Universidad Nacional de Educación a Distancia
Escuela Internacional de Doctorado

Tesis Doctoral

Estrategias de Control Basadas en Eventos Aplicadas a Robots Móviles

Rafael Socas Gutiérrez
INGENIERO DE TELECOMUNICACIÓN
MÁSTER EN INGENIERÍA DE SISTEMAS Y CONTROL

Doctorado en Ingeniería de Sistemas y Control

Director: Dr. Sebastián Dormido Bencomo
Co-director: Dra. Raquel Dormido Canto

Madrid, Mayo 2017

Departamento
Informática y Automática
E.T.S. de Ingeniería Informática, UNED

Título de la Tesis
Estrategias de Control Basadas en Eventos Aplicadas a Robots Móviles

Autor
Rafael Socas Gutiérrez

Titulación
Ingeniero de Telecomunicación
Máster en Ingeniería de Sistemas y Control

Director
Dr. Sebastián Dormido Bencomo

Co-director
Dra. Raquel Dormido Canto

Madrid, Mayo 2017

A Sergio, a Nati y a mis padres.

Resumen

Los sistemas de control basados en tiempo continuo y discreto se utilizan de forma masiva para resolver los problemas de navegación en robots móviles. Estas técnicas de control gozan de una base teórica sólida y de metodologías que facilitan su diseño y sintonía en este tipo de aplicaciones. Por otro lado, en los últimos años, las técnicas de control basadas en eventos están ganando relevancia y se están posicionando como una alternativa a los esquemas clásicos por sus ventajas diferenciales respecto al consumo de recursos.

En esta tesis, se profundiza en el estudio de las técnicas de control basadas en eventos y se aplican estos métodos a los algoritmos de navegación en robótica móvil. Los esquemas de control planteados resuelven el problema de navegación con una precisión similar a los métodos clásicos basados en tiempo continuo/discreto. Por otro lado, se definen los criterios que deben cumplir estos nuevos sistemas para garantizar su estabilidad. Como resultado, estas nuevas estructuras obtienen grandes eficiencias en el consumo de recursos, convirtiéndose así, en una alternativa a los esquemas tradicionales.

Se hace un análisis exhaustivo de los algoritmos de navegación más comunes usados en robótica móvil y se define su implementación en los nuevos esquemas de control planteados. Seguidamente, se desarrollan estos mecanismos de control para robot móviles diferenciales. Gracias a la flexibilidad de las estructuras planteadas, estas soluciones son de aplicación directa en cualquier otro tipo de robot móvil.

Se demuestra que el ruido en estos sistemas de control tiene un alto impacto en el consumo de recursos. En este sentido, se plantean mecanismos dinámicos de sintonía que minimizan estos efectos, consiguiendo además hacer estos nuevos

esquemas más inmunes a las perturbaciones.

Dada la alta dependencia que estos algoritmos tienen con la información de posicionamiento, en esta tesis se plantean nuevas metodologías de localización para robots móviles en interiores que mejoran de forma considerable los esquemas tradicionales.

Por último, a partir de este trabajo se abren nuevas líneas de investigación. Éstas van dirigidas a experimentar estos esquemas de control en vehículos móviles más complejos, analizar nuevos tipos de perturbaciones y plantear desarrollos teóricos para analizar la estabilidad de las arquitecturas de control aquí plateadas.

Palabras clave: Robots Móviles, Sistemas de Control en Red, Control Basado en Eventos, Algoritmos de Navegación, Ajuste Dinámico de Umbrales, Eficiencia en el Consumo de Recursos, Modelos de Ruido, Odometría, Algoritmos de Posicionamiento, Navegación Inercial, Robots mOway.

Abstract

The continuous-time and discrete-time control systems are massively used in navigation problems for mobile robots. These control techniques have a solid theoretical base and a lot of methodologies that help in the designing and the setting of the control system. On the other hand, in recent years, the event-based control techniques are getting relevance and they are positioning as an alternative to the classical method thanks to their differential advantages about the consumption of resources.

In this thesis, a deep analysis of the event-based control techniques is done, then these methods are applied in mobile robots navigation algorithms. The proposed control schemes resolve the navigation problem with an equivalent accuracy to classical system based on continuous or discrete-time. On the other hand, the criteria to guarantee the stability of the system are defined. As a result, these new control structures provide big efficiencies in the consumption of resources, thus becoming an alternative to classical control schemes.

The most common navigation algorithms used in mobile robotics have been investigated exhaustively and their implementation in the proposed control systems is defined. Subsequently, these control schemes are developed for differential mobile robots, although these techniques can be applied to other kind of robots in a direct way.

It is shown that the noise in these control systems has a high impact in the resources usage. In this sense, some dynamic mechanisms to set the system and to minimize the effects of the noise are proposed, getting at the same time to do these schemes more immune to the perturbations.

The accuracy of these algorithms has a high dependence of the positioning information, in this thesis, new methodologies for locating indoor mobile robots are proposed. These new algorithms improve considerably the traditional schemes.

Finally, as a future work, new lines of research can be explored. The proposed control architectures can be applied in more complex vehicles, the analysis of new perturbations have to be studied and novel theoretical methods should be proposed to study the stability of the presented control systems.

Keywords: Mobile Robots, Network Control Systems, Event-Based Control, Navigation Algorithms, Dynamic Threshold Setting, Efficiency in Consumption of Resources, Noise Models, Odometry, Positioning Algorithms, Inertial Navigation, mOway Robots.

Agradecimientos

Esta tesis se ha elaborado gracias a muchas personas e instituciones que me han ayudado durante estos años de investigación. Con su apoyo y colaboración he logrado obtener resultados satisfactorios y lo más importante de todo, que no haya perdido la motivación y la fuerza para llegar hasta el final.

A las primeras personas que quiero agradecer su apoyo y orientación es a mis directores de tesis, el Dr. Sebastián Dormido Bencomo y la Dra. Raquel Dormido Canto. Gracias a su excelente labor académica he podido finalizar mis estudios de Máster así como realizar esta tesis doctoral. Con su ayuda he podido publicar trabajos en diferentes congresos, revistas y libros de impacto a nivel internacional. Por último, su contribución ha sido fundamental para mejorar de forma notable mis capacidades de divulgación de trabajos científicos.

Quiero agradecer también al departamento de Informática y Automática de la UNED que me haya permitido el acceso a todos los recursos informáticos, bibliográficos y plataformas de robots. Gracias a estos recursos he podido desarrollar mi investigación con todos los medios necesarios.

Al Dr. Ernesto Fabregas Acosta por su ayuda y orientación en el uso de las plataformas de robots móviles, con su inestimable ayuda consiguió que en muy poco tiempo yo pudiera desarrollar una plataforma de experimentación de gran ayuda en el desarrollo de esta tesis.

No puedo dejar de mencionar al profesor Dr. Luis Gómez Déniz del departamento de Ingeniería Electrónica y Automática de la Universidad de Las Palmas de Gran Canaria. Su apoyo y sus reflexiones sobre aspectos científicos me han hecho madurar en este campo. Con él he pasado largas tardes discutiendo sobre

aspectos relacionados con esta investigación y en general sobre temas de ingeniería que tanto nos motivan a ambos. Al mismo tiempo, quiero agradecer a la Escuela de Ingenierías Industriales y Civiles (EIIC) de la Universidad de las Palmas por brindarme la oportunidad de co-dirigir dos proyectos de fin de grado durante este periodo. En este sentido, dar las gracias también a mis dos proyectandos Saúl Robaina y José Miguel Betancor por hacer realidad mi sueño de ser profesor de universidad.

