Estrategia de Planificación Conmutada para la Navegación de Robots Móviles en Entornos Agrícolas usando SLAM

Fernando A. Auat Cheein, Ricardo Carelli, *Senior-Member, IEEE* •

Resumen

En este trabajo se presenta una arquitectura de planificación de caminos conmutada para la navegación en entornos agrícolas. Se demuestra además la convergencia de la arquitectura de planificación propuesta al implementar un criterio de conmutación basado en la probabilidad de éxito del camino propuesto por cada planificador. Los caminos que son seleccionados por el criterio de conmutación remiten sus referencias a un controlador de seguimiento de caminos que genera los comandos de control del robot móvil utilizado. La información tanto interna del robot como así también del ambiente, es manejada por un algoritmo de SLAM (por sus siglas en inglés de ***Simultaneous Localization and Mapping***). El algoritmo de SLAM estima recursivamente la localización del vehículo dentro del ambiente y los parámetros que describen geométricamente los troncos de los árboles del entorno. Esta información es usada por los planificadores y por el controlador para ejecutar la navegación de una forma estable. Acompañan este trabajo resultados de simulación y experimentación en tiempo real en entornos reales de agricultura.

Palabras Claves: Robots Móviles, Planificación de Camino, SLAM.

Fecha de recepción del original: 30/09/2010 | Fecha de evaluación del original: 18/10/2010

• Artículo enviado el 30 de septiembre de 2010. Este trabajo fue parcialmente financiado por el Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET), Argentina, y por la Facultad de Ingeniería de la Universidad Nacional de San Juan.

Fernando A. Auat Cheein trabaja en el Instituto de Automática de la Universidad Nacional de San Juan, Av. Libertador Gral. José de San Martín 1109 Oeste, 5400 (teléfono: 0264 4213303, fax: 0264 4213672; e-mail: fauat@inaut.unsj.edu.ar)

Ricardo Carelli es el director del Instituto de Automática de la Universidad Nacional de San Juan y Senior-Member del IEEE (e-mail: rcarelli@inaut.unsj.edu.ar).

I. Introducción

Las arquitecturas de navegación autónomas constituyen un problema central en el campo de la robótica móvil. Estas abarcan los problemas de construcción de mapas y localización (SLAM, por sus siglas en inglés de *Simultaneous Localization and Mapping*), los controladores de navegación y la determinación de los objetivos de la navegación [1, 2, 3]. Una arquitectura de navegación exitosa asegurará el cumplimiento de la tarea por parte del robot móvil. Ejemplos de sistemas de navegación autónoma se pueden encontrar en robots móviles para ambientes de agricultura [4], vigilancia y supervisión de entornos cerrados [3], pilotos automáticos de aviones [5], etc. Aunque estas aplicaciones han mostrado ser exitosas sufren de falta de adaptabilidad. Así, si la arquitectura de navegación autónoma fue diseñada para ambientes abiertos (*open space environments*), es muy probable que fracase si esa misma arquitectura pretende ser usada en ambientes restringidos. También, si un robot móvil de interiores es usado en entornos de agricultura o urbanos -siempre que la mecánica del robot permita ese cambio de ambiente- la navegación autónoma podría fracasar pues los algoritmos de detección de paredes o puertas seguramente no serán apropiados para detectar árboles, surcos, rocas, etc. Aún más, terrenos que no sean planos requieren diferentes controladores de movimiento de tal forma que estabilicen la navegación del robot ante depresiones o elevaciones del terreno. Por lo tanto, la adaptación del robot móvil a cada escenario de navegación hace que el concepto de autonomía necesite ser reformulado [6], puesto que requiere la intervención de un supervisor (por ejemplo, un supervisor humano) para actualizar los requerimientos del robot.

Un camino, en robótica, se define como un conjunto de puntos, libre de obstáculos, entre el robot y el destino de la navegación. Un robot móvil autónomo debe ser capaz de moverse siguiendo ese camino de forma tal de alcanzar el objetivo de la navegación [2]. Existen varias técnicas de planificación de caminos propuestas en la literatura científica. Por ejemplo, el algoritmo de *Bug* y sus derivaciones [7]. El algoritmo de *Bug* genera caminos a partir de la posición del robot teniendo en cuenta la disposición del ambiente y algunos criterios previamente definidos para la evasión de obstáculos (por ejemplo, desvío tangencial [7]). Aunque el algoritmo de *Bug* es ampliamente utilizado, éste no ejecuta ningún plan basado en información previa que pudiera tener, siendo por lo tanto, una técnica de generación de caminos puramente reactiva. Por otro lado, los grafos [2] y los mapas basados en grillas [8] son dos alternativas para encontrar caminos óptimos entre el robot y el punto destino. Ambos métodos se basan en la información que manejan del ambiente y en algún criterio de optimización -tales como energía cinética, distancia, etc.- para encontrar el camino óptimo.

Más allá del tipo de algoritmo de planificación usado, es claro que los algoritmos más eficientes son dependientes de la información que manejan, tal como la información concerniente al entorno o al estado interno del robot móvil. Así, la implementación de un algoritmo de SLAM junto a la estrategia de planificación proveerá de la estimación de la información de ambos: el ambiente y el vehículo, a los planificadores usados. El algoritmo de SLAM estima recursivamente la pose -posición y orientación- del robot dentro del ambiente y las características del mismo, minimizando errores [1]. La disponibilidad de un mapa para una planificación eficiente se vuelve un problema crucial, dado que sin el conocimiento total o parcial del entorno, una planificación no-reactiva sería imposible de implementar. Además, el conocimiento del ambiente también permitirá una planificación libre de colisiones.

Para que la navegación de un robot móvil pueda ser realizable, se necesitan controladores de bajo y alto nivel. Los controladores de bajo nivel son directamente implementados en los motores del robot para asegurar una entrada apropiada de corriente y tensión. Los controladores de alto nivel son directamente aplicados a los modelos cinemáticos y dinámicos del robot de acuerdo a la tarea que se pretende realizar [9]. Así, para objetivos de navegación, un robot móvil necesita de controladores no reactivos de alto nivel, tales como controladores de seguimiento de trayectorias o de seguimiento de caminos.

En este trabajo se presenta la implementación de una *estrategia de planificación de caminos paralela conmutada* para la generación de referencias de caminos para la navegación de un robot móvil. Un conjunto finito de planificadores -no necesariamente iguales ni trabajando en el mismo espacio- son conectados en paralelo y un criterio de conmutación conmuta continuamente entre ellos. El criterio de conmutación se basa en la probabilidad de éxito de cada planificador para un instante dado de tiempo. A su vez, un algoritmo de SLAM está implementado en el robot móvil. Este algoritmo de SLAM provee de la información del mapa del ambiente como así también de la pose del robot dentro de ese entorno. Esta información es manejada tanto por el controlador como por los planificadores. Finalmente, el camino generado y navegado por el robot, resulta ser el mejor camino entre todos los posibles caminos generados por los planificadores.

