soluciones a este problema, el enfoque más simple se conoce como localización pasiva. La localización pasiva consiste en estimar la posición del robot a partir de un mapa y de las percepciones que obtiene el robot a medida que navega en el entorno. Existe otro enfoque conocido como localización activa que se diferencia de la localización pasiva en que el robot ejecuta acciones intencionalmente para mejorar su localización. En este sentido, la mayoría de los trabajos sobre localización activa tienen como principal objetivo seleccionar las acciones que dirijan al robot (o a sus sensores) hacia zonas del mapa relevantes, aumentando la riqueza de la información adquirida en las observaciones del entorno, y en consecuencia, mejorando la precisión y disminuyendo la incertidumbre de la estimación de la posición del robot en el mapa. Según la literatura, la localización activa ha tenido mejores resultados que la localización pasiva. Otra categorización aplicada a los problemas de localización los divide entre localización global y seguimiento de posición. Localización global consiste en que inicialmente el robot no sabe en qué zona del entorno se encuentra y tiene como objetivo localizarse, mientras que en el seguimiento de posición el robot conoce su posición inicial y el objetivo es mantenerse ubicado a medida que navega por el entorno.

Este trabajo estudia la localización activa en interiores en el contexto del problema de seguimiento de posición. El sistema propuesto en esta tesis extiende el trabajo de Li et al. *Active localization with dynamic obstacles* [2016, IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 1902–1909]. En el trabajo de Li et al. se mantiene un conjunto de hipótesis sobre los posibles estados del robot (x, y, θ) en el mapa, se agrupan utilizando DBSCAN y luego se elige un representante por agrupación. Los representantes y sus mapas asociados se colocan en un marco de referencia

común, y se genera un mapa compuesto que permite saber cuales son los puntos del mapa que aportan más información sobre la localización. Luego, se elige la acción que dirija a los sensores hacia el punto que brinde mayor ganancia de información. Se espera que las observaciones del punto elegido descarten una cantidad relevante de hipótesis, mejorando la estimación sobre la posición del robot. Esta estrategia es aplicada en un contexto de localización global. En esta tesis se estudió la estrategia presentada por Li et al. y se extendió aplicándola al problema de seguimiento de posición, combinándola diversos algoritmos de agrupamiento como Kmeans++ y Spectral Clustering.

Se realizaron experimentos en distintos escenarios simulados y en un escenario real, con una ruta de navegación preestablecida, comparando cuatro estrategias diferentes (tres de localización activa y una de localización pasiva). Los experimentos presentan mejores resultados en la estimación de la posición para las estrategias propuestas en esta tesis (basadas en Kmeans++ y Spectral Clustering) con respecto a la estrategia utilizada en el trabajo de Li et al. (basado en DBSCAN), y a un algoritmo de localización pasiva.

Palabras claves:
Robótica móvil, Navegación, Robots de servicio, Localización activa, Filtros de partículas.

Lista de figuras

Lista de tablas

Lista de siglas

AMCL Localización Monte Carlo Adaptativa, por sus siglas en inglés.

CSV Valores Separados por Coma, por sus siglas en inglés.

DBSCAN Agrupamiento Espacial Basado en Densidad para Aplicaciones con Ruido, por sus siglas en inglés.

GPS Sistema de Posicionamiento Global, por sus siglas en inglés.

INCO Instituto de Computación de la Facultad de Ingeniería, UdelaR.

MATLAB Lenguaje de programación y entorno numérico de cómputo multi paradigma.

ROS Sistema Operativo para Robots, por sus siglas en inglés.

SLAM Localización y Mapeo Simultáneos, por sus siglas en inglés.

SPLAM Localización, Mapeo y Planificación simultáneos, por sus siglas en inglés.

Tabla de contenidos

Capítulo 1

Introducción

Uno de los desafíos fundamentales de los robots móviles es la localización. Junto al mapeo y a la planificación de trayectorias, forman la base de los problemas de navegación en robots móviles autónomos. En el contexto de la navegación y robótica móvil, un robot que no se puede localizar carece de utilidad. En la figura 1.1 se presentan las tres áreas básicas de la robótica, mapeo, localización y navegación, y las subáreas resultantes de su intersección, exploración, SLAM (Localización y Mapeo Simultáneos, por sus siglas en inglés.), localización activa y SPLAM (Localización, Mapeo y Planificación simultáneos, por sus siglas en inglés.). El objeto de estudio de esta tesis, la localización activa, se conforma por las áreas localización y planificación.

El enfoque básico de los problemas de localización es conocido en la literatura como la localización pasiva (Thrun et al., 2005). Este tipo de algoritmos se encarga de estimar de la posición (x, y, θ) del robot respecto a un mapa dado, basados únicamente en la información sensorial (interna y externa) que el robot obtiene a medida que navega por el entorno (p.e.: ODOMETRÍA, información láser, visión). En el caso de localización pasiva se asume que ni el movimiento del robot ni la posición de los sensores puede ser controlada por el algoritmo de localización. Durante la navegación el robot mantiene una *creencia* (del inglés BELIEF) sobre su ubicación, que consiste mantener una distribución de probabilidad sobre el estado del robot. En caso de que el robot esté perdido o no esté del todo bien localizado, deberá esperar a que casualmente, mientras navega con el objetivo de cumplir una misión determinada, obtenga la información necesaria para poder ubicarse.

Por otro lado, como fue demostrado por Burgard et al. (1997), si la ru-

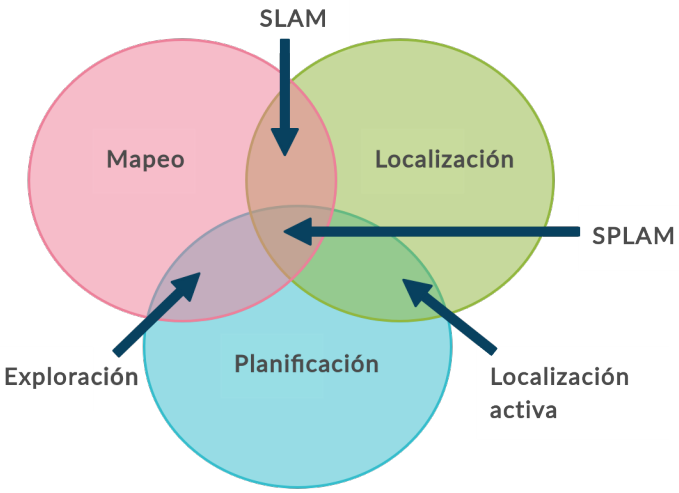


Figura 1.1: Áreas básicas de la navegación en robot móviles.

tina de localización puede controlar al robot parcial o totalmente, mejora la eficiencia y la robustez de la localización. En este contexto, la pregunta que se quiere responder es *¿hacia dónde moverse y/o hacia dónde mirar para disminuir la incertidumbre en la estimación de la posición del robot?*. Luego, controlar al robot para localizarlo es particularmente beneficioso en entornos que poseen pocas características distintivas. Algunos enfoques proponen planificar trayectorias que pasen cerca de los límites del entorno (Roy and Thrun, 2000), mientras que otros proponen apuntar los sensores del robot hacia lugares relevantes del ambiente (Correa and Soto, 2010). Como desventaja de mejorar la precisión de la ubicación del robot moviendo sus sensores o el propio robot a un punto del mapa particular, se tiene una pérdida de eficiencia asociada generalmente a necesitar más tiempo para cumplir la misión o un aumento de la distancia recorrida.

Una de las principales motivaciones para mejorar la estimación de la posición del robot se debe al incremento del uso de robots en ambientes cerrados como hogares, supermercados y oficinas, entre otros, donde el uso de sistemas de localización global no funcionan bien, o son muy caros, y además pueden requerir la instalación de hardware adicional.

El principal objetivo de investigación de esta tesis es minimizar el error en la localización de un robot móvil a través de la ejecución de acciones locales específicas para este propósito. Además, se espera que estas acciones incidan lo menos posible en la ejecución de la misión del robot. Si bien el trabajo está en-

focado en robots de servicio, la solución presentada y los resultados obtenidos pueden generalizarse para la mayoría de los robots móviles que se mueven en entornos similares (casas, oficinas, galpones, entre otros). Las contribuciones concretas de este trabajo son: una revisión profunda del estado del arte asociada a la localización activa, y un algoritmo de localización activa aplicado a un entorno local para el problema de seguimiento de posición.

Las preguntas de investigación específicas que se quieren responder en este trabajo son:

- *¿Se puede mejorar la precisión de la ubicación del robot con la técnica de localización activa utilizando solapamiento de hipótesis aplicada a un robot que inicialmente está localizado (seguimiento de posición)?*
- *¿Qué otros algoritmos de agrupamiento (del inglés clustering) se pueden adecuar a la estrategia en lugar de DBSCAN, teniendo en cuenta que la técnica se aplica a un robot inicialmente localizado?*

A continuación, se presenta la estructura del resto del documento de tesis. En el capítulo 2 se estudia la literatura asociada a la localización activa para sus distintos enfoques, localización global, seguimiento de posición y planificación con localización. Luego, en el capítulo 3 se revisan los principales conceptos teóricos sobre localización en robot móviles, localización activa y algoritmos de agrupamiento. En el capítulo 4 se define formalmente el problema y se propone una solución basada en el uso de algoritmos de agrupamiento distintos a los que aparecen en los trabajos relacionados. Más adelante, en el capítulo 5 se muestran los experimentos realizados, primero en simulación y luego con un robot real. Además, se reportan los resultados obtenidos. Finalmente, en el capítulo 6 se presentan las conclusiones de los resultados obtenidos en esta tesis. Para terminar se exponen varias propuestas de desarrollo a futuro motivadas por el trabajo realizado.

Capítulo 2

Trabajos relacionados

Este capítulo presenta el estudio de la literatura relacionada al tema de esta tesis. La sección 2.1 revisa los trabajos pioneros en el tema, que sientan las bases de la localización activa. Luego la sección 2.2 estudia específicamente la literatura relacionada al problema de un robot que se encuentra totalmente perdido en el entorno. A continuación, en la sección 2.3 se presentan los trabajos relevantes relacionados a la planificación de trayectorias que incluyen a la localización entre los parámetros de optimización. En la sección 2.4 se estudian trabajos asociados a las acciones de localización realizadas por el robot en un periodo de tiempo muy corto. Finalmente, en la sección 2.5 se resume el estudio de la literatura realizado en una tabla.

2.1. Artículos fundacionales

Las ventajas de utilizar localización activa en el contexto de la localización en robot móviles han sido introducidas por Burgard et al. (1997). Los autores proponen explotar la oportunidad de controlar al robot durante la localización y demuestran experimentalmente que controlar activamente los movimientos del robot mejora la eficiencia de la localización frente a los enfoques pasivos. En particular, la estrategia de los autores consiste en controlar la dirección del robot y además determinar hacia donde se deben apuntar los sensores para localizar al robot de forma más eficiente. Para elegir qué acciones realizar, se evalúa la entropía (medida de desorden del sistema) esperada (debido a que la posición real del robot, que es la variable de interés, no se puede sensar directamente) sobre la posición (x, y, θ) del robot (en adelante *pose*) para cada

posible acción a ejecutar. La estimación de la posición es mantenida utilizando el algoritmo de localización de Markov.

Dudek et al. (1998) presentan el concepto de polígono visible y lo utilizan para generar un conjunto de hipótesis posibles sobre la posición del robot. Los autores modelan el entorno como un polígono P, y las observaciones z_t del robot como un polígono V al que llaman polígono visible. A partir de V generan un conjunto de hipótesis H que contiene todas las posibles formas de colocar V en P. Luego con los elementos de H se genera un árbol de decisión de localización que contiene dos tipos de nodo: nodos de reducción de hipótesis y nodos de percepción, como se muestra en la figura 2.1. En la parte superior de la figura se presenta un mapa y cuatro hipótesis sobre la posición del robot $\{p_1, p_2, p_3, p_4\}$ a partir de una acción de sensado previa. En la pare inferior se presenta el árbol de decisión de localización con los nodos de reducción R y los nodos de percepción S. Los nodos de reducción de hipótesis son caminos relativos al robot hacia posiciones del mapa que diferencian a las hipótesis, y los nodos de percepción son puntos donde el robot realiza una observación y evalúa cuales hipótesis sobreviven. Siguiendo una de las ramas del árbol desde la raíz hacia las hojas, se logra reducir todas las hipótesis a una sola, localizando al robot con la mínima distancia recorrida. Los autores concluyen que si bien este problema es NP-complejo, la solución presentada lo resuelve en un tiempo razonable.

La propuesta de incorporar en la planificación la habilidad de localizar al robot fue desarrollada por Roy and Thrun (2000), y consiste en generar un camino que no solo optimice la distancia recorrida, sino que también considere que el robot llegue a su destino con la menor incertidumbre posible sobre su pose y como consecuencia una mejor estimación de su posición. En este trabajo se propone un esquema de navegación costera (basado en la navegación utilizada antiguamente por los barcos) en el que se busca que la ruta planificada pase cerca de lugares que tengan características relevantes. En la figura 2.2 se presenta un ejemplo en donde el robot elige el camino que minimiza la distancia entre la posición de salida y el objetivo (cuadro 2.2a). Por otro lado en el cuadro 2.2b se combina la minimización de la distancia con el objetivo de mantener mejor localizado al robot haciéndolo pasar cerca de lugares con características distinguibles. En este trabajo se asume que la distribución de probabilidad de la estimación de la pose del robot es una distribución gausiana, para disminuir la complejidad computacional del problema. Se precomputa

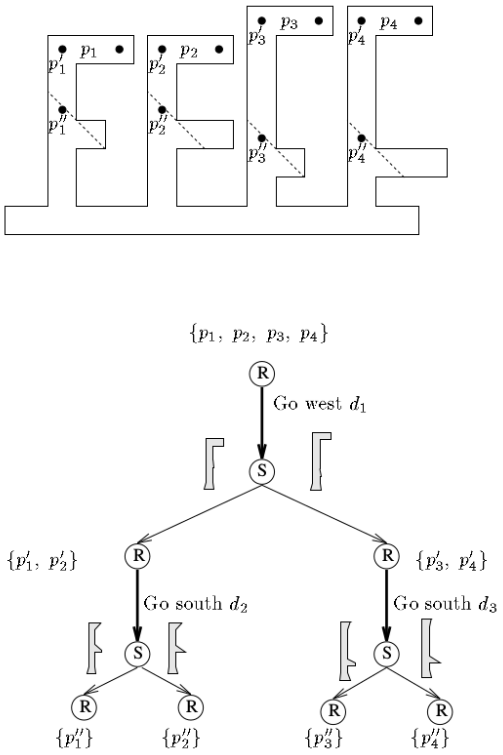
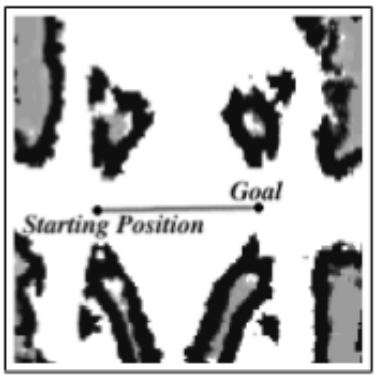
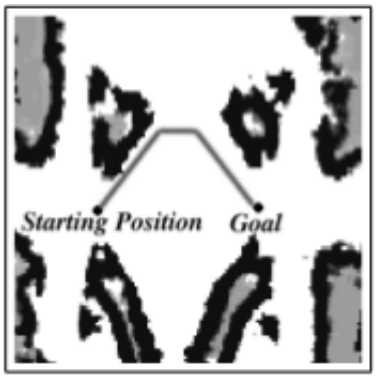


Figura 2.1: Árbol de decisión de localización. Figura extraída de Dudek et al. (1998).



(a) Navegación estándar



(b) Navegación costera

Figura 2.2: Dos estrategias de navegación. Figura extraída de Roy and Thrun (2000).

la esperanza de la entropía de las posibles poses en base a la probabilidad del sensado. Este cálculo se realiza una única vez a priori. El problema es modelado como un proceso de Markov, estocástico y se utiliza *value iteration* para determinar la acción óptima para cada estado. Los autores demuestran la propuesta experimentalmente con un robot RWI B-18 y un sensor láser que cubre 360 grados, obteniendo mejores resultados que otros planificadores convencionales.

2.2. Robot perdido en el entorno :: localización activa global

En el contexto de localización global, Li et al. (2016) proponen una estrategia de localización que consiste en guiar al robot a posiciones del mapa que resuelven ambigüedades y eliminan hipótesis en forma sistemática. Los autores utilizan un filtro de partículas (AMCL) para mantener la distribución de probabilidad de la estimación sobre la pose del robot. A partir del filtro de partículas, se agrupan las hipótesis utilizando el algoritmo DBSCAN (Agrupamiento Espacial Basado en Densidad para Aplicaciones con Ruido, por sus siglas en inglés.), y se elige el centroide de cada agrupamiento como representante. Luego se coloca a todos los representantes en un marco de referencia común, y se superponen los mapas asociados a cada uno, generando un mapa compuesto. Esto último se muestra en la figura 2.3. Cada celda del mapa compuesto toma un valor entero correspondiente a la suma de los valores de

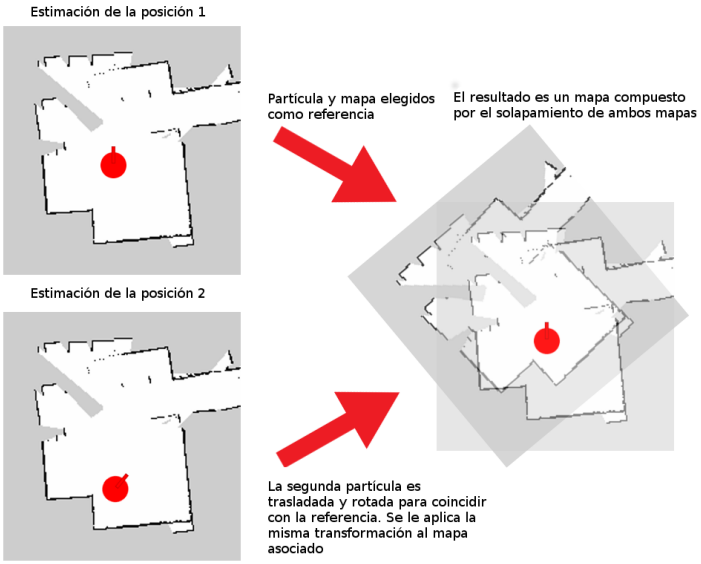


Figura 2.3: En esta figura se muestra como se superponen dos partículas representantes con sus respectivos mapas asociados.

las celdas de los mapas de cada representante. Se suma uno cuando la celda está ocupada y cero cuando está libre. Así, para maximizar el número de partículas que se descartarán, el robot se deberá mover hacia una posición del mapa compuesto que esté libre para algunos pero no para todos los representantes. Para elegir la celda destino se utilizan dos criterios: la cantidad de partículas que se espera descartar y la distancia que se debe recorrer. Una función de utilidad combina ambos criterios y devuelve la posición destino. Una vez alcanzada, el proceso comienza nuevamente. Se realizaron experimentos simulados utilizando Stage y con un robot real Turtlebot2 equipado con un sensor láser Hokuyo URG04-LX. El software fue desarrollado en ROS (Sistema Operativo para Robots, por sus siglas en inglés., Quigley et al. (2009)), pero no está disponible públicamente para su uso. Los resultados muestran la validez de la propuesta comparando con otros agentes en diversos escenarios.