Dentro del ámbito académico, el departamento de Formación de Telefónica España ha sido fuente de motivación durante los últimos 10 años que he colaborado con ellos impartiendo docencia en la compañía. Gracias a esta oportunidad que me ofrecen, cada día siento mayor motivación por la formación y por seguir ampliando mis conocimientos en el campo de la técnica y la innovación.

Durante el Máster los profesores Dr. Gonzalo Pajares Martinsanz, Dr. Alfonso Urquia Moraleda y el Dr. Sebastián Dormido Bencomo contribuyeron de forma definitiva a que me decidiese a realizar un programa de doctorado. Gracias a ellos, esta tesis doctoral es hoy una realidad. Además, en mi periodo de formación, el Instituto de Tecnologías de Georgia, la Universidad de Standford, el INTA y el COIT han sido fuentes de inspiración muy importantes en el desarrollo de este trabajo. Agradecer a estas instituciones el acceso a sus programas de formación de forma gratuita y desinteresada.

Por último y no por eso menos importante, mi agradecimiento más sincero a mi familia. A Nati por ser el soporte fundamental para que no abandonara este proyecto en los momentos más difíciles y compartir conmigo la satisfacción de conseguir el título de doctor. A mis padres, que han hecho posible el llegar hasta aquí, a ellos les debo quien soy. A Sergio por todas las horas que le he robado para elaborar este trabajo. Espero que este esfuerzo sea para él un referente en su desarrollo académico y profesional. Además deseo que le permita comprender lo importante que es la investigación en el desarrollo de la humanidad y cómo contribuye ésta a tener un mundo mejor.

Gracias a todos,

Rafael Socas Gutiérrez

Índice de Contenidos

Índice de Figuras

Índice de Tablas

Lista de Símbolos y Abreviaturas

Símbolo	**UM**	**Descripción**
t	[s]	Tiempo.
t_k	[s]	Instante en que se produce un evento.
$\mathbf{x}(t)$		Estado de un sistema continuo.
$e_r(t)$		Señal de error.
$V(x,t)$		Función de Lyapunov.
$\bar{e}$		Umbral de eventos (*Event Threshold*).
$\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}$		Matrices del sistema lineal.
$u(t)$		Entrada de un sistema continuo.
$y(t)$		Salida de un sistema continuo.
$d(t)$		Perturbación de un sistema continuo.
$v(t)$		Ruido en los sensores de un sistema continuo.
$w(t)$		Señal de referencia de un sistema continuo.
$u[n]$		Entrada de un sistema discreto.
$y[n]$		Salida de un sistema discreto.
$d[n]$		Perturbación de un sistema discreto.
$v[n]$		Ruido en los sensores de un sistema discreto.
$w[n]$		Señal de referencia de un sistema discreto.
$v_l(t)$	[cm/s]	Velocidad lineal rueda izquierda.
$v_r(t)$	[cm/s]	Velocidad lineal rueda derecha.
r	[cm]	Radio de las ruedas.
L	[cm]	Distancia entre las ruedas.
$\dot{\theta}_l(t)$	[rad/s]	Velocidad angular rueda izquierda.
$\dot{\theta}_r(t)$	[rad/s]	Velocidad angular rueda derecha.
$v_Q(t)$	[cm/s]	Velocidad lineal del punto medio del robot.
x	[cm]	Posición en el eje X del robot.
y	[cm]	Posición en el eje Y del robot.

Símbolo	**UM**	**Descripción**
z	[cm]	Posición en el eje Z del robot.
$\mathbf{p}$	[cm]	Vector posición del robot.
$\hat{\mathbf{p}}$	[cm]	Estimación del vector posición del robot.
P_e	[cm]	Error de posicionamiento.
$\dot{\mathbf{p}}$	[cm/s]	Vector velocidad del robot.
$\mathbf{J}$		Jacobiano del robot diferencial.
R	[cm]	Radio de curvatura instantáneo.
$\boldsymbol{\tau_l}$	[N·m]	Momento rueda izquierda.
$\boldsymbol{\tau_l}$	[N·m]	Momento rueda derecha.
ρ	[cm]	Error de distancia.
α	[rad]	Error de orientación.
K_ρ	[1/s]	Factor de aproximación lineal.
K_α	[1/s]	Factor de aproximación angular.
(x_c, y_c)	[cm]	Posición inicial.
(x_g, y_g)	[cm]	Posición objetivo.
d_l	[cm]	Distancia recorrida por la rueda izquierda.
d_r	[cm]	Distancia recorrida por la rueda derecha.
d_c	[cm]	Distancia recorrida media.
e_ρ	[cm]	Condición de parada.
$\bar{e}_\alpha$	[rad]	Umbral de eventos para el algoritmo DD.
T_c	[s]	Tiempo de convergencia.
φ_{AO}	[rad]	Orientación para evitar obstáculos.
$\bar{e}_{EO}$	[cm]	Umbral de eventos para el algoritmo EO.
$\bar{e}_{SP}$	[cm]	Umbral de eventos para el algoritmo SP.
N_N		Ratio de eventos.
N_R		Ratio de actividad empírico.
η_N	[%]	Eficiencia máxima teórica.
η_R	[%]	Eficiencia en el consumo de un recurso R.
BW	[bps]	Ancho de banda.
η_{BW}	[%]	Eficiencia en el consumo de ancho de banda.
η_{CC}	[%]	Eficiencia en el consumo de recursos computacionales.
η_{CE_C}	[%]	Eficiencia en el consumo de energía en el controlador.
η_{CE_R}	[%]	Eficiencia en el consumo de energía en el robot.
$n(t)$		Señal de ruido en el dominio continuo.

Símbolo	**UM**	**Descripción**
$n[n]$		Señal de ruido en el dominio discreto.
$\hat{n}[n]$		Estimación señal de ruido en el dominio discreto.
μ		Media de la señal de ruido.
σ		Desviación típica de la señal de ruido.
$\bar{e}_d[n]$		Componente dinámica del umbral de eventos.
l		Longitud de la memoria del calculador de umbral.
$Per(N, p)$		Percentil p de la muestra N.
$E_x[n]$	[cm]	Error de posicionamiento en el eje X.
$E_y[n]$	[cm]	Error de posicionamiento en el eje Y.
θ	[rad]	Ángulo de elevación.
ψ	[rad]	Ángulo de alabeo.
a_x	[g]	Aceleración eje X del robot.
a_y	[g]	Aceleración eje Y del robot.
a_z	[g]	Aceleración eje Z del robot.
h	[°]	Banda de guarda inferior.
H	[°]	Banda de guarda superior.
$t_{hx}[n]$	[°]	Umbral de la componente x de la aceleración.
$t_{hy}[n]$	[°]	Umbral de la componente y de la aceleración.
t_{min}	[s]	Tiempo mínimo del sistema de sincronización de la orientación.
φ_o	[rad]	Orientación obtenida a través de odometría.
φ_a	[rad]	Orientación obtenida a través de medidas inerciales.
f_s	[Hz]	Frecuencia de muestreo.
v_w	[1/s]	Velocidad de aproximación a la pared.
w	[cm]	Señal de referencia para los algoritmos EO y SP.
θ_{tilt}	[°]	Inclinación del eje Y del robot.
ψ_{tilt}	[°]	Inclinación del eje X del robot.