Este trabajo se organiza de la siguiente manera. La sección II presenta la arquitectura general del sistema implementado; la sección III introduce los conceptos de la arquitectura de planificación conmutada propuesta; la sección IV introduce brevemente el algoritmo de SLAM mientras que la sección V muestra los resultados de experimentación, tanto en simulación como en tiempo real en entornos reales, de la arquitectura de navegación propuesta.

II. Arquitectura General del Sistema

La arquitectura de planificación conmutada se encuentra representada en la Fig. 1. La misma funciona como se explica a continuación.

- *N* planificadores de caminos funcionan en paralelo. Estos planificadores no necesariamente usan la misma información del entorno para generar caminos. Así, uno podría usar información de las paredes de un ambiente, mientras que otro usa la información relativa al terreno por el cual navega el robot móvil.
- El algoritmo de SLAM provee de la información relativa a la disposición del ambiente y de la pose del robot dentro del entorno.
- Cada planificador usa la información del ambiente y de la pose del robot provista por el algoritmo de SLAM para generar un camino apropiado entre la posición actual del robot y el objetivo de navegación.
- La ley de conmutación se basa en la información provista por el algoritmo de SLAM como así también por cada uno de los *N* planificadores del sistema. Esta ley de conmutación permite decidir, instantáneamente, cual planificador es el más adecuado para lograr el objetivo de navegación.
- El tiempo de conmutación es superior al tiempo de planificación (así, cada planificador devuelve caminos completos).
- Un controlador de seguimiento de caminos conectado a la salida de la conmutación de planificadores, genera los comandos de movimiento que guian al robot desde su posición actual hasta el objetivo de navegación.
- El algoritmo de SLAM adquiere la información del entorno por el cual navega el robot móvil que, combinada con la señal de control, provee de una estima tanto de la pose del robot como así también de los elementos del entorno.
- La ley de conmutación se actualiza constantemente, por lo tanto las referencias del camino a seguir por el robot móvil no son estáticas.

Como se puede ver, la arquitectura de planificación conmutada es un proceso recursivo entre el proceso de planificación y el algoritmo de SLAM. En las secciones siguientes, se explicará cada uno de los bloques de la Fig. 1.

III. Planificación Conmutada

La planificación conmutada consiste de *N* planificadores, no necesariamente definidos en el mismo espacio de trabajo, que generan caminos apropiados entre el robot y el objetivo de navegación. Los planificadores están conectados en paralelo.

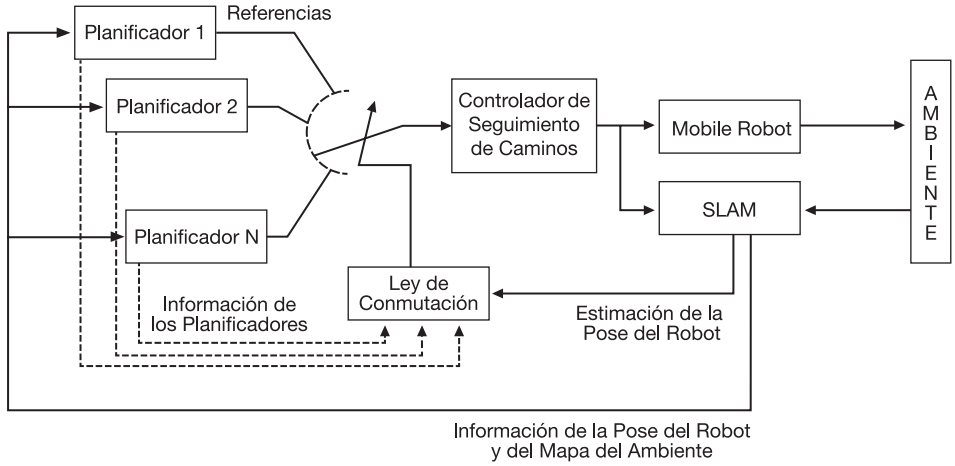


Fig. 1 | **Arquitectura general del sistema**

El algoritmo de SLAM usa la información adquirida del entorno combinada con la ley de control para generar una estima de la pose del robot y de los elementos del entorno. Esa estima es luego utilizada por los planificadores para la generación de posibles caminos.

Una ley de conmutación implementada en el sistema permite conmutar entre los caminos de referencia obtenidos por cada planificador de acuerdo a algún criterio previamente definido. En este trabajo, la ley de conmutación implementada se basa en la *probabilidad de éxito* asociada al camino generado. Así, cada planificador tiene una probabilidad de éxito asociada en un instante de tiempo t. Si, por ejemplo, el planificador $Plan_{j,t}$ tiene la mayor probabilidad de éxito entre los N planificadores en el instante de tiempo t, entonces el camino generado por $Plan_{j,t}$ es pasado al controlador de seguimientos de caminos hasta la siguiente conmutación.

Algunos conceptos sobre planificación usados en este trabajo son presentados a continuación.

A. Conceptos de Planificación de Caminos

El espacio por donde el robot móvil navega se conoce como *espacio de trabajo* y se define como el conjunto de todos los puntos en donde el robot se puede posicionar. El mapa del ambiente, que en este trabajo se denotará como M, es una representación geométrica del entorno por el cual el robot navega. El espacio de trabajo, M_v, es un subconjunto de M ($M_v \subset M$) - M_v se conoce también como *espacio navegable*-. Un obstáculo del ambiente pertenece a la región prohibida

del entorno -se llama región *prohibida* dado que la navegación no es posible en dichos puntos- y se denota como $\overline{M}_v$. $\overline{M}_v$ representa el complemento del espacio de trabajo, así: $M_v \cup \overline{M}_v = M$. Un punto cualquiera en M puede ser tanto un punto libre u ocupado. Un punto está *ocupado* si sus coordenadas pertenecen a un obstáculo del ambiente -por ejemplo, al lugar donde están las paredes-. De manera contraria, el punto se dice que es *libre* o no ocupado. Hay que remarcar que no todos los puntos libres pertenecen a M_v, porque dicho punto podría estar tan próximo a un obstáculo que el robot móvil -debido a sus dimensiones- no podría posicionarse en él.

Un camino c es una curva continua dentro del espacio de trabajo M_v. Así, $c \in C^0$, es decir, el camino pertenece al espacio de las curvas continuas [7]. Dado que un robot móvil tiene un modelo cinemático asociado, el camino debe ser además, cinemáticamente compatible con el modelo del robot [10]. De acuerdo a [7], un camino para robots no-holonómicos del tipo uniciclo debería ser $c \in C^0$ y para robots tipo car-like debería ser $c \in C^1$ -camino continuo con primera derivada continua dentro del espacio de trabajo-. El camino, en su totalidad, debe estar contenido dentro del espacio de trabajo, $c \subset M_v$, caso contrario, el camino visitaría zonas prohibidas o estaría fuera del rango de trabajo.