A diferencia del trabajo anterior, Wang et al. (2012) presentan un mecanismo para acelerar la convergencia de la localización global sin incrementar la complejidad computacional. En este sentido los autores calculan una matriz de localizabilidad a priori y fuera de linea para cada pose del mapa, que permite representar con precisión la distribución de probabilidad de las posibles observaciones. Los autores construyen un modelo de sensado para el sensor láser que permite estimar la covarianza de las observaciones de los distintos puntos del

mapa. Luego para cada posible pose del robot en el mapa se representan las posibles observaciones en la matriz de localizabilidad, considerando cuatro posibles posiciones del robot en cada celda, que cubren los 360 grados alrededor de dicha celda. Luego se evalúa la matriz para todo el conjunto de partículas, y se toma la acción que se estima que dará el mejor resultado. Se realizaron experimentos con un robot de servicio (JiaoLong) equipado con un sensor láser SICK LMS111, sobre la plataforma de software CARMEN. Zhang and Scaramuzza (2019) presentan una propuesta muy similar, con la diferencia que está asociada a caracterizar el entorno 3D en matrices de información para cada marca del entorno. En lugar de calcular lo que se puede sensar desde cada pose, resumen en una estructura de datos toda la información asociada a cada marca. A partir de esta matriz se puede estimar la información que se obtendría de cada marca, desde las distintas poses del mapa. Una vez calculada esa matriz, se puede estimar cuál es la mejor posición de las cámaras para observar las marcas del entorno. Ambos trabajos se basan en la MATRIZ DE INFORMACIÓN DE FISHER.

Jung and Song (2017) presentan una alternativa a las propuestas clásicas de localización que necesitan navegar por el entorno para localizar al robot. Este esquema de localización utiliza un filtro de partículas para mantener el estado del sistema y solamente movimientos rotacionales. El primer paso que utiliza el algoritmo para localizarse es sensar los alrededores del robot con un sensor láser de 360 grados. Luego genera un histograma con las medidas del sensor láser, y además se simula el histograma de cada una de las partículas en el sistema. Se comparan los histogramas de cada partícula con el generado por los sensores (superponiéndolos para lograr la máxima coincidencia), logrando que si una partícula está en la posición de mayor verosimilitud, a pesar de estar rotada, tenga mayor peso que el resto. Tiene la ventaja de que ahorra tiempo ya que el robot no se debe trasladar por el entorno para localizarse, y además esto evita posibles colisiones con obstáculos. En este trabajo también se propone la idea del cálculo dinámico de muestras (partículas) necesarios para localizar al robot en el entorno, a diferencia de la mayoría de los trabajos que establecen un número fijo. Los resultados experimentales demostraron que se puede obtener la pose del robot en forma confiable a partir de movimientos rotacionales.

En el reciente trabajo de Gottipati et al. (2019) se propone un método de selección de acciones basado en redes neuronales y aprendizaje por refuerzo.

Cada vez que se reciben datos de la entrada sensorial z_t del sensor láser, se los convierte en una imagen en dos dimensiones, y es combinada con el mapa (conocido) en una red neuronal convolucional para obtener una distribución de probabilidad sobre las posibles poses del robot. Esta distribución es combinada con la creencia actual que tiene el robot sobre su posición y da como resultado la nueva creencia actual. La distribución resultante se utiliza como entrada del módulo de aprendizaje por refuerzo el cual devuelve una acción, a partir de una política π, que mueve al robot hacia el centroide de alguna de sus celdas adyacentes. Se experimentó con siete políticas distintas con diferentes recompensas (por ejemplo minimizar el error de localización o minimizar la dispersión). Tanto la red neuronal como el módulo de aprendizaje por refuerzo fueron entrenados en simulación. Para mayor robustez del sistema los autores introducen aleatoriedad en el dominio y ruido en las transiciones durante el entrenamiento. Se realizaron experimentos con dos robots reales JAY y Turtlebot2, integrados con dos sensores láser Slamtec A2M8 que cubren los 360 grados alrededor del robot. El software fue desarrollado en ROS. Los resultados muestran la validez de la propuesta de localización activa basada en aprendizaje.

2.3. Planificación con localización :: localización a largo plazo

Inoue et al. (2016) presentan un algoritmo de planificación que disminuye el error en la localización en el largo plazo, planificando rutas por terrenos ricos en características, cuya aplicación está orientada a robots exploradores planetarios. El algoritmo tiene dos objetivos: minimizar la covarianza de la estimación de la posición y minimizar la distancia recorrida. Para disminuir el error en la localización los autores desarrollaron el algoritmo Error Propagation A* (en adelante EPA*) que computa a priori el error de propagación de la covarianza de la localización a partir del modelo de sensado del entorno y del modelo de movimiento. Antes de aplicar el algoritmo los autores clasifican cada píxel del mapa como {*terreno rico en características, terreno pobre en características*}. A partir del mapa procesado, el algoritmo EPA* construye un grafo con las posibles trayectorias entre dos puntos predeterminados (uno de salida y uno de llegada) teniendo en cuenta el error de los modelos de

sensado y movimiento. Luego calcula el error de propagación para cada nodo y finalmente busca un camino que minimiza la función de costo que combina la propagación del error en las aristas del grafo y la distancia a recorrer. Los experimentos se realizaron en simulación a partir de datos reales obtenidos del robot CURIOSITY. Los resultados experimentales de EPA* fueron comparados con el algoritmo A* y presentaron mejores resultados en la localización a pesar de que la distancia recorrida es mayor.

2.4. Acciones en pequeños lapsos de tiempo :: localización a corto plazo

Velez et al. (2011) proponen modificar localmente la planificación de trayectorias en beneficio de obtener mejores observaciones de objetos relevantes que se encuentran en el entorno. La propuesta se divide básicamente en el desarrollo de dos componentes: el modelo de sensado y el algoritmo de planificación para mejorar la detección. El modelo de sensado permite caracterizar un conjunto de celdas alrededor del objeto valorando que tan informativas son esas celdas para el robot con respecto al objeto a sensar. El algoritmo de planificación para mejorar la detección selecciona el conjunto de acciones que minimizan tanto los costos de desviarse por el camino preestablecido (energía y tiempo), como la ganancia de información esperada del sensado del objeto. El procedimiento propuesto por los autores es presentado en la figura 2.4, donde se muestran de izquierda a derecha las tres etapas principales de la propuesta: detectar al objeto, estimar la ganancia de información en las celdas cercanas al objeto y modificar la planificación para recorrer algunas de las celdas que aportan mayor información. Una vez que el robot detecta el objeto, el modelo de sensado procesa las celdas alrededor del objeto (definen un entorno experimental de tres metros). Luego se ejecuta el algoritmo de planificación que devuelve como salida una modificación en la ruta original que contempla el pasaje por celdas que permiten un mejor sensado del objeto. Para la navegación utilizaron un sensor láser, mientras que para sensar a los objetos se utilizó una cámara estéreo. Se realizaron experimentos en simulación y en una silla de rueda autónoma, comparando la propuesta con varios algoritmos similares. El algoritmo propuesto presentó un mejor desempeño.

En el estudio de Correa and Soto (2010) se presenta una estrategia de

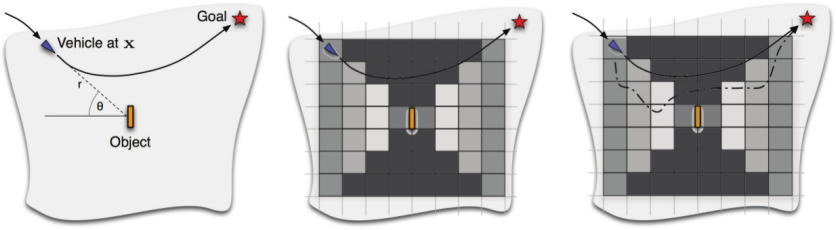


Figura 2.4: Planificación para mejorar la percepción. Figura extraída de Velez et al. (2011)

.

percepción activa utilizando una cámara montada sobre dos motores que trabajan independientes del movimiento del robot. A diferencia de la mayoría de las propuestas, en esta se utiliza un MAPA TOPOLÓGICO del entorno en lugar de un MAPA MÉTRICO. Como en la mayoría de los trabajos presentados en este capítulo, se utiliza un filtro bayesiano para estimar la pose del robot. Además de las etapas clásicas de predicción, observación y actualización del filtro, se agrega una etapa más bajo el nombre de Selección de Acción de Observación que consiste en elegir la acción perceptual que produzca la mayor disminución de la incertidumbre en la estimación de la pose del robot dentro del conjunto de las posibles acciones a tomar. Para elegir las acciones perceptuales a tomar, se aplican conceptos de la teoría de ganancia de información, de forma de cuantificar la entropía resultante de cada posible acción a tomar, y además se consideró el costo de realizar cada acción (el tiempo de mover la cámara de una posición a otra). Se realizaron experimentos en un entorno real con un robot diferencial PIONEER equipado con una cámara con un mecanismo de rotación horizontal y vertical. Se utilizó MATLAB como base para el desarrollo. La propuesta de percepción activa demostró ser conveniente frente a enfoques pasivos.

2.5. Resumen

En la tabla 2.1 se presenta un resumen de los trabajos relevados en la literatura durante el estudio del estado del arte, y se incluye un breve comentario del aspecto más destacado de cada uno.

El estudio de los trabajos relacionados permitió identificar varias técnicas distintas aplicadas a la localización activa en robot móviles. Si bien los artículos están agrupados en categorías de subproblemas diferentes, como localización

Tabla 2.1: Resumen de los trabajos relacionados relevados en la literatura.

Trabajos fundacionales	
referencia	*resumen*
Burgard et al. (1997)	Controla las acciones del robot tanto en la dirección como en la posición de los sensores. Se optimiza la disminución de la entropía y la distancia recorrida.
Roy and Thrun (2000)	Introduce el concepto de planificación costera en la navegación.
Dudek et al. (1998)	Genera un árbol de decisión que permite reducir hipótesis.
Localización global	
referencia	*resumen*
Li et al. (2016)	Resume la información de las muestras en un único mapa y guía al robot hacia el punto de mayor discrepacia.
Jung and Song (2017)	Simula el sensado 360 grados de cada partícula y se busca la mayor verosimilitud con z_t.
Wang et al. (2012)	Crea a priori (OFFLINE) una matriz de localizabilidad del mapa, que estima la incertidumbre del sensado en cada pose.
Zhang and Scaramuzza (2019)	Presenta una estructura de datos que mantiene la información de cada marca que permite estimar la información que se puede obtener desde cualquier pose del mapa.
Gottipati et al. (2019)	Utiliza una red neuronal convolucional para estimar la posición del robot a partir de z_t y aprendizaje por refuerzo para elegir la acción que favorezca la localización.
Localización largo plazo	
referencia	*resumen*
Inoue et al. (2016)	Propone un algoritmo de planificación que minimiza el error en la localización y la distancia.
Localización corto plazo	
referencia	*resumen*
Velez et al. (2011)	Desarrolla un modelo sensorial que permite estimar la entropía del sensado desde poses cercanas a los objetos de interés. Se planifica localmente para visitar dichas poses.
Correa and Soto (2010)	Controla una cámara montada sobre el robot buscando observar los lugares más ricos en información. Se tiene en cuenta el costo de realizar cada acción perceptual.

global, seguimiento de posición y planificación con localización, la mayoría de las ideas que aparecen en estos trabajos son aplicables al problema de estudio de esta tesis, que es la *localización activa a corto plazo, en el contexto del problema de seguimiento de posición*. En el estudio también se observa que la mayoría de los trabajos utilizan filtros de partículas para mantener la estimación de la posición del robot. Otra observación que surge del análisis literario, es que a pesar del uso de distintas técnicas o algoritmos, la mayoría de los métodos se pueden resumir en *elegir la acción que maximice la ganancia de información en las observaciones del robot en los próximos pasos*, y de esto se espera una disminución de la incertidumbre en la estimación de la posición que mantiene el robot. En particular, esta tesis extiende la propuesta original de crear un mapa compuesto a partir de las hipótesis, aplicada a localización global (Li et al., 2016), adaptando dicha propuesta para su aplicación en el problema de seguimiento de posición. A diferencia de la propuesta original, en este trabajo se estudió el uso de otros algoritmos de agrupamiento que son más aplicables al problema de seguimiento de posición. La mayoría de las propuestas trabajan con el mapa completo del entorno, en esta tesis se utiliza un entorno acotado de celdas a partir de ideas tomadas de Velez et al. (2011).

La principal contribución de esta tesis es la aplicación del método de la generación de un mapa compuesto al problema de seguimiento de posición, proponiendo el uso de algoritmos de agrupamiento diferentes al de la propuesta original, y además la aplicación de la técnica en un entorno local, que reduce el cómputo del sistema. No se encontró en la literatura ningún trabajo que combine estas ideas como se hace en este trabajo de tesis.

Capítulo 3

Localización activa y algoritmos de agrupamiento

En este capítulo se presentan los conceptos principales asociados al tema de esta tesis. La sección 3.1 introduce el tema de localización y las principales taxonomías asociadas. En la sección 3.2 se presentan los enfoques temporales de los algoritmos de localización activa. En la sección 3.3 se introduce la base teórica de los algoritmos de localización probabilísticos, en particular los Filtros de Partículas. Finalmente, en la sección 3.4 presentan los algoritmos de agrupamiento relevados que se utilizan en el postprocesamiento de las partículas.

3.1. Localización en robots móviles

El contenido de esta sección está basado principalmente en el libro Probabilistic Robotics (Thrun et al., 2005).

La localización en robot móviles es el problema de determinar la posición de un robot relativa a un mapa del entorno. También es referido como un problema de estimación de la posición. Como se mencionó en la sección 2.1 la posición de un robot es conocida en la literatura como *pose*, y en el caso particular de un robot que se mueve en un plano es determinada por el vector $X = (x, y, \theta)$, donde x e y son las coordenadas cartesianas y θ es la rotación con respecto al eje z, en el marco de referencia del mapa. En la figura 3.1 se presenta un modelo gráfico del problema de localización en robots móviles, donde los círculos con relleno gris son conocidos: el mapa m, las medidas z_i obtenidas a través de

los sensores y las acciones u_i realizadas por el robot (también conocidas como *controles* en la literatura). El objetivo de la localización es inferir la variable $x_{t+1} = (x, y, \theta)$ (pose del robot) en función de la estimación anterior x_t, del sensado z_{t+1} y de los controles u_{t+1}. En otras palabras, el robot conoce el mapa del entorno y su meta es determinar su posición relativa al mapa basado en sus percepciones (sensado z) del entorno y es sus propios movimientos (control u). La calidad del sistema de localización está determinada por la capacidad del algoritmo de minimizar la diferencia entre la posición estimada y la posición real del robot. La diferencia entre la estimación y la posición real es una de las principales métricas para evaluar algoritmos de localización y se presenta con mayor detalle en la sección 5.1.4.

La localización se puede ver como un problema de transformación de coordenadas. Los mapas están descriptos en un sistema de coordenadas global que es independiente de la posición del robot. La localización es el proceso de establecer una correspondencia entre el sistema de coordenadas del mapa y el sistema de coordenadas local del robot. Desafortunadamente la pose (o estado) X_{t+1} del robot no puede ser sensada directamente sino que debe ser inferida del sensado del entorno y el movimiento propio, por ello en la literatura X_{t+1} también es definido como estado oculto.

Si bien existen algunos métodos de localización global: GPS, cámara global, triangulación con sensores incorporados en el entorno, entre otros, muchos de ellos son imprecisos o costosos en cuanto al despliegue de elementos de hardware, implantación y mantenimiento.

En cuanto a los sensores del robot existen dificultades adicionales como el hecho de que un solo sensor suele ser insuficiente para determinar la pose, por lo que se debe integrar la información de varios sensores en el tiempo para determinar su posición. Además se debe tener en cuenta que los sensores presentan ruido en sus mediciones.

Para comprender la idea se puede pensar en un robot que se encuentra en un pasillo de oficinas como se presenta en la figura 3.2. En este caso el lugar es muy simétrico y al robot no le alcanza con tomar una medida para saber dónde se encuentra, sino que le va a llevar varios pasos converger a la pose en la que se encuentra.

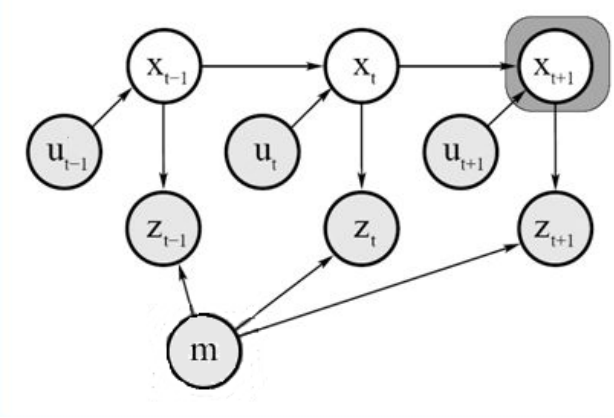


Figura 3.1: Modelo gráfico de localización de robots móviles. Figura extraída de Thrun et al. (2005).

3.1.1. Taxonomía de los problemas de localización

No todos los problemas de localización son igualmente difíciles de resolver. El nivel de dificultad cambia según el contexto del problema. Por ello se clasifica a los problemas de localización según características como: el tipo de algoritmo, el conocimiento inicial del robot, la naturaleza del entorno y la cantidad de robots, entre otras. A continuación se presentan las taxonomías más relevantes para este trabajo. Existen otras taxonomías que se pueden consultar en la literatura (Thrun et al., 2005).

Pasivo versus activo

Una clasificación que se aplica a los problemas de localización se basa en el hecho de que el algoritmo de localización controle o no los movimientos del robot. Distinguimos dos casos, el primero es localización pasiva, en la cual solamente se observa la información de movimiento ($u_{1:t}$) y la información sensorial ($z_{1:t}$) en el tiempo, y se utiliza para inferir la pose del robot. El robot se controla a través de algún algoritmo de navegación que no tiene como objetivo facilitar la localización. Un ejemplo puede ser un robot que navega por un camino fijo preestablecido que minimiza la distancia al objetivo o que se mueve en forma aleatoria. En estos casos el módulo de localización trabaja en forma independiente con la información obtenida del sensado y del movimiento.