Abreviatura	**Descripción**
CCD	Control Digital Directo.
SCCD	Sistema de Control por Computador Descentralizado.
NCS	*Network Control System.*
MADB	*Maximum Allowable Delay Bound.*
SISO	*Single-Input Single-Output.*
MEMS	*Micro Electro Mechanical Systems.*

Abreviatura	**Descripción**
WNS	*Wireless Sensor Networks.*
EBC	*Event-Based Control.*
MAC	*Minimum Attention Control.*
AAC	*Anytime Attention Control.*
ZOH	*Zero Order Hold.*
GUUB	*Globally Ultimately Uniformly Bounded.*
EG	*Event Generator.*
CIG	*Control Input Generator.*
UAV	*Unmanned Aerial Vehicle.*
WMR	*Wheeled Mobile Robots.*
LMR	*Legged Mobile Robots.*
AUV	*Autonomous Underwater Vehicle.*
DOF	*Degree Of Freedom.*
ICC	*Instantaneous Center of Curvature.*
A/D	Analógico/Digital.
DD	Algoritmo Directo al Destino (*Go to Goal*).
EO	Algoritmo para Evitar Obstáculos (*Avoid Obstacles*).
SP	Algoritmo para el Seguimiento de Paredes (*Wall Following*).
IAE	*Integral Absolute Error.*
IAEP	*Integral Absolute Error compared to Periodic loop.*
PI	Control Proporcional Integral.
MIMO	*Multiple-Input Multiple-Output.*
VA	Variable Aleatoria.
EMV	Estimador de Máxima Verosimilitud.
BBDD	Bases de Datos.
GPS	*Global Positioning System.*
ME	Máquina de Estados Finitos.
IIR	*Infinite Impulse Response.*
UL	*UpLink.*
DL	*DownLink.*

Capítulo 1

Introducción

La idea de usar sistemas digitales para fines de control surge en la década de los 50. En aquella época, los sistemas de computación eran lentos y poco fiables, con capacidades de memoria y computación muy limitadas siendo su campo de aplicación el registro y la gestión de datos. A medida que la fiabilidad iba mejorando, estos sistemas de computación se comenzaron a utilizar como sistemas de supervisión de operaciones.

En 1962 se produjo un avance radical a cargo del Imperial Chemical Industries Ltd. en el Reino Unido: la instalación del ordenador Ferranti Argus capaz de medir 224 variables y manipular directamente 129 válvulas. Es la primera vez que un ordenador se integra directamente a controlar un sistema industrial, considerándose el inicio de la era del Control Digital Directo (CDD). El crecimiento del CDD fue explosivo desde entonces, ayudado por los bajos costes y por el aumento del rendimiento y la fiabilidad de la tecnología digital. Mientras que las primeras implementaciones del CDD fueron restringidas a los enlaces dedicados entre controlador y actuadores/sensores, las necesidades de los usuarios y los avances en las tecnologías de las comunicaciones propiciaron el primer Sistema de Control por Computador Descentralizado (SCCD) a mediados de los años 70.

Los sistemas de control descentralizados se integraron rápidamente en la empresas manufactureras y en la industria en general. El primer trabajo analizando el uso del control descentralizado en la industria aparece también por esa época (Duffie [37]). Importantes trabajos que trataban los aspectos fundamentales de los sistemas de control descentralizado se desarrollaron también por esa época. Por ejemplo, la idea de un reloj global como base fundamental para las aplicaciones descentralizadas se presentó en Kopetz [67], además de contemplar protocolos de

comunicaciones basados en datagramas más acordes a las aplicaciones de tiempo real en vez de los convencionales basados en retransmisiones. Los niveles de descentralización fueron definidos en Enslow [41] y Franta et al. [47]. Estas ideas dieron lugar a una nueva rama de la teoría de control de cuyas aplicaciones más relevantes proviene la tecnología del bus de campo (p.e. FIP y PROFIBUS) y otros buses de comunicaciones como el CAN empleado con éxito durante décadas en la industria de procesos y automatización.

El importante avance que han tenido las tecnologías de la comunicación en las últimas décadas, que se ha materializado en los nuevos protocolos, algoritmos de conmutación y enrutamiento y en las redes basada en paquetes, ha despertado un interés creciente en la comunidad de control. El uso de una red compartida multiuso para conectar elementos de control descentralizado produjo mejoras en términos de arquitecturas más flexibles, costes de instalación y mantenimiento, además de una mayor fiabilidad en las comunicaciones respecto a las tradicionales basadas en bus. Este cambio de paradigma planteó también nuevos retos (Murray et al. [98]).

Los Sistemas de Control en Red (*Networked Control Systems* NCS) son sistemas descentralizados en los que la comunicación de los diferentes elementos del bucle de control (sensores, actuadores, y controladores) emplea una red de comunicación digital compartida. Véase la Figura 1.1.

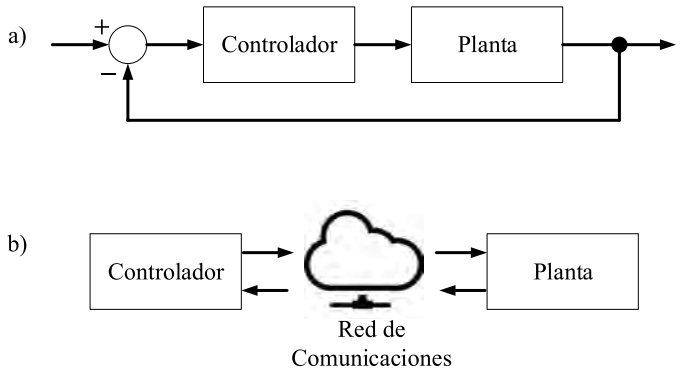

Figura 1.1: Sistemas de control. a) Esquema de control clásico, b) Sistema de Control en Red (NCS).

Por tanto, los sistemas NCSs se tratan en una disciplina que combina las teorías del control y de las comunicaciones. Favorecido por la gran cantidad de aplicaciones y aspectos a resolver, en los últimos años los sistemas NCS se han convertido en un problema común para muchos grupos de investigación de control de todo el mundo. De hecho, al menos dos de las áreas técnicas de la Federación

Internacional de Control Automático (IFAC) se dedican a este campo. También un nuevo IEEE Transactions sobre este tema se puso en marcha en 2014. Además existe un número creciente de conferencias especializadas como el IFAC Workshop on Distributed Estimation and Control in Networked Systems (NecSys) o el International Symposium on Control Systems (SICE).

1.1. Sistema de Control en Red - NCS

Típicamente, un sistema de control se compone de los siguiente elementos: sistema o planta a controlar, sensores que miden las salidas de la planta y las transmiten al controlador y finalmente el controlador que recibe las salidas de la planta, genera las señales de control y las envía a los actuadores para que las apliquen a la planta. Enlaces de comunicación punto a punto entre los diferentes elementos transmiten las señales de los sensores y de control sobre un canal de comunicación que se considera ideal. Este medio de comunicación se aproxima a un canal con ausencia de retardos, manteniéndose la integridad de la información y con un ancho de banda ilimitado (Figura 1.1 a)).