El *punto destino*, también llamado *objetivo de navegación*, es un punto del espacio de trabajo que se asume como el destino del proceso de navegación. Usualmente, el punto de inicio del camino es la posición actual del robot móvil y el punto final del camino es el objetivo de navegación.

Si el robot móvil es capaz de alcanzar un punto destino P_d siguiendo el camino c, entonces tanto P_d como c se llaman *puntos alcanzables* [8].

Sea P_d cualquier punto destino de M_v y P_r la posición actual del robot móvil también dentro de M_v, sea además un camino $c \subset M_v$ entre P_r y P_d. Sea t_0 el instante de tiempo del robot en P_r. Luego, $length_c(P_r, P_d) \to 0$ cuando $t_0 \to t_f$, donde $length_c$ es la longitud del camino c. Así, el robot debería ser capaz de alcanzar P_d en una cantidad de tiempo finita. Aunque esta restricción de tiempo no es obligatoria para la planificación de caminos, asegura que el robot eventualmente llegará hasta un entorno de P_d.

En este trabajo, la ley de conmutación empleada se basa en la probabilidad instantánea de éxito asociada a un camino c ($P_t(c)$). La probabilidad de éxito da una medida del éxito del robot en alcanzar el punto destino P_d dada la posición inicial del vehículo. Así, si $c \subset M_v$, entonces $P_t(c) \to 1$ cuando $length_c(P_r, P_d) \to 0$, o de manera similar, $P_t(c) \to 1$ cuando $t_0 \to t_f$. Estos enunciados serán demostrados en la siguiente sección.

En este trabajo se asume que el punto destino es un punto alcanzable del entorno como así también que los mapas no poseen componentes estocásticas.

B. Probabilidad de Éxito

Como fue establecido en la sección anterior, la información manejada por el *i*-ésimo planificador no necesariamente debe ser igual a la información manejada por el *j*-ésimo planificador. Aún más, el *i*-ésimo y el *j*-ésimo planificador podrían compartir información común al ambiente por el cual navega el robot móvil.

Por ejemplo, sea M_1 la percepción del ambiente M por el primer planificador in la Fig. 1. Entonces, M_{v1} es el espacio navegable determinado a partir de M_1 y $\overline{M}_{v1}$ es el espacio no navegable para el primer planificador. Sea ahora M_2 la percepción del ambiente M por el Segundo planificador, y sean también M_{v2} y $\overline{M}_{v2}$ el conjunto de puntos navegables y no navegables, respectivamente, asociados a M_2. Un camino generado por el primer planificador de acuerdo a la información provista por M_{v1} sería de la forma:

$$c_1 \subset M_{v1}. \qquad (1)$$

De igual manera, un camino generado por el planificador 2 sería de la forma mostrada en el Ec. (2).

$$c_2 \subset M_{v2}. \qquad (2)$$

Así, de acuerdo a las Ecs. (1) y (2), un camino común a ambos planificadores sería de la forma:

$$c_{1,2} \subset (M_{v1}, M_{v2}). \qquad (3)$$

Tal como se indicó anteriormente, si el punto destino es alcanzable, entonces $P_t(c_1 \mid M_{v1}) \to 1$ cuando $t_{0,1} \to t_{f,1}$ (donde $t_{0,1}$ es el instante inicial del robot de acuerdo al planificador 1). Similarmente, se aplica para $P_t(c_2 \mid M_{v2})$ (es la probabilidad de alcanzar el punto destino a partir del camino c_2 generado usando la información del ambiente provista por M_{v2}). Así, se define como *probabilidad de éxito* de un camino dado c_k, a la medida de realizabilidad de ese camino dado el espacio navegable común a todos los planificadores, tal como se muestra en la ecuación (4). Esto último significa que, por ejemplo, si una región del ambiente es prohibida para un planificador dado, esa misma región -o, en caso de que fuera posible, una proyección de la misma- es también prohibida para el resto de los planificadores.

$$P(c_k \mid M_v) = \eta P(c_k \cap (M_{v1} \cap M_{v2} \cap ... \cap M_{vN})), \quad con\ 1 \le k \le N. \qquad (4)$$

En la Ec. (4), η es un valor de normalización, por la regla de Bayes. Si se asume que:

$$\begin{cases} P(c_k \mid M_v) \equiv 1 \text{ si } c_k \cap \overline{M}_{vj} \equiv \phi \; \forall j = 1...N \\ P(c_k \mid M_v) \equiv 0 \text{ si } \exists \overline{M}_{vj} : c_k \cap \overline{M}_{vj} \neq \phi, \text{ para algún } j = 1...N \end{cases} \quad (5)$$

Entonces la probabilidad de realizabilidad del camino c_k dado M_v -el espacio navegable común asociado a todos los planificadores- es igual a 1 si no cruza una región prohibida; es igual a cero para cualquier otro caso.

Así, de acuerdo a la Ec. (5), la probabilidad de éxito de un camino c_k, con $1 \leq k \leq N$, depende de que el camino pertenezca, en su totalidad, al espacio de trabajo del entorno por el cual se navega (M_v) y así, c_k no estaría atravesando ninguna región prohibida.

Aunque la Ec. (5) se aplica para el camino total, también puede ser aplicada para obtener la probabilidad de éxito de un camino a partir del marco de referencia local al robot móvil. A partir de que el marco de referencia local al robot móvil se considera asociado a su pose [2], sea $B_R(P_r)$ una vecindad de radio R centrada en la posición del robot P_r. De acuerdo a esto, la probabilidad de éxito debería ser calculada dentro del espacio navegable contenido en $B_R(P_r)$. Así

$$\begin{aligned} P(c_k \mid B_R(P_r)) &= P(c_k \mid (B_R(P_r) \cap M_v) \quad (6) \\ &= P(c_k \mid (B_R(P_r) \cap M_{v1} \cap M_{v2} \cap ... \cap M_{vN})). \end{aligned}$$

En la Ec. (6), en vez de calcular la probabilidad de realizabilidad del camino c_k con respecto a todo el ambiente navegable (M_v), se calcula dentro de una vecindad $B_R(P_r)$ en la posición del robot móvil. Así, incrementando sucesivamente R en la Ec. (6), se tiene que $P(c_k \mid B_R(P_r)) \rightarrow P(c_k \mid M_v)$.

Finalmente, de acuerdo a la Ec. (4), es posible ver que si $M_{v1} \cap M_{v2} \cap ... \cap M_{vN} = \phi$, entonces no existen punto comunes de navegación entre todos los planificadores, luego $P(c_k \mid M_v) \equiv 0$, lo cual implica que la navegación no es posible dados los actuales planificadores.