Por otro lado los algoritmos de localización activa sí controlan al robot con el fin de mejorar la localización y minimizar la incertidumbre de la estimación. En este contexto, el algoritmo moverá al robot en la dirección que le aporte

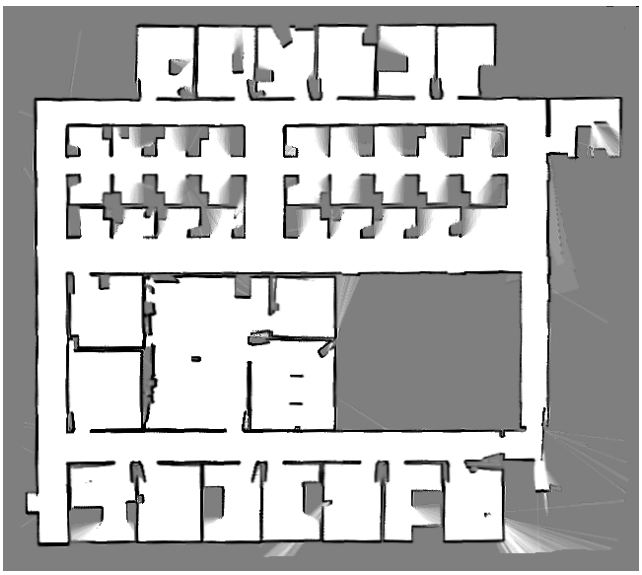


Figura 3.2: Mapa de oficinas caracterizado por un pasillo simétrico que presenta un desafío de localización debido a la simetría de los espacios.

mayor información sobre su ubicación. Los enfoques activos suelen tener mejores resultados de localización frente a los pasivos (Burgard et al., 1997). Un ejemplo de ello es la navegación costera (del inglés *coastal navigation*) presentado en la figura 2.2 en el capítulo 2. Otro ejemplo es cuando el robot se encuentra en un entorno muy simétrico y tiene una distribución multimodal (es decir que su creencia sobre la pose está dividida en por lo menos dos lugares con alta probabilidad) asociada a su pose como se muestra en la figura 3.3. En el caso de la figura el algoritmo de localización estima que está al final de un pasillo al lado de una puerta, dado que la información sensorial obtenida por el láser coincide con esos espacios del mapa, pero no sabe sobre cuál de las puertas se encuentra. Deberá tomar alguna acción para discernir cuál es la posición correcta entre sus dos máximos locales. Para desambiguar la posición el robot deberá moverse hacia lugares con características *relevantes*. En casos como este la localización activa presenta mejores resultados. En lugar de solamente esperar a que el robot accidentalmente entre en alguna de las oficinas (como sería el caso de la localización pasiva), el algoritmo de localización activa reconoce que el robot se encuentra en esta situación y resuelve entrar en una de las oficinas para eliminar las hipótesis menos verosímiles.

Sin embargo, la principal limitación de la localización activa es que requiere control del robot, y esto se debe conjugar con el hecho de que el robot esté desenvolviendo una tarea principal (como llegar a un punto determinado

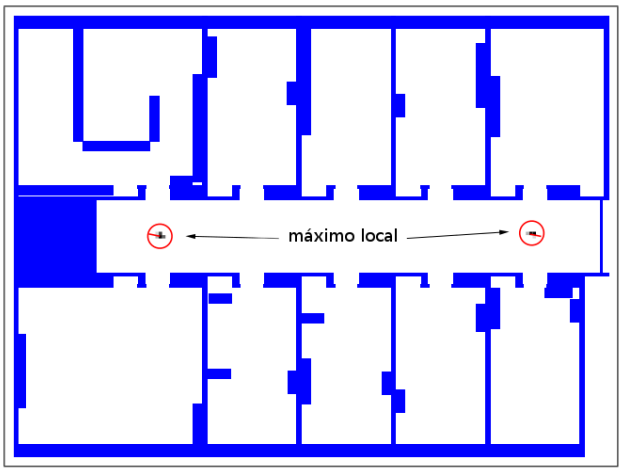


Figura 3.3: Distribución de probabilidad multimodal. Figura extraída de Thrun et al. (2005).

del mapa). Por ello el problema se transforma en un problema de combinar objetivos específicos con el objetivo de mantenerse localizado, lo cual lo lleva a un problema de toma de decisiones.

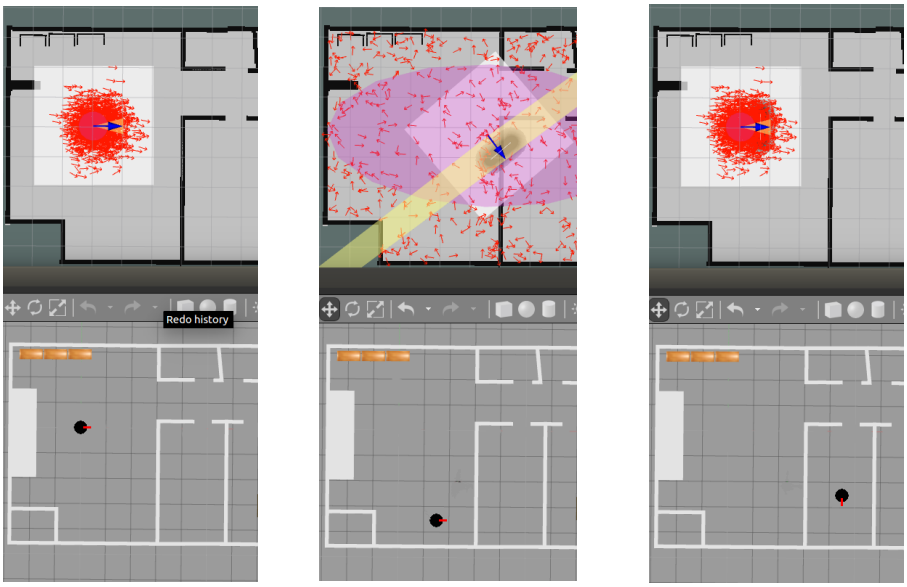
Algunas técnicas de localización activa se construyen sobre técnicas pasivas, como una extensión. En la sección 3.2 se presenta una explicación más detallada sobre el funcionamiento los algoritmos de localización activa.

Local versus global

En la figura 3.4 se presentan tres enfoques asociados a la categorización del conocimiento inicial que posee el robot. Uno de los problemas más simples es conocido como seguimiento de posición (del inglés *position tracking*) y consiste en el que el robot conoce la posición inicial. Localizarse se transforma en poder lidiar con el ruido en el sensado a medida que el robot se mueve. La posición del robot se suele aproximar con una distribución unimodal, típicamente una *gaussiana*.

En segundo lugar tenemos la localización global y se refiere a que inicialmente el robot no conoce su ubicación en el mapa. Este problema es más difícil que el anterior y no responde a una distribución unimodal. El robot asigna equiprobabilidad $1/n$ a cada pose del mapa.

El robot secuestrado es una variante de los problemas anteriores que lo vuelve un poco más difícil. Durante una operación normal (se asume que el robot está localizado), el robot es secuestrado y transportado a otra posición.



(a) Seguimiento de posición (b) Localización global (c) Robot secuetrado.

Figura 3.4: En las figuras se presentan los tres tipos de localización asociadas al conocimiento inicial que tiene el robot. En los cuadros de arriba se presenta una imagen de la representación que mantiene el robot, donde aparece el mapa, las flechas rojas (partículas) representando cada hipótesis de pose que mantiene el robot, y la flecha azul representa el promedio de las hipótesis. En los cuadros de abajo se muestra una imagen de la simulación con la posición real del robot (el círculo negro). En la figura (a) se muestra que la media (flecha azul) de la distribución de probabilidad de la estimación de la pose del robot coincide con su posición real en el escenario. En la figura (b) el robot cree que puede estar en cualquier parte del escenario con igual probabilidad. En la figura (c) podemos ver que no coincide la creencia del robot con su posición real.

Luego de este evento, la información de localización que el robot posee no se corresponde con la real. Si bien esta situación no es común, probar algoritmos de localización secuestrando al robot mide su habilidad de recuperarse de fallas en la localización.

Entornos estáticos versus entornos dinámicos

Otra dimensión que impacta directamente en la dificultad del problema de localización es el entorno. Una clasificación clásica divide a los entornos entre estáticos o dinámicos. Entornos estáticos son aquellos en los que la única variable es la pose del robot. El resto de los objetos permanecen inmóviles. Este tipo de entornos tiene propiedades matemáticas que lo hacen eficiente para la estimación probabilística.

Los entornos dinámicos poseen objetos propios que pueden cambiar su posición o configuración con el pasar del tiempo. Se presta principal atención en los cambios que persisten en el tiempo y que impactan en más de una lectura de los sensores. Los cambios que afectan pocas lecturas suelen ser tratados como ruido, como por ejemplo un animal o una persona que se cruza en el camino del robot. Ejemplos de cambios persistentes pueden ser la luz del sol, muebles o puertas. Existen dos enfoques principales en entornos dinámicos. El primero es incluir las entidades dinámicas en el vector de estado (además de la pose del robot), eso justifica las asunciones en un esquema de Markov pero la complejidad computacional del problema crece muchísimo. El otro enfoque es filtrar la información sensorial tratando de eliminar los objetos dinámicos no modelados.

Un solo robot versus multi-robot

El caso de localización con un solo robot es el más estudiado y sencillo. Este caso ofrece la ventaja de tener una única plataforma que recolecta la información.

En el caso de múltiples robots trabajando en equipo en el problema de localización, se puede mejorar el tiempo de localización si comparten las localizaciones independientes y pueden reconocerse entre ellos. Sin embargo, existen problemas principalmente en el mantenimiento de la representación, en la comunicación entre los agentes, y la distribución de los datos.

Figura 3.5: Localización activa desde le punto de vista temporal.

3.2. Localización activa

Como se mencionó en el capítulo 1, la localización activa asume que la rutina de localización tiene control parcial o total sobre el robot, permitiéndole mejorar la eficiencia (debido que consigue localizarse más rápido), y la robustez de la localización (debido a que tiene una forma de decidir qué hacer para localizarse mejor). Las preguntas principales que busca responder esta técnica son *¿hacia dónde moverse y/o hacia dónde mirar para disminuir la incertidumbre en la pose del robot?* (Burgard et al., 1997).

Dentro de la localización activa podemos encontrar un espectro de enfoques dependiendo del lapso de tiempo en el que se aplica la técnica. Este espectro se presenta en la figura 3.5. En el extremo izquierdo del espectro podemos encontrar la optimización de la incertidumbre en la pose como uno más de los parámetros a optimizar por el planificador global (además de la distancia al objetivo, evasión segura de obstáculos, entre otros). En el extremo derecho del espectro podemos encontrar una rutina de localización activa similar a un COMPORTAMIENTO REACTIVO (Brooks, 1986), que solamente se dispara condicionalmente. Un ejemplo sería que se dispare si la incertidumbre supera un umbral determinado, teniendo una participación muy concreta durante la misión del robot.

3.2.1. Localización activa en el largo plazo

Los planificadores tradicionales para robots móviles (Siciliano and Khatib, 2007) suelen optimizar la llegada al objetivo, la distancia recorrida, la distancia a los obstáculos, entre otros. En general estos algoritmos asumen que el robot se mantiene bien localizado durante su misión, sin embargo esto no siempre se cumple, principalmente en entornos reales. Existen trabajos que integran en el planificador global el control de la incertidumbre de la pose (Roy and Thrun, 2000; Inoue et al., 2016; Velez et al., 2011). Un ejemplo de esto es el trabajo de

Roy and Thrun (2000) que combina los objetivos estándar de un planificador global con la navegación que utilizan los barcos para no perderse (navegación costera).

3.2.2. Localización activa en el corto plazo

En el extremo derecho de la figura 3.5 tenemos la localización activa a corto plazo, esto significa que la técnica se aplica en un contexto local, más acotado, y no en la planificación de largo aliento como se mencionó al inicio de esta sección. En este caso aplicamos la localización activa en un margen de tiempo acotado que podría implicar que el robot detenga su misión temporalmente y se mueva hasta un punto determinado o que apunte sus sensores hacia una zona rica en características o ambas (Correa and Soto, 2010; Liu et al., 2012). Si bien la ejecución de este tipo de conductas es acotada en el tiempo, las mismas pueden ejecutarse múltiples veces a lo largo del desarrollo de la misión del robot.

Como síntesis de las clasificaciones presentadas el autor quiere resaltar que este trabajo de tesis se enmarca en el problema de la localización activa local a corto plazo, en un entorno estático y con un solo robot.

3.3. Algoritmo de localización pasiva

A continuación se describe un algoritmo de localización pasiva basado en un filtro de partículas. Si bien esta tesis está enfocada en la localización activa, es relevante presentar el funcionamiento del filtro de partículas dado que el algoritmo realizado por el autor se relaciona fuertemente con este filtro. Previamente se hará mención al filtro de Bayes, que es un caso más general del filtro de partículas.

3.3.1. Filtro de Bayes

Los filtros de Bayes (Thrun et al., 2005) son algoritmos genéricos que permiten estimar el estado oculto del sistema $x_{1:t}$ (la pose del robot (x, y, θ)) en función de la información de sensado $z_{1:t}$ y odometría $u_{1:t}$, como se presenta en la figura 3.1. El problema de localización se puede formular como:

$$p(x_{1:t}|u_{1:t}, z_{1:t}) \tag{3.1}$$

Es decir, la distribución de probabilidad del estado $x_{1:t}$ $(x_1..x_t)$ que se adapta mejor a las observaciones $z_{1:t}$ y controles $u_{1:t}$. Alternativamente, como se mencionó en el capítulo 1, se puede encontrar en la literatura esta probabilidad expresada como:

$$bel(x_{1:t}|u_{1:t}, z_{1:t}) \tag{3.2}$$

para hacer énfasis en el hecho de que es la creencia del robot sobre el estado del sistema. Asumiendo que tenemos la estimación del estado anterior x_{t-1}, la expresión del filtro se puede dividir en dos ecuaciones que se presentan a continuación:

$$\overline{bel}(x_t) = \int p(x_t|u_t, x_{t-1}) \times bel(x_{t-1})dx_{t-1} \tag{3.3}$$

$$bel(x_t) = \mu p(z_t|x_t) \times \overline{bel}(x_t) \tag{3.4}$$

De las ecuaciones 3.3 y 3.4 se deduce que es posible obtener la distribución de probabilidad del estado en el tiempo t a partir del estado en el tiempo anterior $t-1$ utilizando como paso intermedio el cálculo de $\overline{bel}(x_t)$. En la ecuación 3.3 se procesa la acción del robot u_t y eso se hace integrando la distribución asignada a x_{t-1} con la probabilidad que el control u_t induce en la transición desde x_{t-1} a x_t. Luego, la ecuación 3.4 multiplica a $\overline{bel}(x_t)$ por la probabilidad de haber observado z_t desde x_t. Esto permite mantener una distribución de probabilidad del estado oculto procesando la información de sensado y odometría de a una a la vez. El parámetro μ es utilizado para normalizar el resultado. Ambas ecuaciones se aplican para cada una de las hipótesis, como se detallará a continuación cuando se presente el filtro de partículas.

3.3.2. Filtro de partículas

Los filtros de partículas (Thrun et al., 2005; Llofriu and Andrade, 2012) aproximan la distribución de probabilidad del estado del sistema X_t utilizando métodos de Monte Carlo. Para ello mantienen un conjunto finito de N partículas que conforman la distribución de probabilidad del estado del sistema X_i. Cada partícula representa una posible pose en la que puede estar el robot, es decir, mantiene una ubicación (x, y, θ) y un mapa. Un filtro de partículas actualiza la distribución cada vez que tiene información disponible,

tanto de sensado como de odometría (movimiento del robot). Para el tiempo t, el conjunto de muestras X_t es:

$$X_t = x_t^{[1]}, x_t^{[2]}, ..., x_t^{[N]} \tag{3.5}$$

Cada partícula $x_t^{[m]}$ (con $1 \leq m \leq N$) es una instancia concreta de un posible estado del robot en el tiempo t. Dicho de otra manera, una partícula es una hipótesis del estado del robot en el tiempo t. Aquí, N denota el número de partículas en el conjunto X_t. En la práctica este número es generalmente grande, el rango de valores más usado en la literatura oscila entre $N = 1000$ y $N = 30000$.

Actualización de la distribución de probabilidad

La dinámica de actualización de la función de densidad del filtro de partículas consta de tres pasos, el de predicción, el de actualización o cálculo del peso y el de remuestreo. Estos tres pasos forman parte del algoritmo 1. Las partículas son inicializadas en torno a la posición inicial del robot y con una varianza que representa la incertidumbre inicial sobre la ubicación del robot.

Paso de predicción (del inglés *predict*) El paso de predicción busca modificar la función de densidad para reflejar el movimiento realizado por el robot. Para esto, se modifica la posición de cada una de las partículas según:

$$x_t = f(x_{t-1}, u_t) \tag{3.6}$$

donde f representa el modelo de movimiento a utilizar. En el paso 4 del algoritmo 1 es donde se realiza la predicción del movimiento.

En la figura 3.6 se puede observar el efecto de aplicar sucesivamente el paso de predicción a un conjunto de partículas. Las imágenes muestran el efecto de sucesivos movimientos en línea recta seguidos de rotaciones. Esta imagen ilustra como en este paso del algoritmo aumenta la dispersión de las partículas, y por consiguiente la varianza de la distribución. Esto se debe a que el modelo de movimiento incluye una componente de ruido para modelar la naturaleza estocástica del movimiento del robot. Este ruido se va acumulando en cada paso de predicción y aumenta la incertidumbre de la estimación.

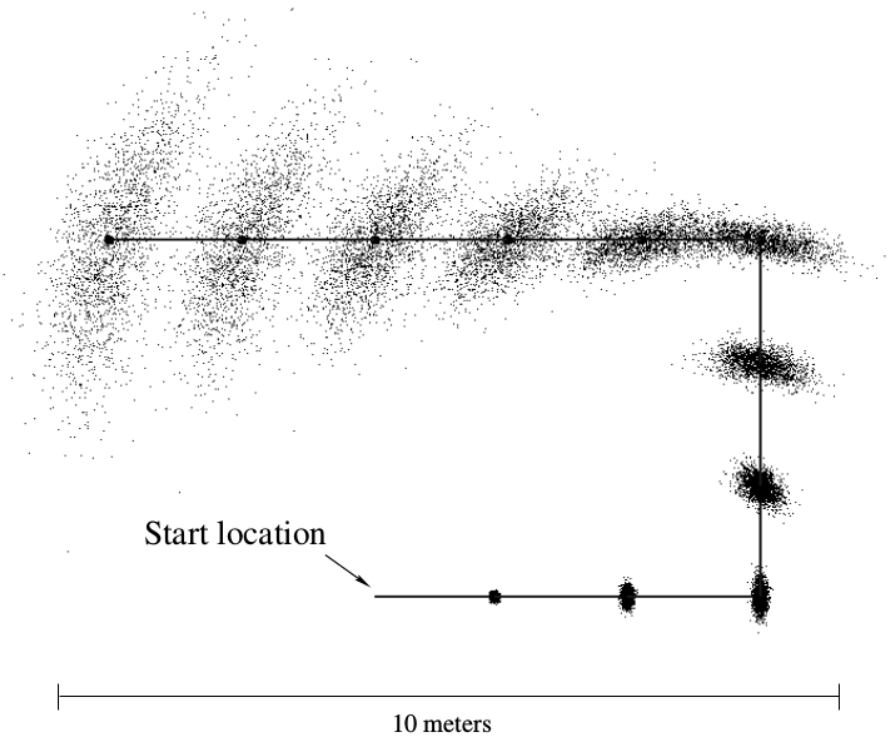


Figura 3.6: Paso de predicción aplicado sucesivas veces. Imagen extraída de Thrun et al. (2005).