Si se considera ahora un sistema de control en red NCS, éste se distingue de un esquema de control clásico en que existe una red de comunicaciones que afecta al bucle de control (Figura 1.1 b)). En este caso, no puede asumirse un canal de comunicaciones ideal ya que las redes de comunicación estándar generalmente están diseñadas para preservar la integridad de los datos, pero no tienen por qué satisfacer los estrictos requerimientos en tiempo real del lazo de control. Por otro lado, hay una serie de beneficios cuando se utiliza una red digital de comunicación genérica en el lazo de control. Las ventajas pueden enumerarse en:

- *Bajo coste*: Utilizar una comunicación punto a punto en sistemas a gran escala o en plantas geográficamente distribuidas generalmente es una solución costosa y poco práctica. Una red inalámbrica(wireless) o incluso redes con cables pueden reducir el número de conexiones y la longitud de los cables necesarios y conseguir así reducir los costes de implementación y mantenimiento.

- *Fiabilidad*: Los protocolos de comunicaciones en una topología de red mallada intrínsecamente mejora la fiabilidad. Además, los enrutamientos dinámicos permiten encontrar rutas alternativas en caso de fallos en los enlaces.

- *Mantenimiento*: La reducción en la complejidad de cableado facilita el diagnóstico y mantenimiento del sistema.
- *Flexibilidad*: Los sistemas de red estructurados ofrecen arquitecturas flexibles que facilitan la reconfiguración de las diferentes partes del sistema y permiten incorporar nuevos dispositivos de forma sencilla.
- *Accesibilidad*: Estos sistemas ofrecen nuevas características tales como la modularidad, control descentralizado o diagnóstico integrado.

También, en un gran número de situaciones prácticas la utilización de redes de comunicación para conectar los diferentes elementos de los sistemas de control incorpora grandes ventajas al sistema. Se detallan a continuación algunos ejemplos:

- *Limitaciones de espacio y peso*: Limitaciones de este tipo deben realizarse por ejemplo en aviónica (aviones comerciales, vehículos aéreos no tripulados) o sistemas integrados en la industria automotriz.
- *Cobertura de grandes distancias*: Plantas químicas, fábricas de gran escala y sistemas de automatización.
- *Control de aplicaciones donde el cableado no es posible*: Flota de vehículos autónomos, sistemas teleoperados, robots móviles, etc.

1.2. Campos de Investigación

A finales de la década de los 90, los investigadores comenzaron a identificar los principales problemas dentro del campo de los NCSs. Dichos problemas son los que han marcado las principales aportaciones en este campo desde esa fecha hasta el momento actual. Los temas más relevantes a tratar dentro de los sistemas NCS se enumeran de forma resumida en las siguientes secciones.

1.2.1. Retardo y Pérdida de Paquetes

En general, hay dos métodos para manejar los retrasos inducidos por la red. El primer método se basa en diseñar los algoritmos de control considerando los retardos de la red, estas son las ideas que se desarrollan en Yang [153] y Mahmoud

and Ismail [80]. El segundo método se basa en reducir los retardos compartiendo los recursos comunes de la red. En Jianyong et al. [61] y Li et al. [71] la investigación se ha focalizado en cómo gestionar los recursos de la red para reducir al máximo los retardos. Otros enfoques abordan el problema desde una perspectiva de control robusto que garantice la estabilidad y el rendimiento a pesar de la presencia de retrasos. En muchos de estos diseños, se establece un límite máximo de retardo permitido (*Maximum Allowable Delay Bound* MADB). El MADB se define como el intervalo máximo permitido desde el instante en que el sensor obtiene los datos de una planta hasta el instante en que los actuadores aplican a la planta correspondiente acciones de control. Para garantizar que un NCS es estable, los periodos de muestreo deben ser inferiores a los correspondientes límites de máximo retardo permisible (MADBs) (Yue et al. [156], Yue et al. [157], Millán et al. [88]).

Otra diferencia importante entre NCSs y el control digital estándar es la posibilidad de que se puedan perder los paquetes de datos en el tránsito a través de la red. Para una determinada frecuencia de muestreo, la aplicación de métodos de estimación en un NCS pueden reducir el tráfico de red y aumentar el ancho de banda efectivo del sistema y por tanto, disminuir la probabilidad de pérdida de paquetes (Zhang et al. [159]).

1.2.2. Canales con Ancho de Banda Limitado

Un esfuerzo significativo de investigación se ha dedicado al problema de determinar la tasa mínima de transmisión necesaria para estabilizar un sistema lineal a través de retroalimentación (Braslavsky et al. [21], Hespanha et al. [58], Elia and Mitter [39]). Posteriormente, también se han hecho algunos progresos de cara a definir la estabilidad en problemas con sistemas no lineales (Liberzon and Hespanha [75], Nair et al. [100]). Otros trabajos se han centrado en la obtención de las condiciones de estabilidad basado en la información transmitida en cada instante (Sahai [118]) y en el estudio de las limitaciones de rendimiento en los sistemas realimentados mediante canales sin memoria y de capacidad finita (Martins and Dahleh [83]).

1.2.3. Estabilidad de los Sistemas NCS

En sistemas de tiempo real, particularmente en sistemas de control, los retrasos o la pérdida de paquetes pueden causar inestabilidad en el proceso. Por otra parte, en los sistemas NCS se puede inducir inestabilidad debido al hecho de que existan retrasos variables, esto puede afectar al sistema incluso cuando el NCS es asintóticamente estable para cualquier constante de retraso (Wittenmark et al. [150]).

La estabilidad en estas circunstancias ha sido analizada por un gran número de investigadores. Los primeros resultados se obtuvieron aplicando las herramientas clásicas como en Kao and Lincoln [63], donde se propone un criterio de estabilidad en el dominio de la frecuencia, basado en el teorema de pequeña ganancia, para investigar la estabilidad de plantas NCS de una sola entrada y una sola salida (SISO). Un enfoque diferente de modelado se utiliza en Naghshtabrizi and Hespanha [99] y Yu et al. [155], donde se propone una descripción de tiempo continuo con un controlador basado en un retenedor de orden cero. Otros resultados relevantes relativos a la estabilidad de los sistemas NCS se pueden encontrar en Li et al. [72], Heemels and van de Wouw [57], Cloosterman et al. [27], Zhang et al. [161] y Liu and Zhang [77].

1.2.4. Eficiencia Energética

En todos los campos de la ingeniería la eficiencia energética está ganando mucha relevancia debido a las preocupaciones económicas y ambientales. En los sistemas de control en red, especialmente si se emplean dispositivos con baterías en entornos wireless, el ahorro de energía es clave para aumentar la vida útil del sistema e indirectamente, reducir costes. Por otro lado, en algunas aplicaciones, los dispositivos de la red se pueden implementar en lugares peligrosos o inaccesibles y puede ser costoso o poco práctico reemplazar las baterías. Esto ha motivado el interés actual en el desarrollo de metodologías NCS que consideren como requisito imprescindible reducir el consumo de energía. En particular, se investiga en protocolos para reducir el ratio de transmisiones en el sistema, ya que como es bien sabido, los dispositivos inalámbricos consumen la mayor parte de la energía cuando está transmitiendo el interfaz radio. Estas ideas y metodologías se analizan en Shah and Rabaey [124] y Arisha et al. [8].