C. Ley de Conmutación

En esta sección se presentará el criterio de conmutación utilizado teniendo en cuenta el concepto de probabilidad de éxito definido en la sección anterior. Una ley de navegación apropiada debe asegurar el cumplimiento del objetivo de navegación.

Sea $length_{c,k}(P_r, P_d)$ la longitud del camino obtenido por el planificador *k*-ésimo, P_d es el punto destino dentro del espacio navegable M_v y P_r es la posición actual del robot. Supóngase también que: $length_{c,1}(P_r, P_d) \neq length_{c,2}(P_r, P_d) \neq ... \neq length_{c,N}(P_r, P_d)$, es

decir, todos los planificadores dan caminos diferentes. La Ec. (4) establece que la probabilidad de éxito de un camino no varía con el tiempo, lo cual significa que, una vez encontrado un camino, el mismo no se actualiza. Tal como se estableció en la sección anterior, la ley de conmutación se basa en la probabilidad de éxito asociada a un instante dado *t*. De acuerdo a la Fig. 1, esto último significa que el sistema conmuta entre los distintos planificadores basándose en el valor instantáneo de probabilidad de éxito asociada a cada planificador. La Ec. (7) muestra este criterio de conmutación.

$$\begin{aligned}\alpha(t) = \max(&P(c_1 \mid B_{R1}(P_r))f_1(t), P(c_2 \mid B_{R2}(P_r))f_2(t),... \\ &..., P(c_N \mid B_{RN}(P_r))f_N(t)) \\ &\text{para todo } R \in \Re \text{ y } R_1 = ... = R_N\end{aligned} \tag{7}$$

En la Ec. (7), α es el criterio de conmutación. Representa la máxima probabilidad de éxito asociada a los planificadores del sistema. De acuerdo a la Fig. 1, si el *i*-ésimo planificador tiene la máxima probabilidad de éxito entre todos los planificadores en el instante de tiempo *t*, entonces ese planificador entrega sus referencias de camino al controlador en el instante *t*. El hecho de que $R_1 = ... = R_N$ en la Ec. (7), implica que todos los planificadores actúan dentro de un espacio de las mismas dimensiones. Además, en la Ec. (7), la probabilidad de éxito de cada planificador se encuentra afectada por una función $f_k(t)$, con $k = 1...N$. Se define ahora la *probabilidad de éxito instantánea* como:

$$\begin{aligned}&P_t(c_k \mid M_v) = P(c_k \mid B_{Rk}) f_k(t) = \\ &= P_t(c_k \mid B_{Rk}) e^{-length_{c,k,t}(P_r,P_d)}, 1 \le k \le N\end{aligned} \tag{8}$$

Así, de acuerdo a la Ec. (8), se define la función f_k como: $f_k = e^{-length_{c,k,t}(P_r,P_d)}$. De esta manera, mientras la longitud del camino entre la posición del robot y el punto destino decrezca, entonces la probabilidad de éxito instantánea aumenta. De la Ec. (8) es posible ver también que: $0 \le P_t(c_k \mid B_{Rk}) e^{-length_{c,k,t}(P_r,P_d)} \le 1$.

IV. Localización y Reconstrucción Simultánea de Entornos

Los problemas de mapeo y localización fueron estudiados inicialmente de manera independiente. En [3] se puede apreciar un estado del arte más completo sobre las primeras investigaciones en mapeo y localización.

En la Fig. 2 se puede observar la forma en que los problemas de *localización* se relacionan con el problema de *mapeo* y con el algoritmo de *SLAM*. El algoritmo de SLAM surge para fusionar la información obtenida por la técnica usada para

la localización con la información derivada del mapa. Para completar el concepto de autonomía de un robot móvil, es preciso agregarle al algoritmo de SLAM una estrategia de navegación o exploración, de forma tal que pueda obtener un mapa consistente del entorno y una localización eficaz de sí mismo y de las características del ambiente a medida que navega.

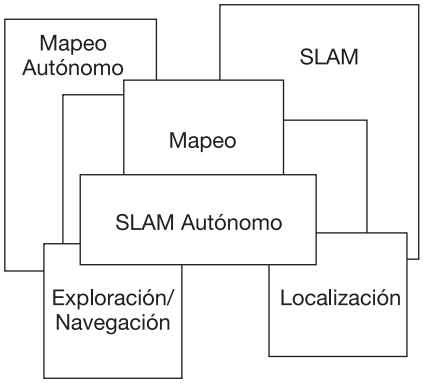


Fig. 2 | **Requerimientos de un SLAM autónomo.**

El algoritmo de SLAM implementado en este trabajo se basa en el filtro Extendido de Kalman (EKF, por sus siglas en inglés de *Extended Kalman Filter*) en su forma secuencial [1] presentado en la ecuación (9).

$$\begin{aligned}
\hat{x}_t^- &= f(\hat{x}_t, u_t) \\
P_t^- &= A_t P_{t-1} A_t^T + W_t Q_{t-1} W_t^T \\
K_t &= P_t^- H_t^T (H_t P_t^- H_t^T + R_t)^{-1} \\
\hat{x}_t &= \hat{x}_t^- + K_t (z_t - h(\hat{x}_t^-)) \\
P_t &= (I - K_t H_t) P_t^-
\end{aligned} \qquad (9)$$

En (9), $\hat{x}$ es el vector de estados del SLAM; u es el vector de control impartido al robot móvil; P^- es la matriz de covarianza predicha asociada con el vector de estados del SLAM y P es su matriz de covarianza corregida; f es el modelo cinemático del vector de estados y A su matriz jacobiana; Q es el ruido asociado al proceso mientras que W su matriz jacobiana; K es la ganancia de Kalman y H es la matriz jacobiana del modelo de la observación; R es el ruido asociado a la característica observada. El factor $(z - h(\hat{x}))$ se conoce como vector innovación, y es la diferencia entre la característica observada (z) y el modelo predicho de la misma ($h(\hat{x})$). Las primeras dos expresiones en (9) se conocen como *etapa de predicción*, mientras que las últimas tres constituyen la *etapa de corrección*.