Paso de actualización (del inglés *update*) Este paso ajusta los pesos de las muestras a partir de la función de densidad a la última información de sensado recibida. Para esto se le asigna a cada partícula un peso w_i proporcional a la verosimilitud de la información de sensado acorde al estado actual del sistema. Es decir,

$$w_i = p(z_t|x_t) \tag{3.7}$$

Este peso indica, en resumen, que tan verosímil es el modelo representado por la partícula en función de la última información de sensado. El conjunto de las partículas ponderadas representa la nueva creencia del filtro de Bayes. Este paso se presenta en la linea 5 del algoritmo 1.

Paso de remuestreo (del inglés *resampling*) Finalmente se ejecuta un tercer paso denominado remuestreo en el que se extraen con reposición N partículas del conjunto de partículas actuales, siendo la probabilidad de seleccionar una partícula proporcional a su peso w_i. Este paso forma parte del ciclo de iteración que se muestra en las líneas 7, 8, y 9 del algoritmo 1. Este último paso disminuye la incertidumbre de la estimación, es decir, disminuye la dispersión de las partículas. En otras palabras, se eligen N nuevas partículas a partir de las N actuales. La forma de elegir es sorteo ponderado. Eso permite que las partículas con mayor verosimilitud sean elegidas con mayor frecuencia. Nótese que una partícula puede ser elegida más de una vez, mientras otras pueden no se elegidas para el nuevo conjunto. El conjunto final suele tener muchas copias de partículas iguales.

Algoritmo 1: Algoritmo de filtro de partículas

Input: X_{t-1}, u_t, z_t

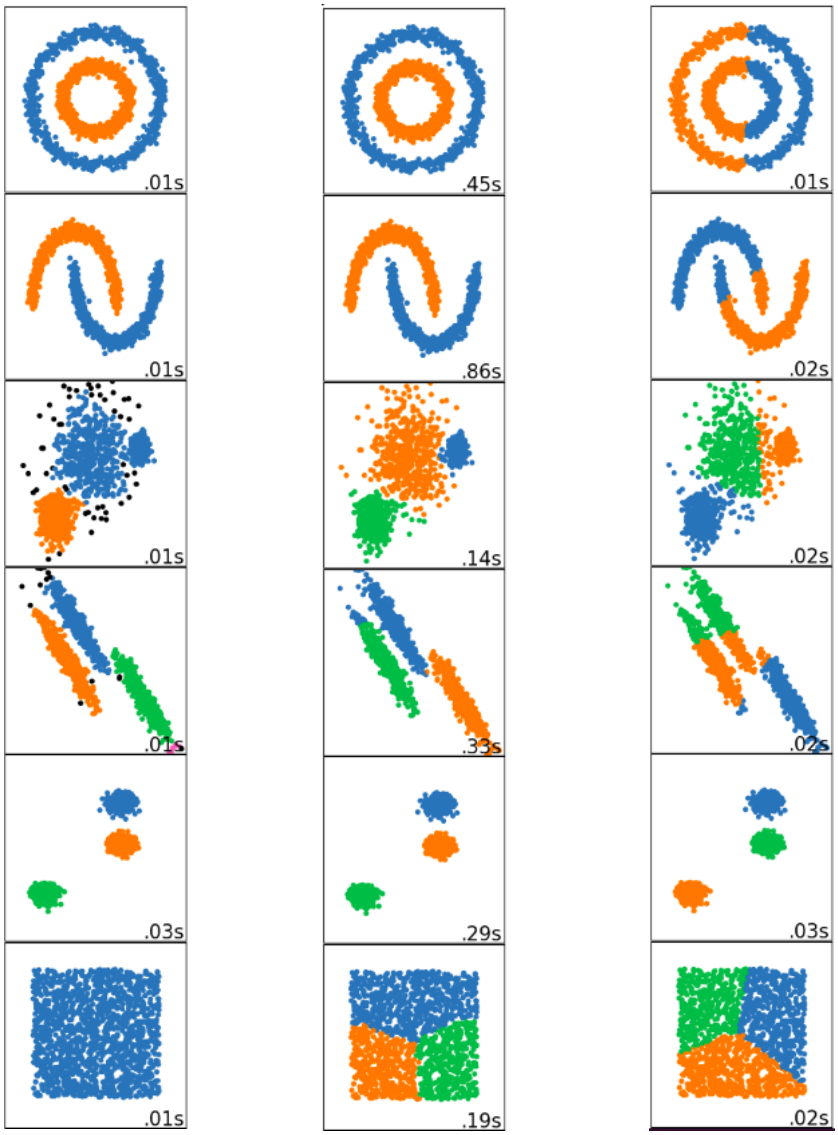
1 $\overline{X}_t = X_t = 0$;
2 **for** $m = 1$ to N **do**
3 $\quad$ *muestra* $x_t^{[m]} \sim p(x_t|u_t, x_{t-1}^{[m]})$;
4 $\quad$ $w_i^{[m]} = p(z_t|x_t^{[m]})$;
5 $\quad$ $\overline{X}_t = \overline{X}_t + \langle x_t^{[m]}, w_i^{[m]} \rangle$;
6 **for** $m = 1$ to M **do**
7 $\quad$ sorteo i con probabilidad $\propto w_i^{[m]}$;
8 $\quad$ agrego $x_i^{[m]}$ a X_t;
9 **return** X_t

3.4. Algoritmos de agrupamiento

Debido a que la cantidad de partículas utilizadas para mantener la estimación de la posición suele estar entre las miles o decenas de miles de partículas y que además realizar cálculos sobre miles de partículas es muy costoso, surge la necesidad de utilizar técnicas de agrupamiento para obtener un subconjunto de partículas que represente a la totalidad, pero que simplifique el procesamiento. En esta sección se presentan tres algoritmos de agrupamiento utilizados con la finalidad de formar subconjuntos de partículas a partir del conjunto de partículas del filtro. Los algoritmos que se presentan son DBSCAN, Kmeans++ y Spectral Clustering (Buitinck et al., 2013). En la figura 3.7 se muestra la aplicación de los tres algoritmos sobre diversos conjuntos de muestras similares a las formas que podría adoptar un grupo de partículas mientras el robot está navegando. En 3.7a se aplica DBSCAN sobre diversos conjuntos de muestras. Se debe tener en cuenta que este algoritmo no recibe el número de agrupamientos que debe formar como parámetro y que agrupa por densidad. En la figura 3.7b se aplica Spectral Clustering sobre el mismo conjunto de muestras. Si bien este algoritmo sí recibe como parámetro el número de agrupamientos, debido a su forma de agrupar, se respeta la distancia entre las muestras y la forma de los agrupamientos. Finalmente, en la figura 3.7c se presenta la agrupación realizada por el algoritmo Kmeans++ para el número de agrupamientos definido (dos para los dos primeros dos casos y tres para los siguientes cuatro casos).

3.4.1. Algoritmo DBSCAN

El algoritmo DBSCAN (Ester et al., 1996; Pedregosa et al., 2011) entiende agrupamientos como áreas de alta densidad separadas por áreas de baja densidad. Debido a esta forma de interpretar los agrupamientos, DBSCAN puede encontrar agrupaciones con formas arbitrarias, a diferencia de Kmeans++ que asume agrupamientos convexos. El componente principal de DBSCAN es el concepto de muestras *núcleo*, que son muestras que están en un área de alta densidad. En la figura 3.7a se muestra como el algoritmo agrupa las muestras según la concentración de las mismas sin importar la forma. Los agrupamientos se presentan formados por puntos de distintos colores (naranja, azul y verde), mientras que los puntos negros son marcados como ruido y no forman parte de ningún grupo.



(a) DBSCAN **(b)** Spectral Clustering **(c)** Kmeans++

Figura 3.7: En la figura se presenta el resultado de aplicar tres algoritmos de agrupamiento sobre distintos conjuntos de muestras. Figura extraída de Pedregosa et al. (2011).

DBSCAN recibe como entrada dos parámetros: *número mínimo de muestras* y *distancia máxima al núcleo*. El número mínimo de muestras define cuántas muestras son necesarias para formar un *núcleo* de muestras. Este parámetro controla principalmente la tolerancia del algoritmo al ruido, mientras que el parámetro distancia máxima al núcleo, que define la distancia máxima de separación que puede haber entre muestras del mismo núcleo, define la densidad esperada de los agrupamientos. Las muestras pueden ser etiquetadas como *núcleo* si en un radio de *distancia máxima al núcleo* hay al menos el *número mínimo de muestras* definido como parámetro, *borde* si no se cumple lo anterior pero hay al menos alguna muestra núcleo a menos de *distancia máxima al núcleo*, y *ruido* si no hay ninguna muestra núcleo en un radio de *distancia máxima al núcleo*.

DBSCAN es un algoritmo determinista si los datos son presentados siempre en el mismo orden. Sin embargo, si los datos cambian de orden (algo poco frecuente) ni las muestras etiquetadas como *núcleo*, ni las etiquetadas como *ruido* se ven afectadas, pero las muestras *borde* cercanas a más de un núcleo pueden variar su asignación (Ester et al., 1996).

3.4.2. Algoritmo Kmeans y Kmeans++

Kmeans (Arthur and Vassilvitskii, 2007; Pedregosa et al., 2011) es un algoritmo que agrupa muestras tratando de separarlas en grupos de igual varianza, minimizando el criterio conocido como inercia o distancia cuadrada al centro. Es un algoritmo muy utilizado en la literatura y tiene la virtud de escalar para números de muestras grandes.

El algoritmo divide el conjunto $X = \{x_1, x_2, ..., x_N\}$ de N muestras en k agrupamientos disjuntos C, cada uno conformado por la media μ_j $(1 \leq j \leq k)$. Las medias son llamadas centroides. El objetivo del algoritmo es elegir k centros de forma de minimizar la distancia entre cada muestra y el centroide como se presenta en la ecuación 3.8.

$$\sum_{i=0}^{n} \min_{\mu_j \in C} \parallel x_i - \mu_j \parallel^2 . \tag{3.8}$$

El algoritmo sufre de algunos inconvenientes. Uno de ellos es asumir que los agrupamientos son convexos. Esto presenta un resultado ineficaz frente a las muestras dispuestas en forma elongada o formas irregulares en general, como

se ve en la figura 3.7c. En espacios multidimencionales es necesario normalizar los valores previamente al uso del algoritmo, de lo contrario el algoritmo sufre de la *maldición de la dimensionalidad*.

Los k centros se eligen al azar entre las muestras. Si bien es un algoritmo rápido y simple, su éxito depende en gran medida de la inicialización de los centroides.

El funcionamiento completo de Kmeans se presenta en el algoritmo 2.

Algoritmo 2: Algoritmo Kmeans

1 Elección arbitraria de los k centros iniciales $C = c1, c2, , ck$;
2 Para cada $i \in 1, ..., k$, elegir al agrupamiento C_i como el conjunto de puntos de X más cercano a c_i que a c_j para todo los $j \neq i$;
3 Para cada $i \in 1, ..., k$, elegir c_i como el centro de masa de los puntos pertenecientes a $C_i : ci = \frac{1}{|C_i|} \sum_{x \in C_i} x$;
4 Repetir los pasos 2 y 3 hasta lograr convergencia.;

Existe una optimización de este algoritmo llamada Kmeans++ y se diferencia del anterior en la forma de elegir los centroides iniciales. El primer centroide se elige al azar, pero el resto se eligen tratando de que estén lo más alejados posible de los que ya fueron establecidos. La forma de elegir los centroides presentada en Arthur and Vassilvitskii (2007) deriva en resultados relevantes con respecto a su versión original.

3.4.3. Algoritmo Spectral Clustering

El algoritmo de agrupamiento Spectral Clustering (Ng et al., 2001; Pedregosa et al., 2011) se puede pensar como una combinación de los anteriores. Tiene la capacidad encontrar grupos con cualquier forma, teniendo en cuenta que requiere el número de agrupamientos como parámetro de entrada. Este algoritmo en una primera instancia genera una matriz de afinidad entre las muestras, luego baja la dimensionalidad de la matriz y sobre el resultado aplica Kmeans. Si bien el algoritmo funciona bien para un número pequeño de agrupamientos, no se recomienda para números grandes. En la figura 3.7b se presenta el resultado de aplicar el algoritmo aplicado a diversos conjuntos de muestras y el tiempo de ejecución en cada caso. Este algoritmo es un poco más lento que los presentados anteriormente.

Capítulo 4

Formulación del problema y presentación de la solución propuesta

En este capítulo se detalla la propuesta de solución al problema de localización activa en el marco del problema de seguimiento de posición. Como se mencionó en capítulos anteriores, esta tesis se basa en la propuesta *Active localization with dynamic obstacles* de Li et al. (2016). El trabajo realizado en el marco de esta tesis se puede dividir en dos partes, la primera parte consiste en la implementación completa del artículo que se utilizó como base (específicamente lo que refiere a la localización activa). La segunda parte versa sobre la adaptación de la propuesta al problema de seguimiento de posición (también conocido en la literatura como localización local). Esta adaptación motiva al autor a revisar la técnica de agrupamiento y la estrategia general utilizadas, y proponer modificaciones que permitan la adaptación mencionada.

Este capítulo se organiza de la siguiente manera: en la primera sección se presenta brevemente la formulación del problema de localización. En la segunda sección se presenta la motivación de la solución, se detalla la propuesta implementada (Li et al., 2016) y se reportan las adaptaciones realizadas. Finalmente, se explica el diseño y la implementación de estas adaptaciones.

4.1. Formulación del problema

Se quiere resolver el problema de localización de un robot móvil en un entorno conocido. Como se presentó en el capítulo 3.3.1 para este fin se requiere conocer la distribución de probabilidad del modelo de movimiento $p(x_t|u_t, x_{t-1})$ y la distribución de probabilidad del sensado $p(z_t|x_t)$. Dado que el robot no conoce su posición inicial, la distribución de probabilidad inicial es una distribución uniforme sobre todos los estados posibles.

Algoritmo 3: Algoritmo del filtro de Bayes

Input: $bel(x_{t-1})$, u_t, z_t

1 **forall** x_t **do**

2 $\quad \overline{bel(x_t)} = \int p(x_t|u_t, x_{t-1}) \times bel(x_{t-1}) dx_t$;

3 $\quad bel(x_t) = p(z_t|x_t) \times \overline{bel(x_t)}$;

4 **return** $bel(x_t)$

El algoritmo del filtro de Bayes (algoritmo 3) es la forma más general de calcular la creencia sobre la posición del robot a partir de la información de movimiento, sensado y el conocimiento inicial. Sin embargo este algoritmo no puede ser implementado directamente, ya que como se presenta en Thrun et al. (2005) el algoritmo del filtro de Bayes solamente puede ser implementado en problemas de estimación muy simples. En caso de espacios de estado grandes, razonar sobre infinitos estados hace computacionalmente imposible su aplicación directa. Luego existen diversos enfoques que restringen su aplicación a una cantidad de estados finita, o realizan asunciones específicas.

En el trabajo de Li et al. (2016) se realizan algunas simplificaciones del problema de localización. Se considera un robot móvil en un entorno cerrado, plano y conocido $E \in R^2$. El entorno E es representado con un mapa $\mathcal{M}$ compuesto por grillas de ocupación (Moravec and Elfes, 1985), donde cada celda puede tener uno de los siguientes estados $\{libre, ocupada\}$. El mapa puede ser cargado manualmente o construido utilizando algún algoritmo de SLAM (Grisetti et al., 2007; Llofriu and Andrade, 2012). Sin perder generalidad podemos modelar al robot como un punto del entorno que incluye una rotación. El estado inicial no es conocido. Para modelar la distribución de probabilidad $bel(x_t)$ del robot se utiliza un filtro de partículas (capítulo 3.3.2) que se actualiza a través del modelo de movimiento y sensado. Para mejorar la estimación de la pose en Li et al. (2016) se propone elegir una secuencia de poses $Q =< q_0, q_1, ..., q_n >$

que el robot debe recorrer. En la sección 4.2.2 se presentan los detalles del trabajo de Li et al. (2016).

Este trabajo de tesis plantea hipótesis similares a las de Li et al. (2016), diferenciándose principalmente en que la posición inicial $bel(x_{t=0})$ es conocida, y que el robot tiene un conjunto de poses preestablecidas $< p_1, p_2, ...p_n >$ que debe recorrer, pero no con el fin de localizarse (como en el caso de Li et al. (2016)), sino como parte de cumplir una tarea.

4.2. Presentación de la solución

En esta sección se presenta la solución desarrollada en esta tesis.

4.2.1. Motivación

En el trabajo de Li et al. (2016) se resuelve el problema de localización global utilizando una estrategia de localización activa que elimina hipótesis en forma sistemática. Si bien la estrategia es novedosa, en el contexto de robots de servicio es más adecuado pensar en un robot inicialmente localizado, dado que el robot se mueve en un entorno conocido y controlado como puede ser una vivienda, un supermercado o un gran almacén. Luego se cree que es más adecuado pensar en el problema de seguimiento de posición con la finalidad de mejorar la precisión en la ubicación del robot, derivando en una mayor eficiencia en la ejecución de tareas, como por ejemplo llevar una bandeja con el desayuno desde la mesada de la cocina hasta la mesa del comedor.

Se considera entonces un robot en un entorno conocido, con un punto de salida y un punto de llegada definidos (y eventualmente puntos intermedios por los que debe pasar), que sigue una trayectoria definida hacia la meta. Se asume, como se presentó en el capítulo 2, que durante el recorrido existen lugares con características relevantes que en caso de ser observados por el robot, podría mejorar la estimación de su posición. En este contexto se quiere elegir el conjunto de acciones a corto plazo que permita al algoritmo de localización mejorar la estimación de su ubicación a través de sensar lugares específicos del mapa los cuales se estima que aportan información sobre la pose del robot.

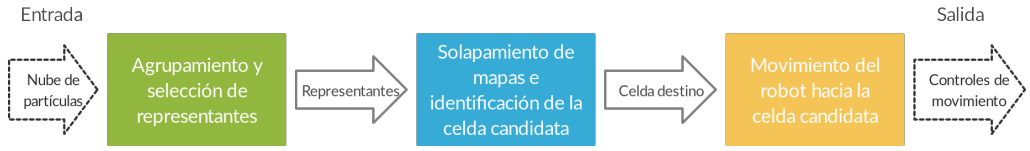


Figura 4.1: Diagrama de etapas del algoritmo de localización activa.

4.2.2. Localización activa global a partir de la eliminación sistemática de hipótesis

En esta sección se describe la propuesta de Li et al. (2016), desde el punto de vista del proceso de localización activa. Parte del trabajo de esta tesis fue implementar la propuesta.