1.2.5. Redes de Sensores Inalámbricos

Un factor tecnológico que sin duda ha amplificado el impacto de las NCSs, tanto en aplicaciones de la industria como en el interés del mundo académico, ha sido el rápido desarrollo de tecnologías inalámbricas en los últimos años. Los últimos avances en miniaturización, tales como MEMS (*Micro Electro Mechanical Systems*) y las nano-tecnologías, han permitido el desarrollo de dispositivos inalámbricos de baja potencia, bajo coste y con la capacidad de establecer redes de sensores inalámbricos WSN (*Wireless Sensor Networks*). Este tipo de tecnologías hará posible la monitorización y la automatización de los sistemas de control dispersos a una escala sin precedentes (Mon et al. [95], Chaudhary et al. [25]).

1.2.6. Sistemas Distribuidos

El desafío para este campo es pasar de la visión tradicional de sistemas de control como un único proceso con un único controlador, a sistemas de control como una colección heterogénea de elementos de información con interacciones e interconexiones complejas. En este contexto, se presentan nuevas posibilidades y retos. Además de cuestiones como el control y estimación distribuido sobre redes WSN, control NCS eficiente o sistemas multi-agente son temas de gran actualidad. Las técnicas de estimación distribuidas han sido desarrolladas considerando diferentes niveles de imperfección en los canales de comunicación (Zhang et al. [158], Wu and Cheng [151], Kibangou [66]). En este campo, también se plantean enfoques unificados de estimación en sistemas de control distribuido como puede encontrarse en Orihuela et al. [106] y Millán et al. [87].

1.3. Control Basado en Eventos - EBC

En este nuevo contexto de los sistemas de control, las técnicas de control basadas en eventos (*Event-Based Control* EBC) pueden resolver multitud de problemas en sistemas de control en red (NCS) (Åström and Bernhardsson [13], Åström [12]). Si se compara con las técnicas de control continuo (Khalil [64], Trentelman et al. [140]) o discreto (Åström [11], Ogata [104]), en esta estrategia de control los eventos son generados solo cuando es necesario (p.e. si cierta variable cruza un determinado umbral) estando el resto del tiempo el sistema en reposo. Como

consecuencia se reduce el número de mensajes que transitan por la red, ya que dependen de los eventos generados. Además el número de acciones de control también disminuye considerablemente. Como resultado disminuye el tráfico en la red y consecuentemente todos los problemas asociados a la comunicación, como pueden ser retardos o pérdidas de mensajes, se vuelven menos críticos. Los dispositivos inalámbricos que basen su modo de trabajo en estas técnicas harán un uso más eficiente de las baterías al reducir el volumen de comunicaciones consiguiendo que éstas tengan una mayor vida útil. Los elementos activos como válvulas, relés, motores, etc., sufrirán menos desgaste al recibir menos acciones de control. Además este esquema de control tiene menos latencias que los sistemas discretos ya que no es necesario esperar al siguiente periodo para tomar una acción de control.

El control basado en eventos no es una idea nueva y tiene sus orígenes a finales de los años 50 cuando Ellis [40] plantea un método de muestreo que consiste en transmitir datos solo cuando existe un cambio significativo en la señal que justifique la adquisición de una nueva muestra. En estos años, una gran variedad de esquemas de control han sido propuestos basados en estas ideas y han demostrado en ciertas aplicaciones grandes ventajas respecto a los métodos de control clásico (Dormido et al. [36]). Por otro lado, se continúa haciendo grandes esfuerzos por desarrollar una base matemática que describa el comportamiento formal de estos sistemas.

1.3.1. Control basado en Eventos en la Literatura

En la mayoría de las implementaciones, un evento se activa cuando alguna función de error excede un límite tolerable. Según como se definan estas funciones de error y este límite se obtienen los diferentes enfoques que se describen en la literatura sobre los sistemas de control basados en eventos. A continuación se hace una recapitulación de la ideas y aportaciones más relevantes que se han hecho en esta rama del control.

Control *Send-on-Delta* o por Cruce de Nivel

El error $e_r(t)$ se define como la diferencia entre el estado en el instante actual y el estado en el instante de la última muestra según se denota en la siguiente

ecuación.

$$\|e_r(t)\| = \|x(t) - x(t_k)\| \leq \bar{e} \tag{1.1}$$

donde t_k hace referencia al instante de tiempo del último evento y t es el tiempo presente. El valor del umbral de disparo $\bar{e}$ determina el rendimiento del sistema y la región en torno al equilibrio en la que el estado permanece confinado Figura 1.2 a) y b). Trabajos relevantes en este sentido pueden encontrarse en Heemels et al. [56], Miskowicz [91] y Sandee [120].

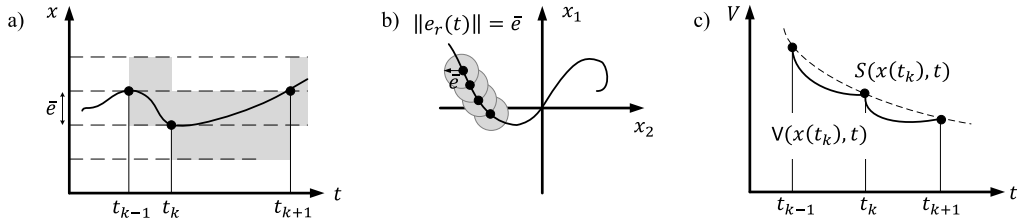

Figura 1.2: Reglas de generación de eventos.

Métodos basados en Funciones de Lyapunov

La estrategia por cruce de nivel no asegura la estabilidad asintótica del sistema. Por esta razón, se han investigado otro tipo de reglas de disparo que permitan alcanzar esta propiedad. En Tabuada [138] se presenta esta idea donde el error está acotado por el estado del sistema mediante:

$$\|e_r(t)\| = \|x(t) - x(t_k)\| \leq \sigma\|x(t)\| \tag{1.2}$$

Esta estrategia permite que el sistema sea asintóticamente estable pero presenta el inconveniente de que el tiempo entre eventos disminuye a medida que el sistema se acerca al equilibrio. Este aspecto es importante en el control basado en eventos ya que el efecto Zeno[1] debe ser evitado. Para ello, se escoge el parámetro σ de acuerdo con ciertas propiedades de la función de Lyapunov. Otros autores han analizado la idea de usar la función de Lyapunov $V(x,t)$ directamente en la función de disparo.Tal es el caso de Mazo et al. [84], donde un evento se produce cuando $V(x,t)$ alcanza una cierta cota de desempeño $S(x,t)$, (véase Figura 1.2 c)).

[1]El fenómeno Zeno (chattering) consiste en una acumulación de impactos en tiempo finito o bien una acumulación de infinitos cambios en el sistema. Por ejemplo, si el sistema modela el funcionamiento de una pieza o el conmutador de una máquina en éste se producirá un excesivo desgaste.

Funciones de Disparo Variables con el Tiempo

Recientemente, se han propuesto como alternativa para la estabilidad asintótica funciones de disparo variables con el tiempo. Por ejemplo en Guinaldo et al. [52] y Seyboth et al. [123] éstas han sido utilizadas para sistemas lineales interconectados y sistemas multi-agente, respectivamente, acotándose el error según la siguiente ecuación.

$$\|e_r(t)\| \leq \bar{e} \cdot \exp(-\alpha t), \quad \alpha > 0 \tag{1.3}$$

Este mecanismo garantiza una cota inferior para el tiempo entre eventos, pudiéndose incluso eliminar el efecto Zeno bajo condiciones de red no ideales.