Las características extraídas del ambiente corresponden a características modeladas como puntos. El vector de estados del sistema se compone de la estima de la pose del robot móvil y de las características del ambiente. Las ecuaciones (10) y (11) muestran la estructura del vector de estados del sistema y su correspondiente matriz de covarianza. Todos los elementos del vector de estados del sistema están referenciados a un sistema de coordenadas global.

$$\hat{x}_t = \begin{bmatrix} \hat{x}_{v,t} \\ \hat{x}_{m,t} \end{bmatrix} \qquad (10)$$

$$P_t = \begin{bmatrix} P_{vv,t} & P_{vm,t} \\ P_{mv,t} & P_{mm,t} \end{bmatrix} \qquad (11)$$

En (10), $\hat{x}_t$ es la estima del vector de estados del sistema en el instante *t*; $\hat{x}_{v,t} = \begin{bmatrix} \hat{x}_{vx,t} & \hat{x}_{vy,t} & \hat{x}_{v\theta,t} \end{bmatrix}^T$ es la estima de la pose del vehículo; $\hat{x}_{m,t}$ representa el mapa del ambiente y está compuesto por las coordenadas Cartesianas que definen una característica del ambiente. El orden en el cual las características aparecen en $\hat{x}_{m,t}$ depende del momento en que fueron detectadas. P_t es la matriz de covarianza asociada con el vector de estados del SLAM; $P_{vv,t}$ es la covarianza de la pose del vehículo y $P_{mm,t}$ es la covarianza asociada a las características del entorno. El resto de los elementos de P_t son las matrices de correlación cruzada.

Las técnicas de inicialización de la matriz de covarianza y la definición del EKF se pueden encontrar en [3]. El algoritmo de EKF secuencial fue implementado con el objetivo de reducir los tiempos de procesamiento durante la estimación [1].

A. Robot Móvil

El robot móvil usado en este trabajo es un robot no-holonómico tipo uniciclo, Pioneer 3AT construido por *ActivMedia*. La Fig. 3 muestra una foto del mismo con un láser *SICK* incorporado. El láser es un sensor de rango que adquiere 181 mediciones entre 0 y 180 grados, con un alcance máximo de 30 metros.

Fig. 3 | **Fotografía del robot Pioneer 3AT.**

El modelo cinemático del robot móvil se presenta en la ecuación (12), donde V_t es el comando de velocidad lineal; W_t es el comando de velocidad angular; Δt es tiempo de muestreo del sistema; x_t , y_t y θ_t definen la pose y la orientación del robot en el instante *t*; Φ_t es el ruido Gaussiano asociado al proceso.

$$\begin{pmatrix} x_t \\ y_t \\ \theta_t \end{pmatrix} = \begin{pmatrix} x_{t-1} \\ y_{t-1} \\ \theta_{t-1} \end{pmatrix} + \Delta t \begin{pmatrix} V_t \cos(\theta_{t-1}) \\ V_t \sin(\theta_{t-1}) \\ W_t \end{pmatrix} + \Phi_t \qquad (12)$$

B. Extracción de Características

Las características usadas por el algoritmo de SLAM presentado en este trabajo corresponden a características basadas en puntos (en inglés: *point-based features*). Las características basadas en puntos son usadas por la literatura científica [11] para representar árboles del entorno. Así, el centro del tronco detectado tiene asociada una característica puntual. La Fig. 4 muestra una representación de dichas características y la Ec. (13) muestra el modelo matemático geométrico de ella. El sensor usado para la extracción de las características fue el sensor de rango láser mostrado en la Fig. 3. Los puntos extraídos por el láser fueron agrupados mediante un algoritmo de *clustering* recursivo [10].

$$\begin{aligned} z_{tronco}(k) &= h_i[x_v(k), p_i(k), w(k)] = \begin{bmatrix} z_R \\ z_\beta \end{bmatrix} \\ &= \begin{bmatrix} \sqrt{\left(x_{v,x}(k) - x_{tronco}\right)^2 + \left(x_{v,y}(k) - y_{tronco}\right)^2} \\ \operatorname{atan}\left(\dfrac{x_{v,y}(k) - y_{tronco}}{x_{v,x} - x_{tronco}}\right) - x_{v,\Psi}(k) \end{bmatrix} + \begin{bmatrix} w_R \\ w_\beta \end{bmatrix} \end{aligned} \qquad (13)$$

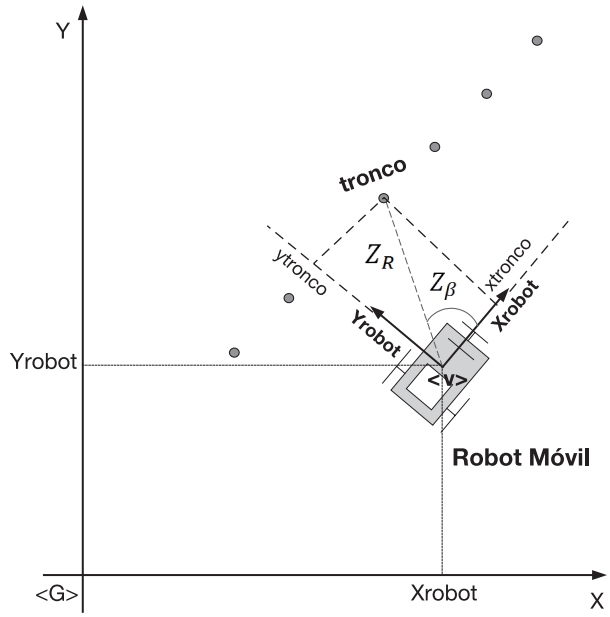


Fig. 4 | **Extracción de características puntuales del ambiente.**

C. Controlador de Seguimiento de Caminos

La ley de control de seguimiento de caminos usada en este trabajo fue propuesta por [12]. Esta ley de control es asintóticamente estable. Las entradas al vehículo son la referencia de la pose $[x_r \quad y_r \quad \theta_r]^T$ y las velocidades de referencia $[V_r \quad W_r]^T$.

El error de pose se define a continuación:

$$\begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} = \begin{bmatrix} \cos\theta_v & \sin\theta_v & 0 \\ -\sin\theta_v & \cos\theta_v & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_r - x_v \\ y_r - y_v \\ \theta_r - \theta_v \end{bmatrix}$$

donde $[x_v \quad y_v \quad \theta_v]^T$ es la pose actual de la silla de ruedas robotizada dentro del entorno. La ley de control es:

$$\begin{bmatrix} V \\ W \end{bmatrix} = \begin{bmatrix} V_r \cos\theta_e + K_x x_e \\ W_r + V_r (K_y y_e + K_\theta \sin\theta_e) \end{bmatrix}$$

donde K_x, K_y and K_θ son constantes positivas.

V. Resultados Experimentales

La estructura de planificación conmutada junto con el algoritmo de SLAM presentados en este trabajo fueron implementados en el robot móvil tipo uniciclo, no-holonómico, Pioneer 3AT mostrado en la Fig. 3. Este robot tiene a su vez un láser tipo SICK montado sobre su estructura como así también un láser tipo HOKUYO orientado hacia el suelo. El láser tipo HOKUYO tiene un rango máximo de 4 metros.

El entorno agrícola donde se realizaron las experimentaciones es una plantación de olivos perteneciente a la estación experimental INTA-San Juan.

Dos planificadores fueron usados en este trabajo. Un planificador de caminos basados en puntos de frontera sucesivos y un planificador de caminos basado en el algoritmo de Dijkstra. El primer planificador genera caminos a partir de la disposición de los olivos (usa el laser SICK), mientras que el segundo planifica en función de la información del terreno (usa el láser HOKUYO). En las siguientes secciones se detallarán ambos planificadores.