Descripción general

La idea general de este algoritmo es utilizar como dato de entrada la estimación generada por el filtro de partículas (presentado en el capítulo 3.3.2) que mantiene el módulo de localización pasiva, y particionar a la nube de partículas en grupos que modelan las diversas hipótesis sobre la pose del robot. Los centroides de cada agrupación son llamados *representantes* y se utilizan para armar un mapa compuesto que combina la estimación de cada agrupación. Luego, se busca en el mapa compuesto los puntos que generan mayor discrepancia entre los representantes (celdas a las que la mitad de los representantes consideren que están libres y la otra mitad de los grupos consideren ocupadas). Se entiende que si el robot sensa celdas con esta característica, las hipótesis que difieren en el estado de la celda tienden a desaparecer, mientras que las otras tienden a mantenerse. Finalmente, se elige la celda del conjunto que optimice la cantidad de hipótesis que se descartarán (la mayor posible) y la distancia relativa al robot (la menor posible), y se dirige al robot hacia la celda elegida. Una vez alcanzado el punto, el ciclo comienza nuevamente. En la figura 4.1 se muestra un diagrama con las etapas más importantes del algoritmo. Como dato de entrada el algoritmo toma la nube de partículas, luego procesa la información en tres etapas diferentes, y finalmente el algoritmo devuelve los controles que llevan al robot a la celda elegida.

Es importante destacar que este algoritmo funciona como complemento al algoritmo de localización pasiva subyacente (el paquete de ROS llamado

AMCL).

A continuación se describe la forma de agrupar a las partículas, la generación del mapa compuesto y el cálculo del punto de destino.

Agrupamiento

Inicialmente las partículas son distribuidas uniformemente en todo el espacio libre disponible en el mapa conocido. A medida que el robot se mueve las partículas naturalmente se concentran en varios lugares del mapa, en el que cada uno representa una posible zona donde el robot podría estar. Nótese que estos agrupamientos dependen de la posición inicial del robot y la información que obtenga el robot del entorno a medida que se mueve. Luego, se aplica sobre las partículas un algoritmo de agrupamiento para concentrar la estrategia de control en unas pocas posiciones candidatas que resuman la información de todas las partículas. El algoritmo de agrupamiento utilizado es DBSCAN descripto en el capítulo 3.4.1. Para mejorar la eficiencia de la aplicación del algoritmo en tiempo real, Li et al. (2016) deciden evaluar solo el 30 % de las partículas. Cada uno de los grupos generados por DBSCAN es considerado como una hipótesis factible sobre la pose actual del robot. Para evitar trabajar con cada una de las partículas, de cada agrupación se identifica el centroide, y se utiliza como el representante de la agrupación. Se asume que este representante es la partícula que tiene mayor similitud con las de su agrupación.

Mapa compuesto

A partir del conjunto de representantes, se elige aleatoriamente uno como fijo, y se transforma el mapa asociado a cada uno de los restantes representantes al marco de coordenadas del que se eligió fijo, superponiéndose cada uno de los mapas uno encima del otro. Luego, se colapsan todos los mapas en uno nuevo, formando un mapa compuesto. Este mapa muestra las diferencias en los objetos estáticos del mapa que se espera observar por las hipótesis de cada grupo en un marco de referencia común. Formalizando lo expresado, dada una pose (x_f, y_f, θ_f) que será tomada como representante fijo para la generación del mapa compuesto, y dada otra pose (x, y, θ) perteneciente a otro representante, se aplica la siguiente matriz de transformación a cada uno de los puntos del mapa de la última pose:

Figura 4.2: Ejemplo de mapa compuesto. En este ejemplo participan seis representantes.

$$T = \begin{pmatrix} \cos(\theta - \theta_f) & \sin(\theta - \theta_f) & -(x - x_f) \\ -\sin(\theta - \theta_f) & \cos(\theta - \theta_f) & -(y - y_f) \\ 0 & 0 & 1 \end{pmatrix}. \tag{4.1}$$

El mapa compuesto obtenido como resultado es similar a un mapa de grillas probabilístico, pero con la diferencia de que en las celdas se almacenan números enteros en lugar de probabilidades. En cada celda se almacena el número de grupos cuyo mapa transformado presentan un obstáculo en dicha celda. Es similar a una votación donde cada representante aporta un voto si la celda está ocupada y ninguno si está libre. Luego, para maximizar la cantidad de partículas que serán descartadas, el robot se deberá mover hacia una posición del mapa compuesto que esté libre para algunos representantes pero no para todos.

En la figura 2.3 del capítulo 2, se presenta un ejemplo de como se arma el mapa compuesto. Por un lado se muestran dos poses, cada una con su mapa asociado. Dejando la primera como fija, se lleva la segunda a la posición de la primera, aplicando una traslación y rotación sobre la partícula. Esta transformación también es aplicada a todos los puntos del mapa asociado a la segunda partícula. Una vez que se solapan los mapas, se integra la información en un único mapa como se describió previamente.

En la imagen izquierda de la figura 4.2 se muestra un ejemplo simple de un mapa compuesto. En este ejemplo el número total de representantes es seis. Las cedas cuyo valor es seis son marcadas como ocupadas, dado que los seis

representantes coincidieron en que estaba ocupada (cada uno vota uno). En las celdas cuyo valor es cero, todos coincidieron en que la celda está vacía. Luego, existen celdas donde dos representantes votan que hay obstáculo y cuatro que no, y finalmente algunas que valen tres donde la mitad proponen que la celda está ocupada y la otra mitad que está libre. El mapa que aparece a la derecha de la imagen muestra una interpretación visual del mapa, donde negro significa ocupado, blanco libre, y los valores de grises intermedios están asociados a la cantidad de votos que tuvo cada celda. Para realizar esta visualización, se normalizaron los valores de las celdas en base a los utilizados por los mapas de grillas en ROS.

Cálculo del punto destino

Sea L el conjunto de celdas pertenecientes al mapa compuesto que presentan mayor discordancia (si nos basamos en el ejemplo de la figura 4.2, serían valores cercanos a tres), se elige el punto $l \in L$ al que se moverá el robot según los siguientes criterios:

- el número esperado de partículas que se descartarán $DP(l)$. Este cálculo puede hacerse simulando la acción para cada partícula;
- la distancia recorrida $d(l)$, que corresponde a la distancia entre el marco de referencia fijo del mapa compuesto y l.

Ambos criterios son combinados en la función de optimización 4.2 que se presenta a continuación:

$$l^* = \arg\max_{l \in L} \Big[\alpha \tilde{DP}(l) + (1-\alpha)\hat{d}(l)\Big], \tag{4.2}$$

donde α es el peso asociado a cada uno de los criterios, y $\tilde{DP}$ y $\hat{d}$ son los valores normalizados entre 0 y 1 de DP y d respectivamente.

Una vez obtenido el punto l^*, como se mencionó previamente se dirige al robot hacia éste.

4.2.3. Adaptación de la localización activa global al problema de seguimiento de posición

El autor considera que el trabajo presentado en la sección anterior propone una estrategia valiosa en el campo de la localización activa. En el marco del

estudio de robots de servicio, parece relevante adaptar la propuesta para el problema en que el robot se encuentra localizado. La principal justificación viene dada porque en el contexto de una casa, museo u hospital, el mapa es conocido y además, sería razonable considerar que se puede especificar con precisión la posición inicial del robot antes de iniciar su actividad. Estas hipótesis, mapa conocido y posición inicial conocida, transforman el problema de localización global en un problema de seguimiento de posición (o localización local).

Para adaptar la propuesta descrita en la sección 4.2.2 al problema de seguimiento de posición se requiere tener en cuenta varias consideraciones. La primera es que dado que el robot ya cuenta con un planificador de trayectorias que genera rutas óptimas, no es deseable modificar la ruta a menos que sea realmente necesario. Esto lleva a que la ejecución de la rutina de localización activa sea local temporalmente a los momentos en los que la incertidumbre sobre la pose supera un umbral determinado. Asociado a lo anterior, la segunda consideración consiste en que el algoritmo de agrupamiento utilizado en la propuesta original no logra generar grupos cuando la nube de partículas presenta continuidad en términos de distancia (este punto será explicado en la próxima subsección), lo que limita la ejecución de los pasos restantes del algoritmo. En tercer lugar, y en concordancia con el primer argumento, la generación de un mapa compuesto utilizando todo el mapa del entorno es muy costoso en términos de cómputo, y además, si los puntos que mejoran la localización están lejos del entorno en el que se encuentra el robot, no parece razonable considerarlos para no desviar demasiado al robot del camino únicamente para mejorar su localización. Finalmente, las acciones que debe realizar el robot deben ser modificadas para estar en concordancia al punto anterior, es decir, evitar la navegación fuera del recorrido establecido por el planificador global.

El proceso descrito anteriormente se ilustra gráficamente en la figura 4.3. En la figura 4.3a se muestra como se agrupa la nube de partículas generando dos grupos. La figura 4.3b muestra la generación del mapa compuesto a partir de dos poses distintas, dejando la que está en el cuadro superior fija y transformando la del cuadro inferior. Luego, en la figura 4.3c se muestra una secuencia en la cual el robot se detiene porque su dispersión supero el umbral preestablecido (cuadro superior) y se ejecuta la rutina de localización activa. En el cuadro del medio el robot realizó la rotación hacia la izquierda para sensar la celda candidata y en consecuencia disminuye la incertidumbre de la estimación sobre la pose. Finalmente, en el cuadro inferior el robot retoma el camino. El

(a) agrupamiento (b) solapamiento (c) aplicación de la acción

Figura 4.3: Adaptación de la propuesta.

autor quiere hacer notar que si bien las figuras 4.3a, 4.3b y 4.3c se utilizan para ilustrar una secuencia de eventos, las figuras mencionadas no forman parte de una única corrida, sino que fueron extraídas de distintos experimentos.

Agrupamiento

En el problema de seguimiento de posición, inicialmente las partículas se encuentran concentradas en la zona del mapa donde se encuentra el robot. Si a medida que el robot se mueve obtiene información de sensado relevante, las partículas se mantienen cercanas entre ellas, es decir que la estimación de la pose mantiene un nivel de incertidumbre bajo. Cuando el robot circula por espacios amplios con pocas características o espacios muy simétricos, las partículas tienden a separarse. Sin embargo, dado que esto sucede por periodos cortos (p.ej: si pensamos su aplicación en casas, museos, almacenes), la separación espacial que ocurre entre las partículas hace que la aplicación del algoritmo de agrupamiento DBSCAN sea ineficaz, dado que este algoritmo busca grupos separados a una distancia mayor que el parámetro *distancia máxima al núcleo* (como se explica en el capítulo 3.4.1), algo que ocurre con una frecuencia muy baja en el contexto de seguimiento de posición aplicado a robots de servicio. Debido a que el autor quiere forzar la aplicación del algoritmo con el objetivo de minimizar la dispersión de las partículas (que equivale a minimizar la incertidumbre), es necesario que se aplique un algoritmo de agrupamiento que pueda agrupar a las hipótesis por similitud aunque estén *cercanas* entre ellas. Luego, se recurre a otros algoritmos de agrupamiento que permitan segmentar el conjunto de partículas con las características presentadas.

Una de las formas de resolver el problema es utilizar algoritmos de agrupa-

miento que reciban como parámetro la cantidad de grupos que se debe formar. Luego de haber relevado diversos algoritmos de agrupamiento, se eligieron Spectral Clustering y Kmeans++ (sección 3.4) por razones que se justifican a continuación. Una de ellas es que permiten definir a priori la cantidad de grupos a generar. Además Spectral Clustering en particular tiene características similares a las de DBSCAN en el sentido de agrupar partículas que están cerca. Por otro lado, Kmeans++ es un algoritmo muy rápido en términos computacionales. Ambos algoritmos fueron presentados en el capítulo 3.4.

El cálculo del número de grupos a formar se obtuvo experimentalmente y está asociado a la dispersión de las partículas. Luego de realizar varios experimentos con la propuesta original, se extrajo un umbral máximo de dispersión. Una vez superado ese umbral se generan al menos dos grupos. Además, la cantidad de grupos crece proporcionalmente a medida que crece la dispersión por encima del umbral. Estos aspectos se presentan más adelante en la subsección 4.2.3.

Tamaño del mapa

En la propuesta original se procesa el mapa completo debido a que el robot potencialmente puede estar en cualquier parte del mapa. En el contexto de esta tesis, por hipótesis tenemos una idea aproximada de la pose del robot, por lo que podemos trabajar con un entorno de celdas reducido y contribuir a minimizar el costo computacional. Además, en este trabajo el robot sigue un camino establecido por el planificador y no es el espíritu de la tesis desviar al robot de su ruta, sino que por el contrario se espera minimizar la incertidumbre con el menor impacto posible sobre la misión del robot. El tamaño de dicho entorno queda determinado por el alcance de los sensores que lleva el robot, lo que permite que pueda sensar las celdas de mayor relevancia solamente *observando* dichas celdas (rotar de forma de que sus sensores apunten a estas celdas - esto se puede hacer o bien rotando al robot, o bien rotando solamente los sensores sin afectar la base del robot).

Selección de acciones

Retomando lo expuesto en el punto anterior sobre el tamaño del mapa, y teniendo en cuenta que las celdas que se evalúan están dentro del rango alcanzable por los sensores del robot, basta con realizar movimientos rotacionales

para sensar las celdas del entorno reducido. Luego, el conjunto de acciones entre los que se puede elegir son el conjunto de rotaciones posibles. En particular se elige la rotación más corta que alinee el frente del robot (el centro del sensor láser) con el punto a observar. Se recuerda al lector que por el funcionamiento del filtro de partículas, cuantas más observaciones se obtenga del punto, más ciclos de predicción, actualización y remuestreo se ejecutarán, permitiendo que las partículas con mayor verosimilitud a las observaciones persistan y cobren más relevancia frente a las que no, que eventualmente desaparecerán. Este comentario refiere a que incluso si el punto está dentro del rango sensado, sensarlo varias veces permite ajustar la estimación del filtro.

Condiciones de disparo de la localización activa

En el trabajo de Li et al. (2016) la primera condición que se debe cumplir para la ejecución del algoritmo es que DBSCAN genere al menos dos agrupamientos. Si esta condición se cumple, se procede a crear el mapa compuesto y se obtiene el conjunto de puntos L. Luego, la segunda condición es que L no sea vacío. Si hay al menos un elemento $l \in L$, el robot tiene un punto al cual dirigirse y el algoritmo prosigue.

A diferencia de la propuesta original, en este trabajo de tesis se utiliza la dispersión de las partículas como primera condición de disparo del algoritmo. En caso de que la dispersión no supere el umbral preestablecido (equivalente a que la incertidumbre sobre la estimación de la pose sea baja), el algoritmo no prosigue. Por el contrario, si la dispersión supera el umbral, el algoritmo prosigue y se calcula la cantidad de agrupamientos que se debe generar según el porcentaje de dispersión. Luego, la segunda condición es análoga para ambos trabajos. En las ecuaciones 4.3, 4.4 y 4.5 se presenta un esquema simplificado con la secuencia de pasos de la condición de disparo. Como datos de entrada del módulo AMCL de ROS se tiene la nube de partículas y una matriz de covarianzas. En primer lugar se calcula la dispersión a partir de la varianza de cada componente:

$$dispersion = \sqrt{var_x} + \sqrt{var_y} + \mu \times \sqrt{var_\theta}, \tag{4.3}$$

donde μ es un factor de normalización para la rotación. Luego, se calcula el porcentaje de dispersión con referencia al valor máximo $MAX\_DISPERSION$ obtenido en forma experimental:

$$\%dispersion = \frac{dispersion}{MAX\_DISPERSION} \times 100. \tag{4.4}$$

Finalmente se calcula la cantidad de grupos a formar:

$$\#clusters = \lfloor \frac{\%dispersion}{\%cluster\_dispersion} \rfloor, \tag{4.5}$$

donde $\%cluster\_dispersion$ es el porcentaje asociado a un agrupamiento. Debido a que la propuesta de Li et al. (2016) y el trabajo de esta tesis se enfocan en condiciones iniciales diferentes, localización global y localización local respectivamente, es razonable que la condición de disparo tenga diferencias. Los parámetros de dispersión máxima ($MAX\_DISPERSION$) y porcentaje de dispersión por agrupamiento ($\%cluster\_dispersion$) fueron obtenidos empíricamente a partir del estudio de la evolución de los mismos en la propuesta original.

4.2.4. Diseño de la solución

La solución fue diseñada utilizando el software ROS (Quigley et al., 2009), desarrollado para facilitar el diseño y la implementación de software aplicado a robots. Presenta como principales ventajas la abstracción del hardware, control de dispositivos de bajo nivel, intercambio de mensajes sincrónicos y asincrónicos entre módulos, sistema distribuido, implementación de funciones básicas, y una creciente comunidad a nivel mundial. La comunicación principal entre los módulos, llamados nodos, es a través de publicación y suscripción a tópicos o servicios donde se intercambian mensajes. Los paquetes de ROS más relevantes utilizados en esta tesis son: SMACH (Bohren and Cousins, 2011), una arquitectura basada en niveles de tareas que permite construir comportamientos complejos para robots, basada en el uso jerárquico de máquinas de estado; AMCL (Fox, 2001) es un algoritmo de localización pasiva adaptativo basado en técnicas de Monte Carlo; y la PILA DE NAVEGACIÓN (Koubaa, 2016) que incluye varios componentes como planficador global, planificador local entre otros.

En la figura 4.4 se presenta un diagrama simplificado de la implementación realizada. La imagen muestra el nodo de agrupamiento, el nodo de procesamiento del mapa y cálculo de la celda candidata, y el nodo de selección de acción. Adicionalmente se presenta el nodo que lleva el control de las métricas

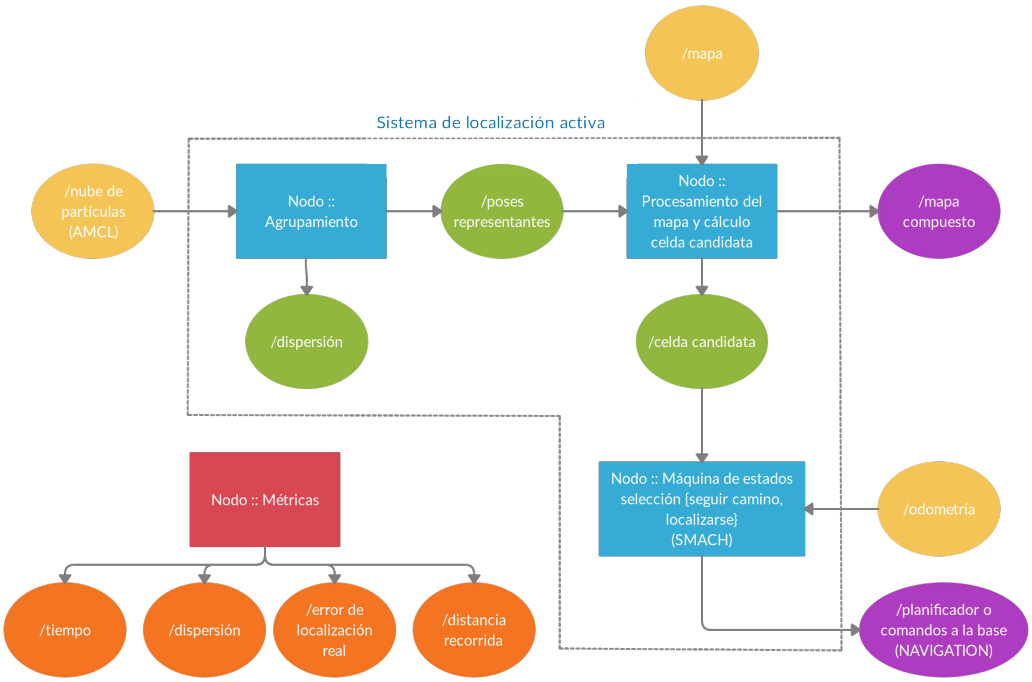


Figura 4.4: Diagrama del sistema implementado.

utilizadas, que también fue implementado en el marco de esta tesis. A continuación se describen las funciones de cada uno de los nodos que aparece en la figura.