Self-Triggering

Las redes de sensores son un caso especial de NCR en las que el consumo de energía tiene un papel fundamental. Por tanto, el muestreo basado en eventos resulta conveniente para reducir el número de transmisiones. Sin embargo, tal como se analiza en Anta and Tabuada [6] y Araújo [7], la mayor parte de la energía consumida por el sensor se debe a la monitorización de la señal y no tanto a la transmisión. Las reglas de disparo analizadas hasta ahora requieren de la medida continua del estado. Por este motivo, ha emergido en los últimos años una nueva estrategia conocida como *self-triggering*, o auto-disparo. La filosofía de los sistemas que trabajan con *self-triggering* calculan el tiempo t_{k+1} en el cual tendría lugar la siguiente ejecución de las tareas relacionadas con el control en función de la medida del último estado x_k. De esta manera, el sensor no monitoriza el proceso hasta que es activado en el instante temporal t_{k+1}, adquiere la medida, la transmite y recalcula el tiempo de la próxima ejecución. El concepto de *self-triggering* fue propuesto por primera vez en Velasco et al. [146]. Un problema inherente a este esquema es el efecto de la existencia de incertidumbre en el modelo o la aparición de perturbaciones externas desconocidas.

Técnicas de Optimización para el Intervalo de Muestreo

Dentro de esta categoría se pueden nombrar dos propuestas. La primera, el control por mínima atención (MAC, *Minimum Attention Control*) que maximiza el siguiente tiempo de ejecución, mientras que garantiza un cierto nivel en el rendimiento del sistema (Anta and Tabuada [5], Donkers et al. [32]). La política es

similar a la de *self-triggering* en el sentido de que el objetivo es que las tareas de control sean ejecutadas lo mínimo posible pero sin utilizar técnicas de emulación. Aunque en Donkers et al. [32] se propone un diseño que permite aliviar el problema de carga computacional que presentaba Anta and Tabuada [5], el control MAC es mucho menos robusto frente a retardos y perturbaciones que el control basado en eventos. La segunda propuesta, el control con atención en cualquier momento (AAC, *Anytime Attention Control*) (Greco et al. [51], Gupta [53]) presenta el mismo tipo de problema. El diseño que se propone en Anta and Tabuada [5] asume que después de cada ejecución de la tarea de control, la señal de control no puede ser recalculada durante un cierto intervalo de tiempo determinado por otro elemento del sistema (el planificador), calculando así la señal de control que maximiza el rendimiento del sistema en lazo cerrado.

Control Periódico basado en Eventos

Esta estrategia, como su propio nombre indica, combina el control periódico y el control basado en eventos, y evita la continua monitorización de las salidas del sistema como en el caso del *self-triggering*, evaluando la condición de disparo únicamente en ciertos instantes de tiempo dados por un periodo de muestreo T_s. Este idea fue formulada Heemels and Donkers [54], y usa funciones de disparo basadas en funciones de Lyapunov, donde como parámetro de diseño, además de σ, está T_s. Se debe resaltar aquí que el efecto Zeno queda excluido, ya que se garantiza que el intervalo entre eventos es de al menos un periodo de muestreo.

Control basado en Modelo y Disparo por Eventos

Todas las estrategias descritas hasta ahora consideran que el actuador mantiene constante la señal de control entre eventos (ZOH, *Zero Order Hold*). A pesar de que bajo esta consideración se simplifica el análisis, se ha demostrado que, si existe un modelo preciso de la planta, un generador de la señal de control puede tratar de emular el comportamiento de un sistema convencional de control en lazo cerrado para obtener un mejor rendimiento que con ZOH (Lunze and Lehmann [79]). La idea de sacar ventaja del conocimiento de un modelo de la planta en NCS se introduce por primera vez en Montestruque and Antsaklis [96], aunque las ejecuciones son periódicas y no basadas en eventos. No obstante, este tipo de estrategias como la propuesta en Lunze and Lehmann [79] requieren sincronización de todos los elementos en el lazo de control, lo que la hace difícilmente

extensible a sistemas de control remoto y a esquemas distribuidos.

1.3.2. Estabilidad de los Sistemas de Control basados en Eventos

Un sistema lineal continuo e invariante en el tiempo está descrito por las Ecuaciones 1.4 y 1.5

$$\dot{\mathbf{x}}(t) = \mathbf{A} \cdot \mathbf{x}(t) + \mathbf{B} \cdot \mathbf{u}(t) + \mathbf{E} \cdot \mathbf{d}(t) \tag{1.4}$$

$$\mathbf{y}(t) = \mathbf{C} \cdot \mathbf{x}(t) \tag{1.5}$$

donde $\mathbf{x} \in \mathbb{R}^n$ se corresponde con el estado del sistema, $\mathbf{u} \in \mathbb{R}^m$ e $\mathbf{y} \in \mathbb{R}^r$ son la entrada y la salida del sistema respectivamente, finalmente $\mathbf{d} \in \mathbb{R}^l$ representa las perturbaciones que afectan a la planta. Además $\mathbf{A} \in \mathbb{R}^{n \times n}$, $\mathbf{B} \in \mathbb{R}^{n \times m}$, $\mathbf{E} \in \mathbb{R}^{n \times l}$ y $\mathbf{C} \in \mathbb{R}^{r \times n}$ son matrices reales.

Definición 1. *Una matriz cuadrada* $\mathbf{A}$ *se denomina Hurwitz si cada autovalor*

$$\lambda_i\{\mathbf{A}\}, \quad i \in 1, 2, 3, \cdots, n$$

de $\mathbf{A}$ *tiene su parte real estrictamente negativa.*

Definiciones análogas se pueden dar para un sistema de tiempo discreto. En ese caso, la condición sobre los autovalores es que éstos estén contenidos dentro del círculo unidad.

Teorema 1. *El sistema descrito por las Ecuaciones 1.4 y 1.5 con entrada nula* ($\mathbf{u}(t) = 0$ *y* $\mathbf{d}(t) = 0$ *para todo instante de tiempo* $t \geq 0$) *es asintóticamente estable, es decir,*

$$\lim_{t \to \infty} \|\mathbf{x}(t)\| = 0 \ ,$$

si la matriz $\mathbf{A}$ *es Hurwitz.*

Sistema GUUB (*Globally Ultimately Uniformly Bounded*)

En los sistemas de control basados en eventos, la estabilidad asintótica definida anteriormente no puede garantizarse, por tanto, es más apropiado en este tipo de sistemas definir el criterio de estabilidad GUUB (véase Figura 1.3).

Definición 2. *La solución* $\mathbf{x}(t)$ *del sistema continuo descrito por las Ecuaciones 1.4 y 1.5 es GUUB si para cada* $\mathbf{x}(0) \in \mathbb{R}^n$ *existe una constante positiva* a *y un instante de tiempo* $\bar{t}$ *que cumple*

$$\mathbf{x}(t) \in \Omega_t = \{\mathbf{x} : \|\mathbf{x}\| \leq a\}, \quad \forall t \geq \bar{t}$$

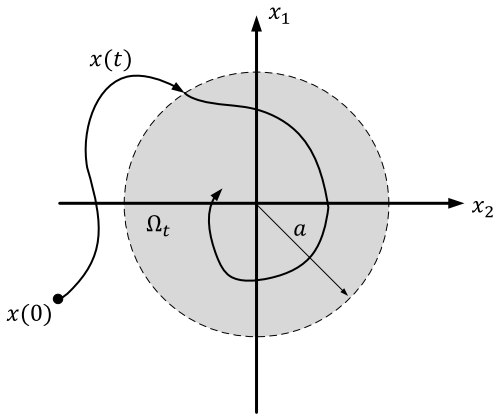

Figura 1.3: Definición de estabilidad GUUB.