A. Planificación basada en puntos de frontera sucesivos

La planificación de caminos basada en puntos de frontera sucesivos fue previamente presentada por [13,10]. Brevemente, este método consiste en encontrar espacios vacíos en los límites del rango del sensor de distancia usado. Cada espacio vacío es considerado como un nodo -ver Fig. 4a-. Variando el rango del sensor, se pueden obtener diferentes nodos a diferentes distancias, tal como se muestra en la Fig. 4b. Finalmente, el conjunto de puntos son unidos mediante segmentos, constituyendo así un camino compatible con el robot móvil. La planificación de caminos por puntos de frontera sucesivos funciona de la siguiente manera:

- Sea δ la distancia entre la posición actual del robot y el punto destino.
- Sea R_g el rango de medición del sensor.
- El primer nodo se busca en una distancia definida como: $Dist_{nodo\_1} = \min\{\delta, R_g\}$, entre todas las mediciones que no alcanzan un obstáculo dentro de la distancia $Dist_{nodo\_1}$. Por ejemplo, si el rango del sensor se fija en 5 metros y el punto destino está a 7 metros del robot, entonces el primer nodo debería encontrar entre todas las mediciones que pertenezcan al límite de 5 metros, es decir, entre aquellas que no alcancen algún elemento del ambiente.
- Considérese ahora que el rango del sensor se fija en $Dist_{nodo\_2} = Dist_{nodo\_1} - \Delta d < Dist_{nodo\_1}$. El próximo nodo del camino se obtendrá entonces por un ventaneo -en ángulo- entre todos los puntos frontera pertenecientes al nuevo rango de $Dist_{nodo\_1} - \Delta d$ -ver Fig. 4b-. El

ventaneo en ángulos es necesario para determinar el nodo más próximo al nodo previamente encontrado. El valor de Δd es definido por diseño. En este trabajo se adoptó $\Delta d = 1$ metro.

- El siguiente nodo se busca en un rango: $R_g = Dist_{node\_1} - 2\Delta d$, y así sucesivamente hasta que el rango del sensor es menor que las dimensiones propias del robot.

Una descripción más completa del método de generación de caminos por puntos de frontera sucesivos se puede encontrar en [10].

En este trabajo, la planificación de caminos basada en puntos de frontera sucesivos es usada para la generación de caminos entre los olivares del ambiente de experimentación.

B. Planificación basada en el algoritmo de Dijkstra

El algoritmo de Dijkstra es ampliamente usado por la comunidad científica para encontrar caminos óptimos dado un conjunto previamente definido de nodos y costos [7]. En este trabajo, se usa el algoritmo de Dijkstra para encontrar un camino óptimo entre la posición actual del robot móvil y el punto destino de navegación, usando la información relativa al terreno por el cual navega el robot. Así, mediante la disposición del láser HOKUYO inclinado, el robot adquiere información del terreno por el cual se mueve. Esa información es filtrada y mediante algoritmos de *clustering* recursivo, se grilla el mapa geométrico del suelo. La Fig. 5 muestra una fotografía del entorno de experimentación (estación experimental INTA-San Juan). En esta fotografía, se puede ver el suelo segmentado. Cada celda de la grilla tiene un costo asociado, dado por la morfología de dicha celda. Así, si una celda contiene una roca de gran tamaño (que podría comprometer la estabilidad de la navegación) o una depresión profunda, dicha celda tendrá asociado un elevado costo de navegación. Caso contrario, su costo será bajo.

Dado que el espacio de trabajo de este planificador es más reducido que el anterior (solamente puede planificar en función de la información local del terreno), el espacio navegable del planificador basado en el algoritmo de Dijkstra es significativamente menor que el espacio navegable del planificador basado en puntos de frontera sucesivos. A medida que el robot avanza por el entorno, la información del mapa del terreno se actualiza. En este trabajo, el número de celdas de la grilla en la Fig. 5 se considera como fijo. Es decir, el algoritmo de Dijkstra siempre planificará con 50 nodos. El nodo inicial corresponde a la posición actual del robot y los 49 restantes se muestran en la Fig. 5. El nodo de salida es aquél que más cerca se encuentre del punto destino de navegación.

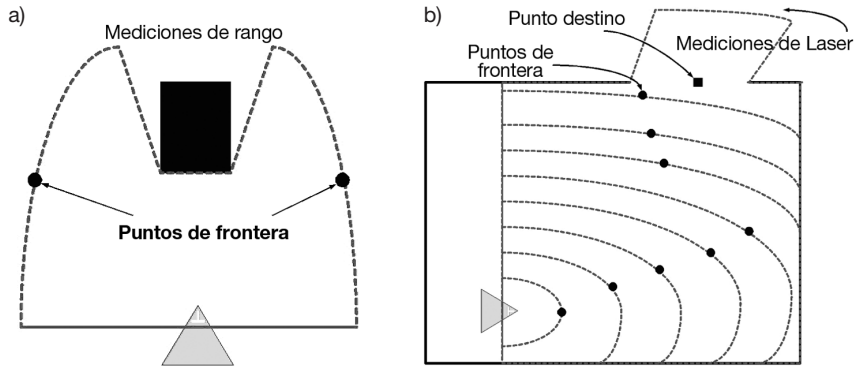


Fig. 4 | **Método de generación de puntos de frontera sucesivos.**

Fig. 5 | **Foto del cultivo de olivos de la estación experimental INTA-San Juan.**

El mapa grillado representa la información del terreno manejada por el robot móvil. Las celdas sombreadas en negro corresponden a celdas con un elevado costo (contienen arbustos, rocas o depresiones profundas).

C. Planificación Conmutada: resultados de simulación

La Fig. 6 muestra algunos resultados de simulación de la arquitectura de planificación conmutada propuesta en este trabajo. El punto destino está representado por el círculo sólido magenta, los bloques amarillos representan obstáculos del ambiente y el rectángulo negro representa el mapa grillado del terreno del entorno. Cada celda del mapa tiene un costo asignado, representativo de una superficie

simulada. En la Fig. 6a, los caminos generados por el método de *puntos de fronteras sucesivos* están representados por líneas sólidas magentas; mientras que los caminos generados por el algoritmo de Dijkstra basado en la información de terreno están representados por líneas sólidas negras. Para fines de visibilidad, sólo se representan algunos caminos generados durante la navegación. La Fig. 6b muestra como las referencias del camino se alternan según sea la probabilidad de éxito de cada planificador. Así, los tramos magenta del camino corresponden al primer planificador mientras que los tramos negros, al segundo. La Fig. 6c muestra un entorno más complejo, donde se puede ver con más frecuencia la conmutación de planificadores. Claramente, el camino alcanza el objetivo de navegación. Aún más, por el criterio de optimización de Bellman [14], se puede demostrar que el camino encontrado es el mejor de cualquiera de los dos métodos de planificación de caminos utilizados.