Nodo :: Agrupamiento, este nodo tiene como entrada de datos la nube de partículas (mantenida por el módulo de localización pasiva AMCL). En primer lugar se filtra un tercio de las partículas para minimizar el costo computacional, tal cual se hace en la propuesta original. A continuación se normaliza cada componente del vector (x, y, θ) (que representa a la partícula) en el intervalo [0..1]. Luego se obtiene la dispersión de las partículas y a partir de ese valor se calcula la cantidad de agrupamientos que se deben formar, y se aplica el algoritmo de agrupamiento elegido (Spectral Clustering o Kmeans++) sobre los datos normalizados. Finalmente se calcula el medioide (la propuesta original calcula el centroide, pero parecía razonable que el representante fuera un elemento del conjunto) de cada agrupamiento. La salida de este nodo es un arreglo de poses, donde cada pose es la representante de alguna agrupación.

Nodo :: Procesamiento del mapa y cálculo celda candidata, en este nodo se recorre el arreglo de representantes y se genera una transformada

que lleva a cada representante y su mapa asociado a un marco común. Luego se colapsan todos los mapas generando el mapa compuesto. Se revisa cada una de las celdas en un entorno reducido alrededor del marco de coordenadas común, y se busca la celda que genera mayor discrepancia. Si hay varias, se elige a las más cercana (la que implique menos rotación). La celda resultado es publicada y será la entrada del siguiente nodo.

Nodo :: Selección de acción, es un nodo que decide entre ejecutar una misión preestablecida o realizar una acción de localización activa para observar a la celda candidata publicada por el nodo descripto previamente. En general el robot va a tener una misión establecida, que puede ser ir de un punto a otro del mapa, o seguir una secuencia de puntos (como es el caso en los experimentos realizados en esta tesis) que podría asociarse a tareas de limpieza, vigilancia o similar. Mientras el robot está siguiendo puntos, se utiliza al planificador para definir la ruta a seguir. En el momento que se recibe una celda candidata (que es el resultado de que la dispersión de las partículas superó el umbral y que a su vez existe una celda que aporta información relevante para eliminar hipótesis), el robot se detiene y se dispara un comportamiento reactivo que lo hace rotar para alinearse con la celda recibida. Una vez hecho esto, se vuelve a solicitar al planificador una nueva ruta hacia el punto que se estaba yendo antes de detener al robot. Si durante la recorrida del robot, en ningún momento la dispersión supera el umbral establecido, el algoritmo de localización activa no se ejecuta.

Nodo :: Métricas, adicionalmente se implementó un nodo de ROS que lleva el conteo del tiempo transcurrido, la distancia recorrida, la dispersión instantánea, y el error entre la estimación y la posición real del robot, realizando además su persistencia en archivos CSV (valores separados por coma). Además, genera automáticamente las gráficas asociadas a cada ejecución.

Capítulo 5

Evaluación experimental y resultados

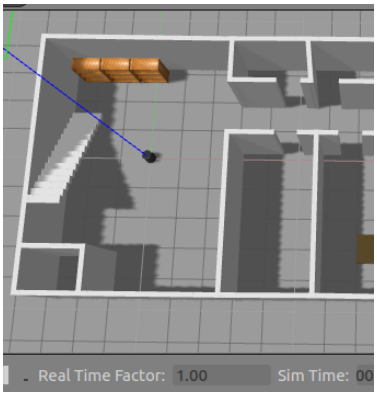
En este capítulo se reportan los resultados experimentales de la aplicación de la técnica de mapas solapados desarrollada en esta tesis al problema de seguimiento de posición. Se realizaron experimentos simulados y experimentos en un robot real, en distintos escenarios y comparando cuatro estrategias diferentes. El capítulo se divide en dos secciones, en la primera sección se presentan los experimentos simulados y se reportan los resultados asociados, y en la segunda sección se muestran los experimentos y resultados realizados con un robot real. Además, en cada caso se hace un breve análisis de los resultados reportados.

5.1. Experimentos simulados

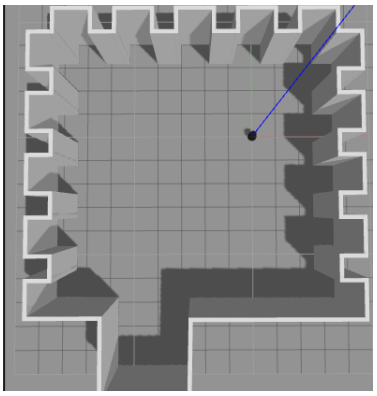
El mayor volumen de experimentos fue realizado en un entorno simulado y son reportados en esta sección. Los experimentos realizados en el robot real, que se presentan en la sección 5.2, tienen un carácter confirmatorio, y por eso fueron realizados en menor volumen, además de la dificultad intrínseca asociada al uso del hardware.

5.1.1. Simulador

El simulador Gazebo (Koenig and Howard, 2004) se presenta como uno de los más utilizados en la literatura asociada al desarrollo de software para robots y por eso fue elegido para la etapa de evaluación de este trabajo. Una de las



(a) Escenario INCO



(b) Escenario simétrico

Figura 5.1: Escenarios utilizados en los experimentos simulados.

principales ventajas es que debido a su creciente uso por la comunidad, existen múltiples robots y escenarios disponibles para su uso. Además está integrado a ROS, lo que facilita su uso durante las etapas de prototipado y evaluación experimental.

5.1.2. Escenarios

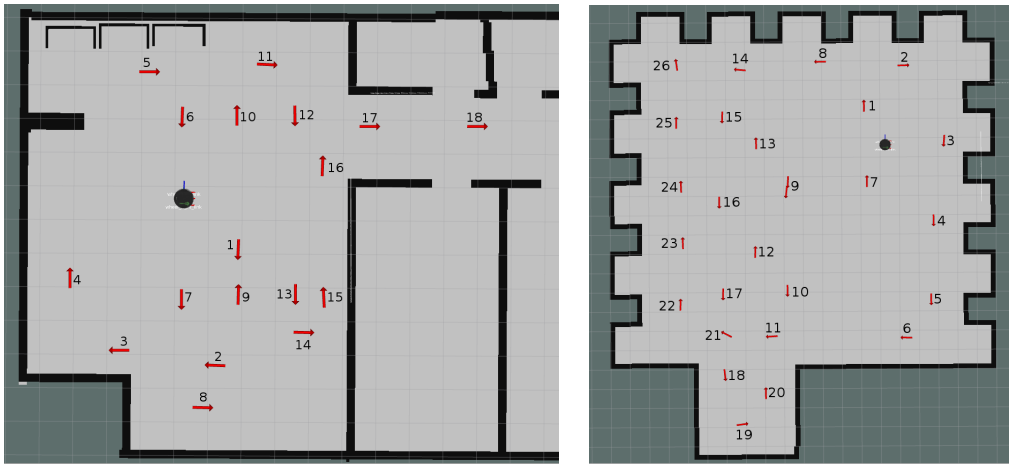
Se utilizaron dos escenarios de diferentes características para el desarrollo de los experimentos simulados. En la figura 5.1 se muestran los dos escenarios: INCO y un espacio abierto con características simétricas, que llamaremos escenario simétrico. El escenario INCO de la figura 5.1a es una representación de una parte del piso cero del Instituto de Computación de la Facultad de Ingeniería (en adelante INCO) y consiste en un espacio de oficinas con características similares a las de una vivienda, en el sentido que cuenta con mobiliario similar, pasillos, oficinas, entre otros. Las dimensiones del espacio utilizado son 7,8 metros de largo por 5,8 metros de ancho y adicionalmente se utilizó una parte del pasillo de 5 metros de largo por 1,5 metros de ancho. En la figura 5.1b se presenta un espacio amplio, con características simétricas que podría ser un museo, un gran almacén o galpón, escenarios donde podría utilizarse un robot de servicio. Las dimensiones del escenario simétrico son 10,5 metros de largo por 11,1 metros de ancho aproximadamente.

5.1.3. Metodología

Para validar la adaptación del algoritmo de localización activa propuesta en este trabajo, se realizaron experimentos con las dos variantes de algoritmos de agrupamiento elegidas durante el desarrollo de la solución: Spectral Clustering y Kmean++. Además, se incluyó una versión con el algoritmo de agrupamiento de la propuesta original DBSCAN, y otra versión que utiliza localización pasiva exclusivamente. En resumen, se comparan tres algoritmos de localización activa y uno de localización pasiva. Debido a la naturaleza estocástica del modelo de movimiento y modelo de sensado del robot, se ejecutaron diez corridas con cada una de las variantes del algoritmo en cada uno de los escenarios, totalizando 80 ejecuciones. En cada escenario se estableció una pose de partida fija y además una secuencia de poses por las cuales el robot debe pasar. El propósito del recorrido es que el robot se mueva por todo el espacio tanto pasando cerca de objetos (como paredes, armarios, etc.) así como por lugares más abiertos, simulando el movimiento que hace un robot de servicio cuando realiza tareas en un hogar, como puede ser buscar un objeto o persona, o ir a un punto específico a recoger algo y llevarlo a otro lugar, entre otras. En la figura 5.2 se presenta la secuencia de poses que debe seguir el robot para cada uno de los escenarios. La base de cada flecha roja representa el punto (x, y) del mapa al que el robot debe navegar, y el ángulo θ con el cual debe llegar queda determinado por la dirección de la flecha. El índice que acompaña a cada flecha determina el orden en el que le son presentadas las poses al robot.

Para los experimentos se utilizó en valor por defecto de la cantidad de partículas del módulo AMCL (mínimo 500, máximo 2000). Si bien en la propuesta original se adapta la cantidad de partículas al tamaño del escenario, esto no es necesario para el problema de seguimiento de posición ya que no se espera cubrir todo el escenario con partículas sino una pequeña porción cercana al robot. Para el agrupamiento se procesa solamente el 30 % de las partículas, tal cual se realiza en la propuesta original. Como se mencionó en el capítulo 4.2.3, existe un umbral de dispersión calculado experimentalmente que genera la activación del algoritmo de localización activa.

En la simulación fue utilizado un robot Turtlebot (Garage, 2011), que consiste en una plataforma móvil, diferencial, equipada con un sensor de profundidad Astra con un rango de trabajo entre 0.45 metros y 4.5 metros, y una apertura horizontal de 70 grados. En sensor Astra se utilizó como si



(a) Escenario INCO. **(b)** Escenario simétrico.

Figura 5.2: Secuencia de poses por escenario.

fuera un sensor láser (la transformación fue hecha utilizando el nodo ROS *deepimage_to_laserscan*). Tanto el simulador Gazebo, como ROS y el resto de los componentes fueron ejecutados en una laptop ASUS Zenbook UX305 con un procesador Intel Core M3-6Y30 de 900Mhz y 8GB de memoria RAM.

5.1.4. Evaluación del desempeño de los algoritmos de localización

La literatura (Burgard et al., 1997; Li et al., 2016; Thrun et al., 2005; Inoue et al., 2016) presenta al error de localización y a la incertidumbre sobre la estimación como las dos métricas principales para evaluar el desempeño de los algoritmos de localización activa. La principal es el error de localización (del inglés *ground truth error*) que consiste en la distancia entre la posición real del robot y la estimación del algoritmo de localización. En forma complementaria la incertidumbre se entiende como la confianza que tiene el algoritmo sobre su estimación. Como métricas adicionales interesa conocer el tiempo que demora el robot en completar la misión y la distancia recorrida, porque nos permiten conocer con mayor detalle el comportamiento del algoritmo (Li et al., 2016; Velez et al., 2011).

El error de localización y la incertidumbre se describen con más detalle en las subsecciones siguientes.

Error de localización

En el contexto de la robótica móvil el término *ground truth* se utiliza para referirse a la posición real del robot. La expresión *ground truth error* (en adelante error de localización), presentado en la ecuación 5.1, es la diferencia entre la estimación de la posición del robot que hace el algoritmo de localización y la posición real del robot. En el caso de un filtro de partículas, la estimación de la posición que realiza el algoritmo es la media ponderada de cada partícula (si bien se podrían utilizar otros estimadores). Por otro lado la posición real del robot se mide generalmente con algún dispositivo externo al sistema, normalmente cámaras con visión global del entorno. La diferencia se mide utilizando la distancia euclidiana en $\mathbb{R}^2$.

$$errorLoc_{\mathbb{R}^2} = \sqrt{(x_{real} - x_{estim.})^2 + (y_{real} - y_{estim.})^2} \tag{5.1}$$

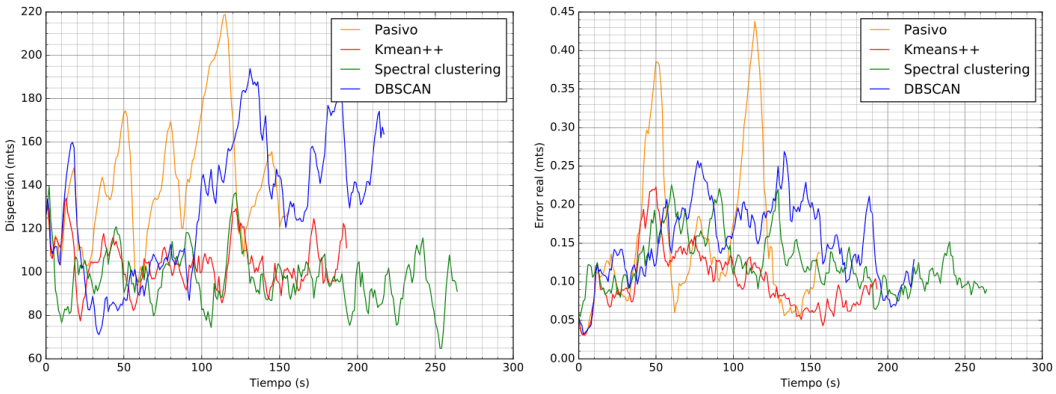
Incertidumbre del sistema

La incertidumbre sobre la estimación de la posición es una medida de la confianza que tiene el algoritmo sobre la estimación. Para el caso de seguimiento de posición se puede modelar la estimación con una distribución de probabilidad gaussiana y utilizar la varianza como medida de dispersión (Thrun et al., 2005). Para el caso general de localización global se utiliza la entropía (denotada en la literatura como H), que es una medida del desorden del sistema.

En este caso se utiliza la varianza como medida de incertidumbre. En el caso de que la varianza sea pequeña, significa que las partículas están muy cercanas a la media, por lo que aumenta la confianza del algoritmo en la estimación que mantiene sobre la pose. Sin embargo, cuando las partículas están muy dispersas, la varianza es mayor, luego la confianza del algoritmo sobre la estimación también es baja.

5.1.5. Resultados

En la presente sección se muestran los resultados experimentales de las ejecuciones en cada uno de los escenarios. Además se resumen todos los resultados en dos tablas y se realiza un breve análisis de los mismos.



(a) Dispersión **(b)** Error de localización

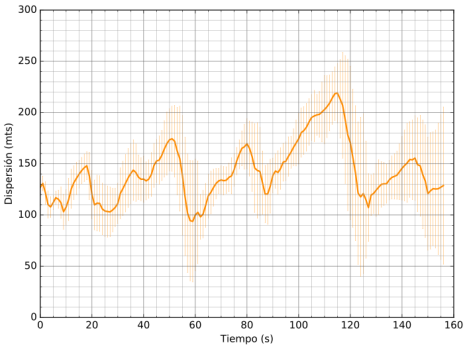
Figura 5.3: Resultados experimentales simulados en el escenario INCO.

5.1.6. Resultado de las ejecuciones en el escenario INCO

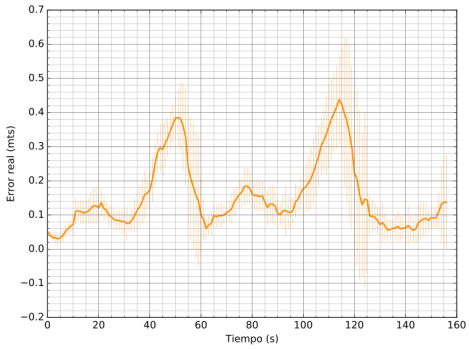
En la figura 5.3a se muestra la evolución promedio de la dispersión en cada instante de tiempo para cada uno de los algoritmos. La gráfica promedio se generó realizando un promedio instantáneo de cada una de las diez corridas de un mismo algoritmo. En todos los casos, todas las corridas completaron la misión en su totalidad. Por otro lado en la figura 5.3b se grafica el error de localización promedio en función del tiempo. Para cada algoritmo se realiza el promedio instantáneo de cada una de las diez corridas. Se puede ver para ambos casos que tanto el algoritmo que utiliza Kmeans++ como Spectral Clustering mantienen en promedio una dispersión de las partículas más acotada y menor error en la estimación. También se aprecia que la duración de la misión es distinta para cada algoritmo (la duración presentada es el tiempo promedio que le llevó a cada ejecución completar la misión). Nótese que en este gráfico no se incluyó la desviación estándar para simplificar la visualización. Por esta razón y para complementar las gráficas que se mencionaron, se presenta cada ejecución promedio en forma individual en la figura 5.4.

5.1.7. Resultados de las ejecuciones en el escenario simétrico

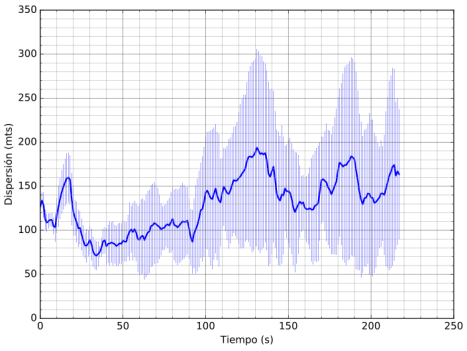
Esta subsección presenta figuras análogas a la subsección anterior pero aplicadas al escenario simétrico. En la figura 5.5a se muestra la evolución promedio



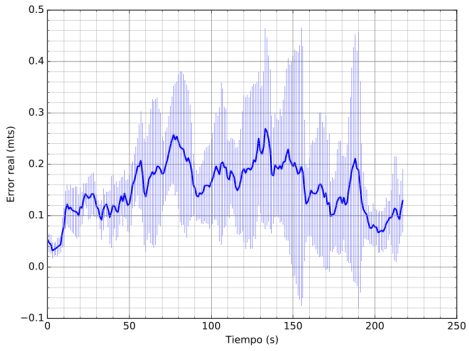
(a) Dispersión localización pasiva.