Definición 3. *El sistema continuo descrito por las Ecuaciones 1.4 y 1.5 se dice que es UB (Ultimately Bounded) o estable si sus estados son GUUB.*

Teorema 2. *El estado* $\mathbf{x}(t)$ *del sistema continuo con Ecuaciones 1.4 y 1.5 es GUUB si la matriz* $\mathbf{A}$ *es Hurwitz y la entrada* $\mathbf{u}(t)$ *y la perturbación* $\mathbf{d}(t)$ *están acotadas.*

En la literatura hay más definiciones de estabilidad equivalentes como puede ser la *input-to-state stability* (Khalil and Grizzle [65], Mazo Jr and Tabuada [86], Sontag [134]) o la *practical stability* (Fagnani and Zampieri [44], Lakshmikantham et al. [68]). En resumen todos estos conceptos de estabilidad se refieren a la misma situación, es decir, que el estado debe introducirse en una región deseada Ω_t y permanecer allí para todo tiempo futuro a pesar de perturbaciones exógenas.

1.3.3. Arquitectura de Control basada en Eventos

Tal como se ha visto anteriormente, en la literatura pueden encontrarse diferentes variantes de estructuras de control basadas en eventos donde cada una de ellas busca un compromiso entre la robustez del sistema y la eficiencia en el uso de recursos (Årzén [9], Pawlowski et al. [110], Sánchez et al. [119]). En esta tesis se propone un esquema de control siguiendo algunas de las ideas presentadas en Lunze and Lehmann [79] cuya arquitectura se presenta en la Figura 1.4.

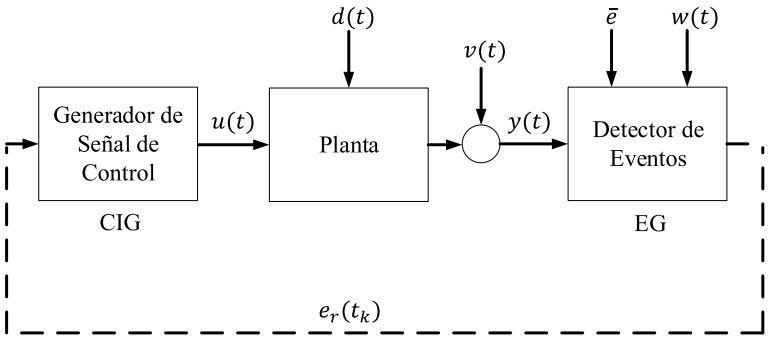

Figura 1.4: Arquitectura general de control basado en eventos.

El esquema de control propuesto se compone de los siguientes elementos:

1. Planta o proceso a controlar el cual dispone de una entrada $u(t)$ y una salida $y(t)$. La salida está influida por las perturbaciones que afectan al proceso $d(t)$ y el ruido presente en los sensores $v(t)$.

2. Un detector de eventos (*Event Generator* EG) el cual a partir de la señal de salida del proceso $y(t)$, la señal de consigna $w(t)$ y un umbral de eventos (*event threshold*) $\bar{e}$, genera la señal de error $e_r(t_k)$. Esta señal de error se envía al generador de señal de control a través del lazo de control en los instantes t_k en los que se produzca un evento en el sistema. El umbral $\bar{e}$ determinará en qué grado puede desviarse la salida con respecto a la consigna y en caso de superarse éste el sistema genere un evento.

3. Finalmente, un generador de señal de control (*Control Input Generator* CIG) cuya misión es generar las señales de control $u(t)$ que gobiernan la planta.

Uno de los parámetros fundamentales de esta arquitectura es el umbral de eventos $\bar{e}$ el cual controla de forma directa las prestaciones globales y estabilidad

del sistema. Este parámetro también influye de forma directa en el número de eventos generados y en consecuencia en el tráfico de la red, en la carga de proceso del controlador y, en general, en el uso de los recursos del sistema. Al ser un parámetro con tanta influencia en la respuesta del sistema, en esta tesis se explorarán diferentes técnicas para sintonizar dicho parámetro de forma dinámica (Romero et al. [117], Socas et al. [127]) en función del comportamiento del proceso o de las perturbaciones que afecten a las medidas de los sensores.

1.3.4. Aplicaciones

La principal razón por la que se usa control en tiempo continuo o en tiempo discreto en gran parte de las aplicaciones es que se dispone de una teoría bien establecida para el análisis y el diseño del lazo de control. Sin embargo, estos enfoques también tienen algunos inconvenientes:

1. La transferencia de información en tiempo continuo sólo es posible si se implementa el bucle de control sobre hardware analógico. Por otro lado, hoy en día casi todos los controladores se basan en hardware digital, por lo que la teoría de control en tiempo continuo sólo se puede aplicar si se cumplen ciertos criterios en los intervalos de muestreo (Ogata [104]).

2. El bucle de control en tiempo discreto es ejecutado por un reloj y, por tanto, independientemente del comportamiento del sistema. En consecuencia, durante intervalos de tiempo en los que las variables del sistema no cambian significativamente no sería necesario retroalimentar la información para cumplir con los requisitos de control del sistema. Por ejemplo, si considera el sistema cuando está en estado estacionario, los recursos cálculo y de comunicación se malgastan innecesariamente (Årzén [9]).

3. Por último, cuando hay perturbaciones que afectan al sistema, el control de tiempo discreto no es capaz de contrarrestar inmediatamente estos efectos debido a que tiene que esperar al siguiente período de muestreo dado por un reloj externo. En este caso, una inadmisible degradación de las prestaciones del sistema solamente se puede evitar utilizando un intervalo de muestreo convenientemente corto. Sin embargo, esto podría ser otra vez inadecuado en el sentido de perder recursos de cálculo y comunicaciones cuando el sistema se encuentra en estado estacionario.

Teniendo en cuenta estas consideraciones, hay un amplio campo de aplicaciones donde el control basado en eventos (EBC) tiene grandes ventajas respecto a las metodologías clásicas. La principal motivación de esta tesis es aplicar estas técnicas de control basado en eventos en los sistemas de control en red (NCS) (Bemporad et al. [18], Cervin and Henningsson [24], Mazo Jr and Tabuada [85], Tipsuwan and Chow [139]), concretamente en aplicaciones de control para robots móviles que usen redes de comunicación inalámbricas.

Otros campos en los que también estas metodologías EBC están teniendo una presencia notable en estos últimos años son: sistemas con sensores de baja resolución (Heemels et al. [55], De Bruin and van den Bosch [29]), control de satélites (auf der Heide et al. [15]) y aceleradores de partículas (Evans [42]). Todos ellos son ejemplos de control donde estas metodologías tienen grandes posibilidades.

Aparte de estos ejemplos de aplicación, el control basado en eventos puede aplicarse en todos aquellos escenarios de control donde es tolerable una desviación limitada del punto de referencia del sistema.

1.4. Objetivos

Los principales objetivos que se plantean en esta tesis son los siguientes:

1. Analizar y proponer arquitecturas de control basadas en eventos que mejor se adapten al problema de control de la navegación de robots móviles y que ofrezcan ventajas diferenciales frente a las soluciones basadas en control clásico.