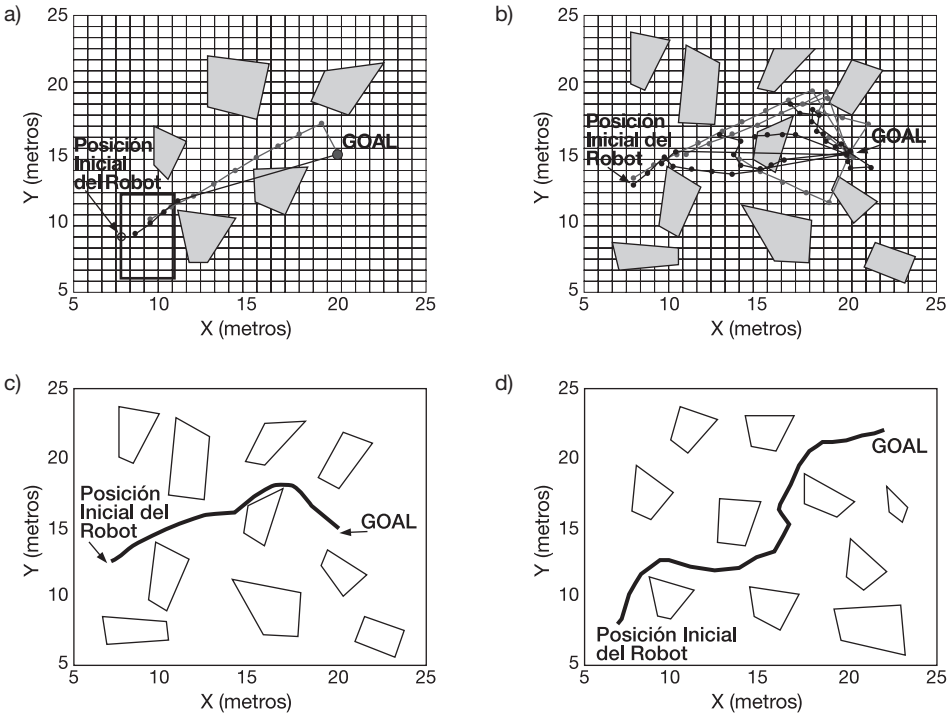


Fig. 6 | **Ejemplos en simulación de la arquitectura de planificación conmutada presentada en este trabajo**

a) Muestra un entorno con 5 obstáculos (en amarillo) visibles al primer planificador (basado en puntos de frontera sucesivos); el terreno del entorno se encuentra grillado y el robot sólo observa la región del suelo comprendida dentro del rectángulo sólido negro. b) Muestra un ejemplo de planificación conmutada donde los caminos en magenta son generados por el primer planificador y los caminos en negro por el segundo. c) y d) Muestran el camino real seguido por el robot móvil. Los segmentos en magenta corresponden a la parte del camino generada por el primer planificador; mientras que los segmentos en negro corresponden a la parte del camino generada por el segundo planificador.

D. Planificación Conmutada: resultados de experimentación

Como se explicó anteriormente, las experiencias fueron realizadas en la Estación Experimental INTA-San Juan. El rango máximo del sensor SICK (asociado al planificador basado en puntos de frontera sucesivos) fue fijado en 30 metros, mientras que el rango del láser HOKUYO (asociado al segundo planificador) fue fijado en 4 metros. El tamaño de la grilla del terreno (ver Fig. 5) asociada al segundo planificador fue de 2.1×2.1 m^2. La Fig. 7 muestra los resultados experimentales en tiempo real de la arquitectura de planificación propuesta junto con los resultados del algoritmo de SLAM aplicado al entorno agrícola. La Fig. 7a muestra el camino recorrido por el robot móvil dentro del olivar. Los tramos magenta corresponden a los caminos trazados por el primer planificador (basado en puntos de frontera sucesivos) mientras que los tramos en negro sólido corresponden al segundo planificador. Como se puede ver, los obstáculos del entorno (los árboles, depresiones del camino o rocas) son evitados mediante la información provista por los sensores asociados a sus respectivos planificadores. Así, el robot navega por la parte más plana del terreno o la región más segura según sea el caso. El punto destino está representado por un círculo rojo sólido. Cada árbol del ambiente está representado por un punto negro sólido y tiene una elipse de covarianza asociada. Esta elipse es extraída directamente de la matriz de covarianza del vector de estados del algoritmo de SLAM (Ec. (10)) y representa la incertidumbre asociada a los parámetros (Ec. (13)) que definen la detección del tronco. La Fig. 7b muestra una experimentación más extensa del algoritmo de navegación. Como se puede ver en las Fig. 7a y 7b, cuando el robot móvil tiene que maniobrar entre surcos, el segundo planificador adquiere un rol prioritario. Esto es así puesto que generalmente, el camino medio entre surcos de cultivos es transitado y por lo tanto, su terreno tiene pocas rocas y depresiones, lo que no ocurre con los extremos de los surcos. Hay que acotar, además, que no todos los troncos del entorno son extraídos por el algoritmo de SLAM. Esto se debe a que algunos troncos son demasiado delgados o inexistentes dentro del surco. Por último, la Fig. 7c muestra la evolución de las matrices de covarianza asociadas a la detección de troncos (cada tronco se considera como una característica individual del ambiente). Como se puede ver, el determinante de la covarianza -asociado al volumen de incertidumbre de la característica- tiende a cero a medida que se ejecuta el algoritmo de SLAM. Esto último implica que el algoritmo de SLAM usado es consistente y convergente.

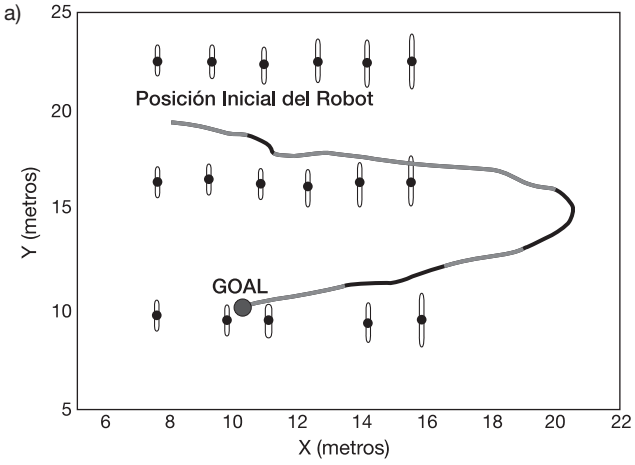
VI. Conclusión

En este trabajo se ha presentado una arquitectura de planificación de caminos paralela conmutada para la navegación de un robot móvil en entornos agrícolas. A su vez, esta estrategia de navegación tuvo incorporado un algoritmo de SLAM basado en el Filtro Extendido de Kalman para la estimación del mapa y de la pose del robot dentro del entorno. Esta información fue a su vez utilizada por los planificadores. Además, un controlador de seguimiento de caminos basado en el modelo cinemático del robot móvil tuvo a su cargo la generación de los comandos de control apropiados para la ejecución de la navegación. El entorno de trabajo fue una plantación de olivos y el mapa relegado del entorno estuvo compuesto por la ubicación de los troncos en el espacio de trabajo, asociados con los árboles detectados.