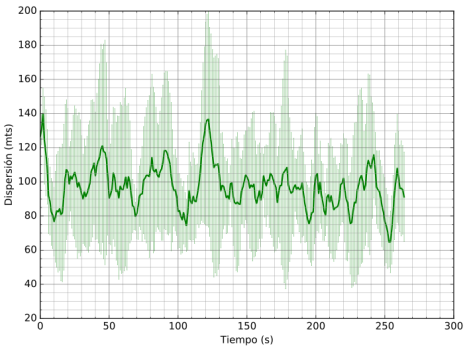
(e) Error real localización pasiva.



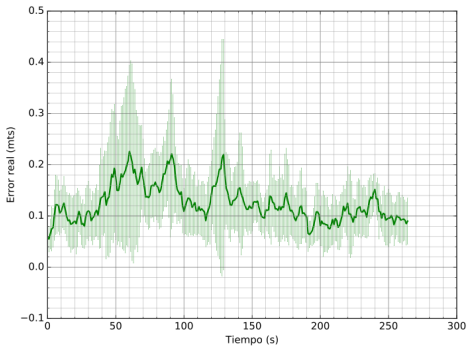
(b) Dispersión DBSCAN.



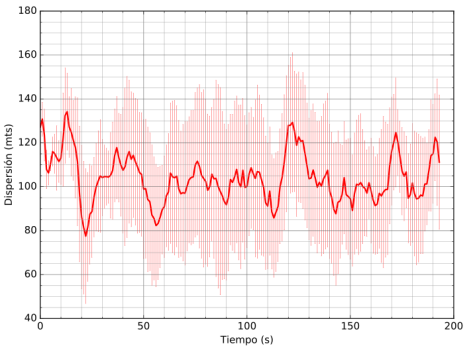
(f) Error real DBSCAN.



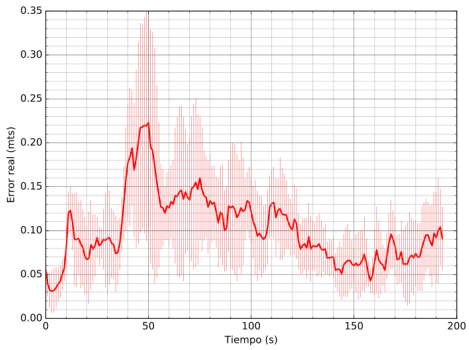
(c) Dispersión Spectral Clustering.



(g) Error real Spectral Clustering.

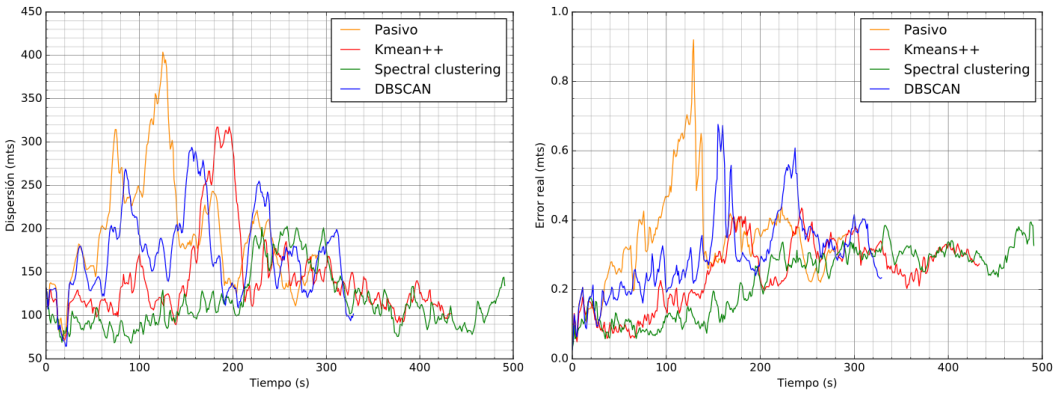


(d) Dispersión Kmeans++



(h) Error real Kmeans++.

Figura 5.4: Resultados experimentales simulados en el escenario INCO individual para cada algoritmo.



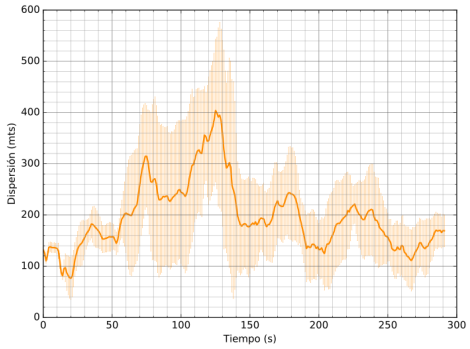
(a) Dispersión **(b)** Error de localización

Figura 5.5: Resultados experimentales simulados en el escenario simétrico.

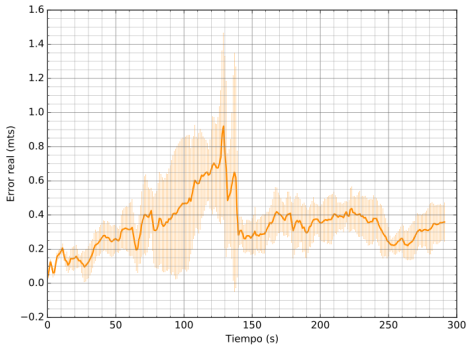
de la dispersión en cada instante de tiempo para cada uno de los algoritmos. En la figura 5.5b se grafica el error de localización promedio en función del tiempo. Se puede ver para ambos casos que tanto el algoritmo que utiliza Kmeans++ como Spectral Clustering mantiene en promedio una dispersión de las partículas más acotada y menor error en la estimación. Al igual que en el escenario INCO también se aprecia que la duración de la misión es distinta para cada algoritmo. Se presenta cada ejecución individual incluyendo su desviación estándar en la figura 5.6.

5.1.8. Resumen de los resultados

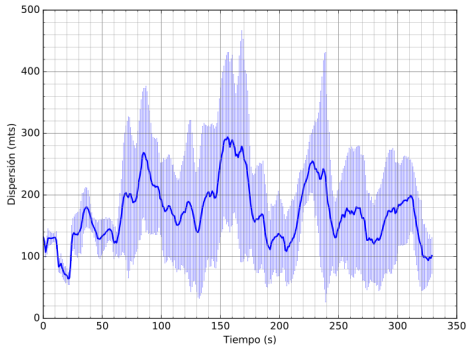
En la tabla 5.1 y tabla 5.2 se presentan en forma resumida los resultados de ejecutar las cuatro estrategias en el escenario INCO y en el escenario simétrico respectivamente. Los valores de la tabla se generan promediando el promedio de cada una de las diez ejecuciones para cada algoritmo. La desviación estándar que se presenta es la del error de los promedios. Las métricas que se incluyen son: dispersión, error en la estimación y tiempo en completar la misión. No se incluye la distancia recorrida ya que no presenta diferencias relevantes entre los algoritmos. En la tabla 5.1 se muestra que la adaptación realizada que utiliza los algoritmos Spectral Clustering y Kmeans++ presenta una mejora cercana al 20 % en la estimación de la posición del robot con respecto a los otros dos agentes. Esto se debe a que el efecto del umbral de disparo y la posibilidad de establecer la cantidad de grupos sobre los algoritmos de agrupamiento permite disparar el algoritmo cada vez que la dispersión aumenta, y eso lleva a que



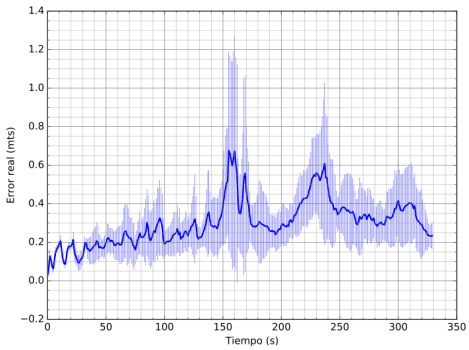
(a) Dispersión localización pasiva.



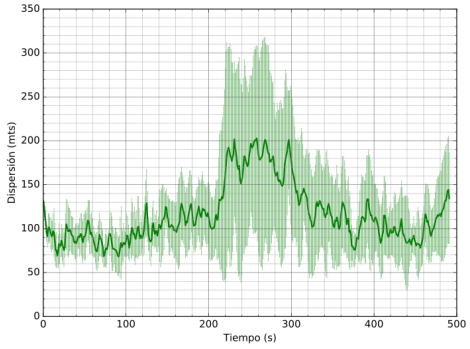
(e) Error real localización pasiva



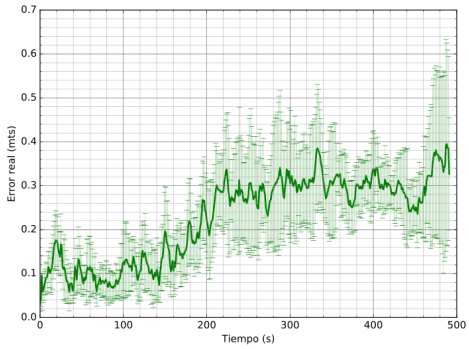
(b) Dispersión DBSCAN.



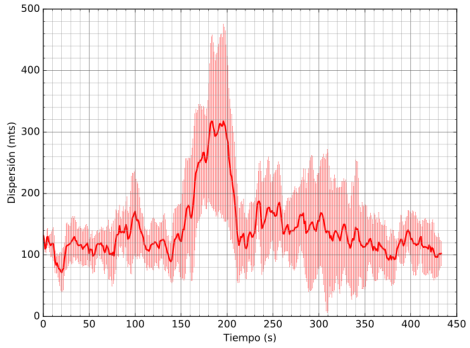
(f) Error real DBSCAN.



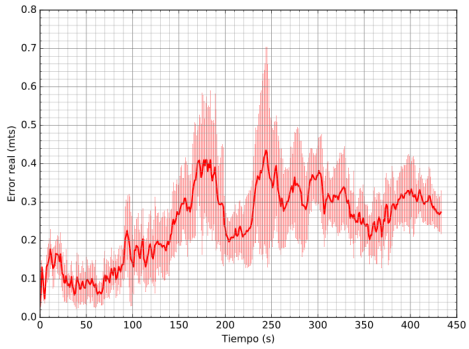
(c) Dispersión Spectral clustering.



(g) Error real Spectral clustering.



(d) Dispersión Kmeans++



(h) Error real Kmeans++.

.

Figura 5.6: Resultados experimentales simulados en el escenario simétrico individual para cada algoritmo.

Tabla 5.1: Resultado promedio de realizar diez ejecuciones con cada agente en el escenario INCO.

Estrategia	Error localización±std (m)	Dispersión±std (m)	Tiempo (s)
Pasivo	0.149±0.020	140.5±13.1	**205**
DBSCAN	0.142±0.017	127.3±22.7	218
Spectral C.	0.112±0.016	103.6±11.7	600
Kmeans++	**0.099±0.007**	**101.8±3.3**	273

sistemáticamente se eliminen a las hipótesis que tienen menos verosimilitud con respecto al sensado. Con respecto a los otros dos casos, se mantuvieron con valores similares entre ellos. Esto se explica porque en el caso de la adaptación que utiliza DBSCAN, dado que las partículas no se alejaron lo suficiente durante la misión del robot, el algoritmo de localización activa se ejecutó muy pocas veces. Con respecto a la dispersión, Kmeans++ y Spectral Clustering mantienen la estimación de la pose con menor incertidumbre. DBSCAN presenta una leve mejoría respecto al algoritmo de localización pasiva. En lo que respecta al tiempo en completar la misión, el algoritmo Pasivo y DBSCAN tuvieron tiempos muy similares (se repite la idea de que DBSCAN se comporto casi como un algoritmo Pasivo por sus pocas ejecuciones). Kmeans++ demoró un 30 % más que el Pasivo, y Spectral Clustering demoró cerca del 300 % más. Esto se debe a la tasa de disparo de la localización activa sumado al tiempo que requiere la ejecución de la rutina (que consiste en detener al robot y rotar hacia la celda candidata).

En el caso del segundo escenario, que se presenta en la tabla 5.2 los resultados son análogos al primer caso. La principal diferencia que se presenta es que Spectral Clustering tuvo un desempeño sensiblemente mejor en cuanto al error en la estimación con respecto a la posición real y la dispersión con respecto a Kmeans++. Sin embargo el tiempo de ejecución fue cercano al 200 % con respecto al Pasivo, mientras que el tiempo de ejecución de Kmeans++ fue del orden del 30 % peor que la estrategia de localización pasiva. En este escenario la estrategia que utiliza DBSCAN se diferenció positivamente de la localización pasiva mejorando levemente su desempeño, sin embargo no logra acercarse a la precisión de la pose obtenida por Spectral Clustering o Kmeans++.

Tabla 5.2: Resultado promedio de realizar diez ejecuciones con cada agente en el escenario simétrico.

Estrategia	Error localización±std (m)	Dispersión±std (m)	Tiempo (s)
Pasivo	0.359± 0.041	202.1±24.9	**338**
DBSCAN	0.30± 0.014	165.6±9.5	391
Spectral C.	**0.232±0.011**	**111.6±5.0**	583
Kmeans++	0.25±0.018	140.2±11.4	455

5.2. Experimentos en el robot real

A continuación se describen los experimentos realizados con el robot real. Como se mencionó anteriormente, estos experimentos tienen un carácter confirmatorio, por eso fueron realizados en un número muy inferior en comparación a los experimentos realizados en los entornos simulados. Los experimentos con el robot físico intentaron replicar lo mejor posible uno de los experimentos simulados, tanto en el tipo de robot y en el tipo de escenario como en la forma de recorrer el entorno. El principal desafío de los experimentos con el robot real fue obtener información de la posición real del robot.

5.2.1. Robot

En este caso los experimentos se hicieron con un robot Turtlebot 2, que se presenta en la figura 5.7, fabricado por Yujin Robot. El robot es de porte medio, y sus dimensionen son una circunferencia de 0,35 metros de radio de base y 0,42 metros de altura, y es el mismo que se utilizó en la propuesta de Li et al. (2016). Es un robot diferencial, equipado por dos ruedas con tracción y dos ruedas libres que se utilizan como apoyo. El robot cuenta con un sensor de profundidad Astra del cual se obtiene una nube de puntos tridimensional del espacio, sin embargo, para los fines del experimento se utilizó como si fuera un sensor láser (la transformación fue hecha utilizando el nodo ROS *deepimage_to_laserscan*). El rango de trabajo del sensor Astra está ubicado entre 0.40 y 8 metros, con una apertura de 60°. El cómputo de robot se ejecutó en una laptop ASUS Zenbook UX305 con un procesador Intel Core M3-6Y30 de 900Mhz y 8GB de memoria RAM.

Figura 5.7: Turtlebot 2.

Figura 5.8: Escenario INCO.

5.2.2. Escenarios

En la figura 5.8 se presenta el piso cero del INCO, donde fueron realizados los experimentos. El INCO consiste en un espacio de oficinas con características muy similares a las de una vivienda. Algunos de los elementos que se encuentran en el escenario son: una mesa con tres sillas, dos bibliotecas, un mueble de seguridad, un sillón de un cuerpo, bicicletas, entre otros objetos. Debido a que el piso cero del INCO es muy amplio, el experimento se realizó solamente en el hall principal y parte del pasillo. Las dimensiones utilizadas del escenario INCO fueron son 7,8 metros de largo por 5,8 metros de ancho del hall principal y adicionalmente se utilizó una parte del pasillo de 5 metros de largo por 1,5 metros de ancho.

5.2.3. Metodología

Para validar los resultados obtenidos en la simulación se trabajó en la versión real del escenario INCO y se utilizó un modelo real del robot muy similar a la versión que se utilizó en la simulación. Al igual que en la simulación se utilizaron tres algoritmos de localización activa y uno de localización pasiva. La metodología utilizada consiste en realizar una ejecución con cada uno de los algoritmos de agrupamiento, obligando al robot a pasar por 15 poses preestablecidas, distanciadas cada una de la siguiente entre uno y cuatro metros. La distancia recorrida aproximada en cada experimento fue de 42 metros. Las poses fueron establecidas en forma manual con el objetivo de que el robot re-

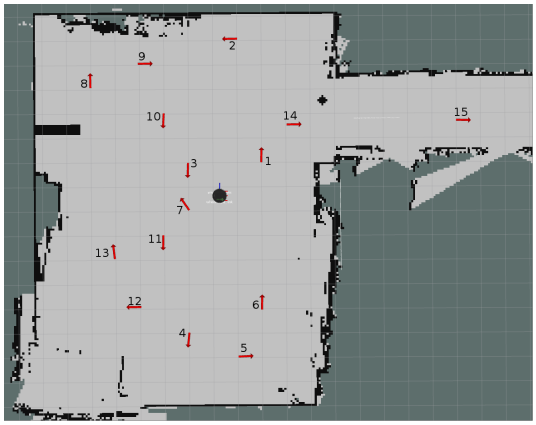


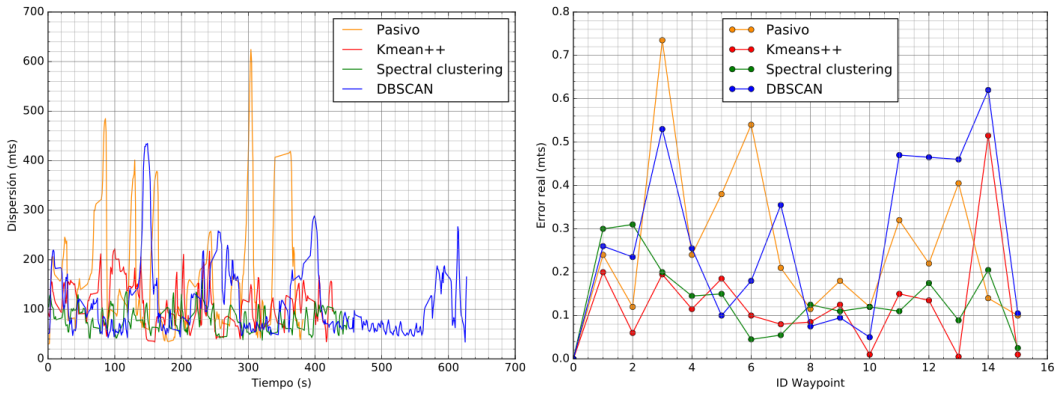
Figura 5.9: Secuencia de poses por las que debe pasar el robot.

corra todo el espacio disponible del hall y pasillo (del mismo modo que en los experimentos simulados), principalmente buscando que se traslade entre los extremos de los ambientes.

La secuencia de poses que debe seguir el robot en el escenario INCO se presenta en la figura 5.9.

5.2.4. Resultados

En la figura 5.10a se muestra la evolución de la dispersión en cada instante de tiempo para cada uno de los algoritmos estudiados. En los cuatro casos las corridas completaron la misión en su totalidad. En la figura 5.10b se gráfica el error real en cada una de las poses preestablecidas. A diferencia de la dispersión (para la que se realizó un muestreo con una frecuencia de 1Hz.), para el error real se tomaron 15 medidas en forma manual para cada experimento debido a la dificultad de conseguir un sistema automático que pudiera medir este valor con precisión. Se puede ver para ambos casos que tanto el algoritmo que utiliza Kmeans++ como el que utiliza Spectral Clustering mantiene una dispersión de las partículas más acotada y menor error en la estimación con respecto al algoritmo que utiliza DBSCAN y el algoritmo Pasivo. También se aprecia que la duración de la misión es distinta para cada algoritmo. Para simplificar la visualización no se incluyó la desviación estandar en ninguna de las dos gráficas de la figura 5.10. Por esta razón y para complementar las gráficas que se mencionaron, en la figura 5.11 se presenta un gráfico individual de cada métrica para cada estrategia utilizada que incluye la desviación estándar.