Los resultados experimentales han demostrado que la arquitectura de navegación propuesta basada en la planificación conmutada genera un camino transitable entre la posición actual del robot y el objetivo de navegación.

Los experimentos realizados en la estación experimental INTA-San Juan arrojan un comportamiento consistente y convergente del algoritmo de SLAM, como así también de una planificación exitosa (el robot llega al objetivo de navegación).

Como trabajos futuros se proponen la generación automática de puntos destinos (tales como la supervisión de troncos en la plantación) y la ampliación de la arquitectura de planificación conmutada a tareas más complejas que no estén necesariamente asociadas a la generación de caminos.



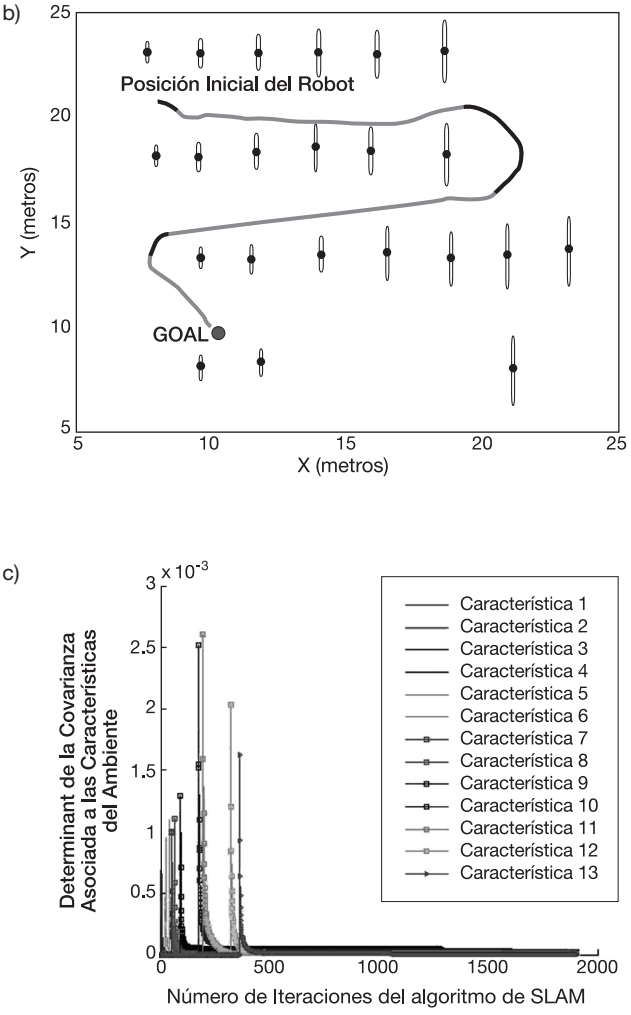


Fig. 7. Experimentaciones reales de la arquitectura de planificación conmutada.

a) y b) Muestran dos escenarios donde se aplicó la estrategia de planificación. Los tramos magenta corresponden a parte del camino que fue generada a partir del primer planificador (basado en puntos de frontera sucesivos) y los tramos en negro corresponden al segundo planificador (basado en la información del terreno). El objetivo de navegación se representa mediante un círculo sólido rojo; los troncos del ambiente se representan mediante puntos sólidos negros y cada uno de ellos tiene su respectiva elipse de covarianza asociada. c) Evolución del determinante de las covarianzas asociadas a cada característica del mapa. Como se puede observar, sus determinantes disminuyen a medida que el algoritmo de SLAM se ejecuta -y el robot revé características pasadas-, mostrando así la consistencia y convergencia del algoritmo.

Agradecimientos

Los autores agradecen al CONICET-Argentina, a la Facultad de Ingeniería de la Universidad Nacional de San Juan y a la Estación Experimental INTA-San Juan, por financiar parcialmente esta investigación.

Referencias

[1] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. Cambridge: MIT Press, 2005.

[2] R. Siegwart and I. Nourbahsh, Introduction to Autonomous Mobile Robots. Cambridge: MIT Press, 2004.

[3] H. D. Whyte and T. Bailey, "Simultaneous localization and mapping (slam): part i essential algorithms," *IEEE Robotics and Automation Magazine*, vol. 13, pp. 99–108, 2006.

[4] F. Masson, J. Guivant, J. Nieto and E. Nebot, "The Hybrid Metric Map: a Solution for Precision Farming", *Latin American Applied Research*, vol. 35, pp. 105-110, 2005.

[5] M. T. Bryson and S. Sukkarieh, "Observability Analysis and Active Control for Airborne SLAM", IEEE Transactions on Aerospace and Electronic Systems, vol. 44, pp. 261-280, 2008.

[6] R. C. Arkin, *Behavior-based Robotics*. Cambridge, USA: MIT Press, 1999.

[7] S. M. LaValle, *Planning Algorithms*, Cambridge University Press, 2006.

[8] K. H. Choset, M. Lynch, S. Hutchinso, G. Kantor, W. Bugard, L. E. Kavraki and S. Thrun, *Principles of Mobile Robot Motion: Theory, Algorithms and Implementations*, MIT Press, 2005.

[9] B. Siciliano and O. Khatib, *Springer Handbook of Robotics*, Springer-Verlag, 2008.

[10] F. A. Auat Cheein, C. De la Cruz, T. F. Basto Filho, and R. Carelli, "Solution to a door crossing problem for an autonomous wheelchair," *Proc. of the IEEE-International Conference on Intelligent Robots and Systems*, pp. 4931–4936, 2009.

[11] J. Nieto, T. Bailey and E., "Nebot, Recursive scan-matching SLAM", *Robotics and Autonomous Systems*, vol. 5, pp. 39-49, 2007.

[12] Y. Kanayama, Y. Kimura, F. Miyazaki, and T. Noguchi, "A stable tracking control method for an autonomous mobile robot," *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 384–389, 1990.

[13] T. Tao, Y. Huang, F. Sun, T. Wang. "Motion Planning for SLAM Based on Frontier Exploration". In: Proc. of the IEEE International Conference on Mechatronics and Automation. 2007.

[14] R. Bellman, *Dynamic Programming*, Princeton University Press, 1957.