(a) Dispersión **(b)** Error de localización

Figura 5.10: Resultados experimentales en el escenario real con el robot Turtlebot 2.

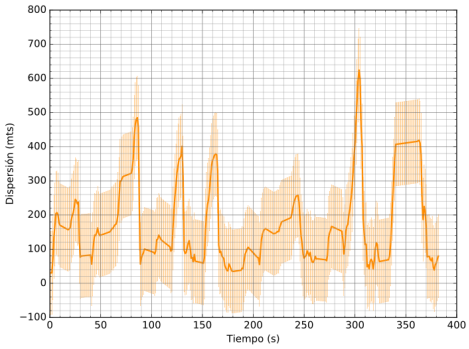
Tabla 5.3: Resultado de realizar una ejecución con cada agente en el escenario INCO.

Estrategia	Error localización±std (m)	Dispersión±std (m)	Tiempo (s)
Pasivo	0.271± 0.137	172.0±121.8	**389**
DBSCAN	0.284±0.160	112.3±69.0	615
Spectral C.	0.144±0.063	**79.1±22.4**	453
Kmeans++	**0.131±0.079**	108.4±41.1	525

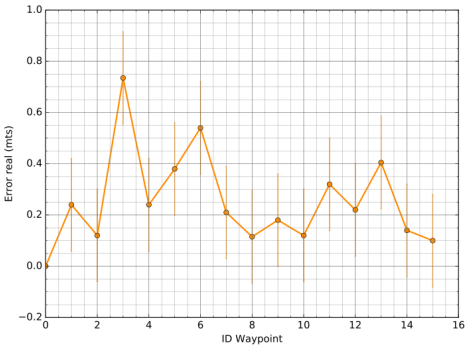
5.2.5. Resumen de los resultados

En la tabla 5.3 se sintetizan los resultados de realizar una corrida con cada una de las cuatro estrategias en el escenario INCO utilizando un robot real. Las métricas que se incluyen son: error de localización, dispersión y tiempo en completar la misión.

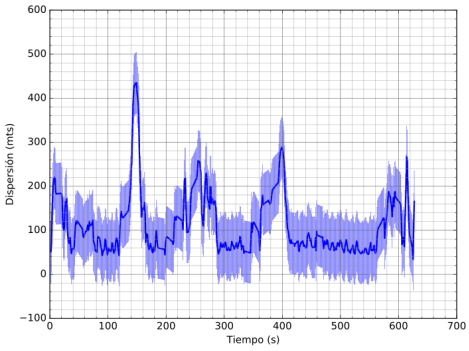
En la tabla 5.3 se puede ver que las adaptaciones que utilizan los algoritmos Spectral Clustering y Kmeans++ presentan una mejora cercana al 50 % en la estimación de la posición del robot con respecto a los otros dos agentes. Con respecto a la dispersión, Kmeans++ y Spectral Clustering mantienen la estimación de la pose con menor incertidumbre, sin embargo la diferencia entre Kmeans++ y DBSCAN no fue muy significativa. Esto podría deberse a alguna particularidad de la propia ejecución, dado que se realizó un solo experimento para cada caso. En lo que respecta al error de localización, DBSCAN y la localización pasiva presentaron valores muy similares. En lo que respecta al tiempo en completar la misión, el algoritmo Pasivo tuvo el mejor desempeño como se



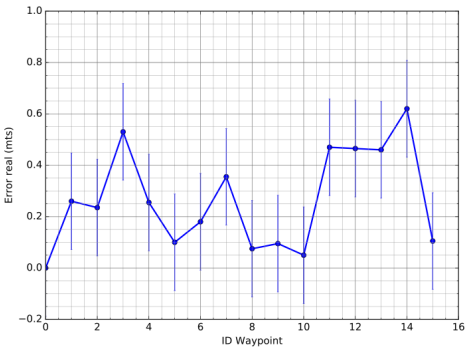
(a) Dispersión localización pasiva.



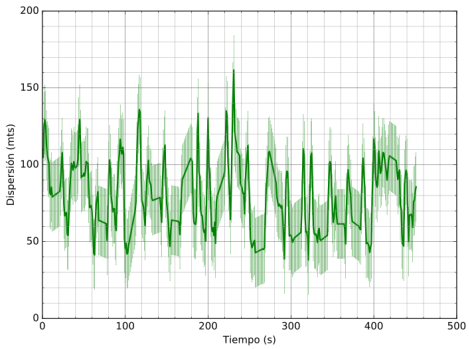
(e) Error real localización pasiva



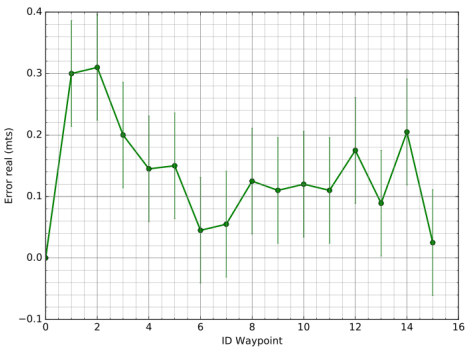
(b) Dispersión DBSCAN.



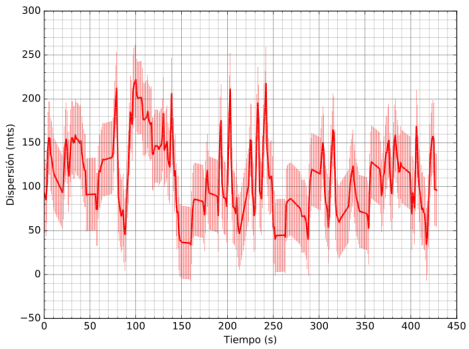
(f) Error real DBSCAN.



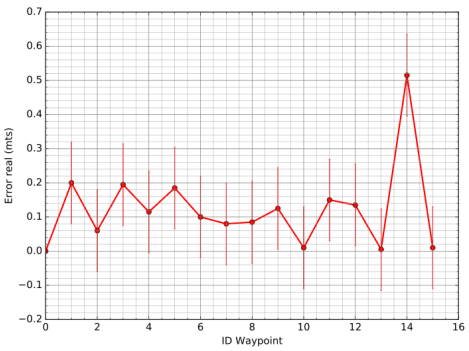
(c) Dispersión Spectral clustering.



(g) Error real Spectral clustering.



(d) Dispersión Kmeans++



(h) Error real Kmeans++.

.

Figura 5.11: Resultados experimentales en el escenario real con el robot Turtlebot 2 discriminado por algoritmo.

esperaba, mientras que Kmeans++ y Spectral Clustering tuvieron tiempos similares. DBSCAN, sin embargo, fue el que más demoró en completar la misión. Finalmente, podemos notar que los resultados obtenidos en los experimentos realizados con el robot real se mantienen en la línea de los resultados obtenidos en simulación.

Capítulo 6

Discusión y Trabajos futuros

Este es el último capítulo del documento y se utiliza para presentar las principales fortalezas, debilidades y conclusiones resultantes del trabajo de investigación realizado durante el desarrollo de esta tesis. Además, se presentan varias lineas de trabajo a futuro que extienden la propuesta de este documento.

6.1. Conclusión

En esta tesis se adaptó la técnica de localización activa global desarrollada por Li et al. (2016) al problema de seguimiento de posición orientado a robots de servicio. Los resultados experimentales validan la hipótesis de que es posible resolver el problema de seguimiento de posición utilizando una técnica de localización activa basada en el solapamiento de hipótesis, y mejorando significativamente los resultados con respecto a la localización pasiva. Las principales características de la propuesta realizada son el uso de dos algoritmos de agrupamiento que se adaptan a las necesidades del problema, el disparo del algoritmo de localización activa solamente cuando la incertidumbre supera un umbral predeterminado, y el cómputo de un conjunto de celdas reducido.

La propuesta realizada en este trabajo disminuiría el error de localización del robot en torno al 20 % y conseguiría que la incertidumbre sobre la pose disminuya un 30 % aproximadamente, lo cual permite al robot realizar tareas de mayor precisión y disminuir las posibilidades de que se pierda. Sin embargo el costo de realizar la misión se vio incrementado en valores cercanos al 30 %, debido al tiempo que agrega la ejecución del algoritmo de localización activa propuesto. Tanto la disminución del error de localización como el incremen-

to del tiempo de navegación pueden ser factores determinantes a la hora de utilizar esta propuesta. Finalmente el rango de celdas analizadas es significativamente menor que la propuesta en la que se basa este trabajo, lo cual también contribuye disminuyendo el costo computacional.

La propuesta de Li et al. (2016) propone una forma novedosa de localizar al robot en forma global utilizando solapamiento de mapas, pero con un algoritmo de agrupamiento que no puede ser aplicado directamente al problema de seguimiento de posición, sin embargo en esta tesis el uso de Kmeans++ y Spectral Clustering permiten adaptar la propuesta en forma conveniente, validando su uso en el contexto de seguimiento de posición. Velez et al. (2011) presentan la idea de desviar levemente al robot para obtener mejores observaciones, lo cual es una idea interesante pero obliga a calcular varios pasos hacia adelante la estimación de la posición lo que repercute en el cómputo. Si bien las ideas se alinean con el trabajo presentado en este documento, el trabajo del autor se diferencia en que la decisión de localizarse se realiza en forma local y solamente cuando es necesario. En el trabajo de Correa and Soto (2010) el algoritmo de localización activa se ejecuta en forma constante, lo que puede llevar a un exceso en el uso del cómputo disponible, a diferencia del trabajo realizado en esta tesis. Además, dado que la cámara está colocada sobre dos motores, se generan inconvenientes en las observaciones. En la propuesta de esta tesis el uso de un sensor láser evita tener que mover constantemente el sensor y con ello imprecisiones en la medición. Sin embargo, el montaje que propone Correa and Soto (2010) permite que el robot pueda seguir la ruta planificada y localizarse activamente a la vez, algo que en este trabajo no se puede realizar, dado que se utiliza el mismo sensor para ambas cosas. Finalmente, en esta tesis los cálculos se realizan localmente, lo cual permite una recuperación rápida en caso de que el robot se encuentre con obstáculos no planificados, a diferencia de Inoue et al. (2016) que ante obstáculos debe realizar todo el cálculo de la ruta nuevamente.

El estudio presentado en este trabajo significa un pequeño avance en la aplicación de técnicas de localización activa al problema de seguimiento de posición. Si bien la aplicabilidad de la adaptación realizada mejoraría los resultados de otras técnicas, la relevancia de los hallazgos dependen de la aplicación en la que se quiera utilizar. Finalmente, como contribución adicional, el software de localización activa generado está disponible en línea para uso de la comunidad, en particular usuarios de ROS.

6.2. Trabajos futuros

Durante el desarrollo de este proyecto de investigación surgen diversos aspectos sobre los cuales se puede extender la propuesta. Las líneas de desarrollo que el autor considera más relevantes se exponen en esta sección.

Un punto a desarrollar es que la cantidad de celdas que se evalúan del mapa sea variable (adaptativo). Esto implica que si en el tamaño actual no se encuentra ninguna celda que aporte la información suficiente para eliminar hipótesis, en el próximo paso se agrande el tamaño de la ventana para contemplar una mayor cantidad de celdas hasta que se encuentre una que permita mejorar la incertidumbre o bien se llegue a un tope.

Otro aspecto que es de interés revisar para el autor es que el peso de los aportes al mapa compuesto de cada agrupamiento esté asociado a la cantidad de partículas que representa. Esto no se hace actualmente y el autor entiende que sería interesante investigar un poco más en este sentido.

Además, sería interesante estudiar hasta qué punto se puede disminuir la cantidad de partículas que se utilizan para disminuir el procesamiento y aprovechar más la técnica de localización activa presentada.

Finalmente, es relevante para el autor contar con un sistema de localización global externo de tiempo real que permita comparar la posición real del robot con la posición estimada por el algoritmo, lo que permitiría mejorar la precisión de los experimentos en el robot físico.

Referencias bibliográficas

Arthur, D. and Vassilvitskii, S. (2007). K-means++: The Advantages of Careful Seeding. In Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA '07, pages 1027–1035, Philadelphia, PA, USA. Society for Industrial and Applied Mathematics.

Bohren, J. and Cousins, S. (2011). The SMACH high-level executive. Robotics & Automation Magazine, IEEE, 17:18 – 20.

Brooks, R. (1986). A robust layered control system for a mobile robot. IEEE Journal on Robotics and Automation, 2(1):14–23.

Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., Niculae, V., Prettenhofer, P., Gramfort, A., Grobler, J., Layton, R., VanderPlas, J., Joly, A., Holt, B., and Varoquaux, G. (2013). API design for machine learning software: experiences from the scikit-learn project. In ECML PKDD Workshop: Languages for Data Mining and Machine Learning, pages 108–122.

Burgard, W., Fox, D., and Thrun, S. (1997). Active mobile robot localization. IJCAI'97 Proceedings of the Fifteenth international joint conference on Artifical intelligence - Volume 2.

Correa, J. and Soto, A. (2010). Active visual perception for mobile robot localization. Journal of Intelligent Robotic Systems.

Dudek, G., Romanik, K., and Whitesides, S. (1998). Localizing a Robot with Minimum Travel. SIAM J. Comput., 27(2):583–604.

Ester, M., Kriegel, H.-P., Sander, J., and Xu, X. (1996). A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In

Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, KDD'96, pages 226–231. AAAI Press.

Fox, D. (2001). KLD-Sampling: Adaptive Particle Filters and Mobile Robot Localization. page 713–720.

Garage, W. (2011). Turtlebot. Website: http://turtlebot.com/, pages 11–25.

Gottipati, S. K., Seo, K., Bhatt, D., Mai, V., Murthy, K., and Paull, L. (2019). Deep Active Localization. IEEE Robotics and Automation Letters, 4(4):4394–4401.

Grisetti, G., Stachniss, C., and Burgard, W. (2007). Improved Techniques for Grid Mapping with Rao-Blackwellized Particle Filters. IEEE Transactions on Robotics, 23(1):34–46.

Inoue, H., Ono, M., Tamaki, S., and Adachi, S. (2016). Active localization for planetary rovers. In 2016 IEEE Aerospace Conference, pages 1–7.

Jung, M. and Song, J.-B. (2017). Efficient autonomous global localization for service robots using dual laser scanners and rotational motion. International Journal of Control, Automation and Systems, 15(2):743–751.

Koenig, N. and Howard, A. (2004). Design and use paradigms for Gazebo, an open-source multi-robot simulator. In 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566), volume 3, pages 2149–2154 vol.3.

Koubaa, A. (2016). Robot Operating System (ROS): The Complete Reference (Volume 1). Springer Publishing Company, Incorporated, 1st edition.

Li, A. Q., Xanthidis, M., O'Kane, J. M., and Rekleitis, I. (2016). Active localization with dynamic obstacles. In 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 1902–1909.

Liu, Z., Chen, W., Wang, Y., and Wang, J. (2012). Localizability estimation for mobile robots based on probabilistic grid map and its applications to localization. In 2012 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI), pages 46–51.

Llofriu, M. and Andrade, F. (2012). Estudio del estado del arte del SLAM e implementación de una plataforma flexible. Master's thesis, Universidad de la República, Facultad de Ingeniería.

Moravec, H. and Elfes, A. (1985). High resolution maps from wide angle sonar. In Proceedings. 1985 IEEE International Conference on Robotics and Automation, volume 2, pages 116–121.

Ng, A. Y., Jordan, M. I., and Weiss, Y. (2001). On Spectral Clustering: Analysis and an Algorithm. In Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic, NIPS'01, pages 849–856, Cambridge, MA, USA. MIT Press.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12:2825–2830.

Quigley, M., Conley, K., Gerkey, B. P., Faust, J., Foote, T., Leibs, J., Wheeler, R., and Ng, A. Y. (2009). ROS: an open-source Robot Operating System. In ICRA Workshop on Open Source Software.

Roy, N. and Thrun, S. (2000). Coastal Navigation with Mobile Robots. MIT Press (2000), pages 1043–1049.

Siciliano, B. and Khatib, O. (2007). Springer Handbook of Robotics. Springer-Verlag, Berlin, Heidelberg.

Thrun, S., Burgard, W., and Fox, D. (2005). Probabilistic robotics. Intelligent robotics and autonomous agents. MIT Press.

Velez, J., Hemann, G., Huang, A., Posner, I., and Roy, N. (2011). Planning to Perceive: Exploiting Mobility for Robuts Object Detection. Proceedings of the Twenty-First International Conference on Automated Planning and Scheduling.

Wang, Y., Chen, W., Wang, J., and Wang, H. (2012). Action Selection Based on Localizability for Active Global Localization of Mobile Robots. IEEE International Conference on Mechatronics and Automation.

Zhang, Z. and Scaramuzza, D. (2019). Beyond Point Clouds: Fisher Information Field for Active Visual Localization. In 2019 International Conference on Robotics and Automation (ICRA), pages 5986–5992.

Glosario

Belief Estimación que mantiene el robot sobre su posición (x, y, θ) en el mapa. También es expresado como $belief(x_t)$ o $bel(x_t)$.

Comportamiento reactivo Comportamiento que vincula la percepción y la acción del robot utilizando un computo mínimo o inexistente.

Curiosity Es el robot explorador con las mayores capacidades hasta la fecha destinado a la exploración de la superficie marciana. Fue lanzado el 26 de noviembre de 2011, habiendo amartizado el 5 de agosto de 2012.

Mapa métrico En un mapa métrico se representa al espacio libre y a los obstáculos mediante medidas espaciales (geométricas). Las representaciones más comunes son celdas de ocupación o geométrico.

Mapa topológico En un mapa topológico se considera al ambiente como un conjunto de lugares y conexiones entre lugares. Ejemplos de lugares pueden ser: cocina, habitación, comedor, mientras que las conexiones pueden ser pasillo, escaleras, *hall*.

Matriz de información de Fisher La matriz de información de Fisher es la cantidad de información que una variable aleatoria mantiene sobre un parámetro oculto del sistema.

Monte Carlo Método no determinista utilizado para aproximar expresiones matemáticas complejas y costosas de evaluar con exactitud.

Odometría Información interna del robot asociada al movimiento propio.

Offline Procesamiento realizado cuando el robot no está en ejecución.

Pila de navegación Paquete de herramientas de ROS que permite a un robot navegar en el entorno a partir de información de odometría y sensorial.

Pioneer El robot Pioneer es una plataforma robótica diferencial y compacta que se utiliza ampliamente en investigación en robótica.

SICK LMS111 Sensor láser de alta calidad muy utilizado tanto a nivel in-

dustrial como en robótica. El sensor trabaja en dos dimensiones con una apertura de 270 grados, con una frecuencia de actualización entre 25 y 50 Hz, y un rango de distancia que va desde los 0,5 a los 20 metros.

Turtlebot2 Kit robótico de bajo costo equipado con software libre. El hardware del robot también es abierto.