2. Estudiar métodos de ajuste dinámico de umbrales de tal forma que los controladores obtenidos se adapten al entorno de trabajo de forma autónoma o con la menor intervención humana posible.

3. Investigar los efectos del ruido de medida en las arquitecturas de control propuestas, analizar sus efectos y plantear metodologías que permitan obtener un modelo de cara a ajustar los umbrales de forma más adecuada.

4. Estudiar y proponer nuevas metodologías de posicionamiento para las plataformas de robots sobre las que se apliquen las arquitecturas de control basadas en eventos desarrolladas en este trabajo.

5. Diseñar e implementar un laboratorio de experimentación que permita validar las ideas y aportaciones planteadas en este trabajo. Este entorno de experimentación debe contemplar la posibilidad de programar los algoritmos desarrollados en control basado en eventos planteados. Además, el sistema debe contemplar la implementación de soluciones equivalentes en control discreto que sirvan de marco de comparación para validar las soluciones propuestas.

1.5. Estructura de la Tesis

La tesis se ha estructurado en una serie de capítulos que tratan de cubrir de forma exhaustiva los objetivos descritos. En este sentido, se estructura de la siguiente forma:

- **Capítulo 1 - Introducción:** En este capítulo se hace una introducción de los sistemas de control basados en eventos cuya estructura básica se apoya en un esquema de control basado en Red (NCS). Se hace una extensa revisión bibliográfica y se presentan los diferentes avances en este campo de investigación. Seguidamente, se analizan de forma genérica las arquitecturas de este tipo de sistemas y las condiciones que deben cumplirse para garantizar su estabilidad. El capítulo termina presentando los objetivos de la tesis, su estructura y las contribuciones realizadas durante el proceso de investigación.

- **Capítulo 2 - Robots Móviles:** Se hace una revisión de los conceptos y modelos matemáticos que describen el comportamiento de los robots móviles. Se describen también los diferentes tipos de robots, profundizando en los robots diferenciales que serán sobre los que se apliquen las ideas y contribuciones desarrolladas en esta tesis. Posteriormente, se hace una revisión de los modelos cinemáticos, dinámicos y de las diferentes técnicas de posicionamiento que se utilizan en este tipo de plataformas. Se continua con una descripción de los sensores más típicos utilizados en este tipo de robots y sus características principales. El capítulo finaliza con una descripción de los objetivos del control en este tipo de robots y se analiza la estabilidad de estos sistemas desde un punto de vista de los algoritmos de navegación.

- **Capítulo 3 - Sistemas de Control basados en Eventos para Robots Móviles:** Se presentan las estructuras de control para robots móviles en entornos inalámbricos. En este entorno, se propone una arquitectura de control basada en eventos que permite implementar los algoritmos de navegación típicos en aplicaciones de robots móviles. Posteriormente, se analizan y se proponen los algoritmos de navegación sobre la arquitectura propuesta, donde se detallan los parámetros que los definen y la forma de sintonizar dichos algoritmos. El capítulo finaliza con un estudio sobre la eficiencia en el uso de los recursos de comunicación, computación y consumo energético que la estructura propuesta obtiene respecto a las soluciones clásicas basadas en control discreto.

- **Capítulo 4 - Mecanismos de Ajuste Dinámico de Umbrales:** Se analizan las diferentes metodologías y estructuras utilizadas para generar eventos en los sistemas de control EBC analizados. Se investigan los efectos que tiene el ruido en la generación de eventos y se analizan las ventajas de tener sistemas con umbrales dinámicos frente a sistemas clásicos con umbrales fijos. Además, se propone una arquitectura de control que estima de forma dinámica los umbrales en el sistema en función del ruido. A su vez, esta arquitectura también puede utilizarse en el caso de que se disponga del modelo de perturbación que afecta a las medidas del sistema. Finalmente, se analiza la respuesta de esta solución en los algoritmos de navegación planteados en este trabajo así como en sistemas más complejos como puede ser un UAV.

- **Capítulo 5 - Análisis y Modelado de Perturbaciones:** Se desarrolla la idea de fijar los umbrales en el sistema de control basado en eventos a partir del modelo de perturbación que afecta a las medidas de los sensores. Se analiza cómo fijar estos umbrales a partir del modelo de perturbación y conseguir de esta forma tener una mayor eficiencia en el consumo de recursos de comunicación, computación y energía. El capítulo termina con la definición de una metodología que permite obtener un modelo de ruido de los sensores a partir de utilizar patrones conocidos, utilizándose luego este modelo para fijar los umbrales en el sistema.

- **Capítulo 6 - Posicionamiento de Robots Móviles:** Se plantean nuevas ideas que permiten mejorar el posicionamiento de los robots móviles de bajo

coste, centrando la atención en los modelos utilizados en la plataforma de experimentación desarrollada en este trabajo. Dado que este tipo de robots obtienen su posicionamiento básicamente con técnicas odométricas, en este capítulo se propone un algoritmo de estimación de posicionamiento que combina la odometría con estimaciones de inclinación basadas en medidas inerciales. Se propone a su vez, un nuevo sistema de sincronización absoluta de la orientación a partir de las medidas del acelerómetro para mejorar las estimaciones de posicionamiento frente a las técnicas clásicas basadas en odometría.

- **Capítulo 7 - Resultados Experimentales:** En este capítulo se pretenden validar las distintas contribuciones de esta tesis de forma empírica. Para ello, se realizan diferentes experimentos utilizando el laboratorio de experimentación desarrollado para este trabajo. La idea general sobre la que se apoyan estos experimentos es comparar las soluciones desarrollas en control basado en eventos con implementaciones clásicas en control discreto que resuelvan el mismo problema de navegación. En dichos experimentos, se analizan los algoritmos de navegación planteados y la estabilidad de éstos. Se estudia también la eficiencia en el consumo de recursos y las mejoras obtenidas. Respecto al análisis de perturbaciones, se obtienen modelos empíricos del ruido que afecta a este tipo de plataformas utilizándose éstos posteriormente para fijar los umbrales en el sistema. Finalmente, se analizan de forma experimental la respuesta de los sistemas de posicionamiento planteados.

- **Capítulo 8 - Conclusiones y Líneas Futuras:** Se analizan las principales conclusiones obtenidas durante el periodo de investigación y se dan las pautas para seguir desarrollando las ideas y contribuciones obtenidas en este trabajo.

1.6. Contribuciones y Publicaciones

Las principales contribuciones de esta tesis doctoral se pueden resumir en:

- Diseño y parametrización de una arquitectura de control basada en eventos para aplicaciones de navegación en robots móviles.

- Definición de los algoritmos clásicos de navegación sobre la arquitectura propuesta. Se fijan los criterios para garantizar su estabilidad y los parámetros que gobiernan dichos algoritmos en el entorno de control basado en eventos desarrollado.

- Planteamiento de una metodología para medir la eficiencia en el consumo de recursos de las estructuras desarrolladas frente a soluciones clásicas equivalentes.

- Diseño y parametrización de un módulo para el cálculo dinámico de umbrales en el sistema de control basado en eventos propuesto. Se investiga su aplicación en los algoritmos de navegación para robots móviles y su extensión a otras plantas complejas.

- Desarrollo de una metodología para la estimación de los modelos de ruido en los sensores y su aplicación en el ajuste de umbrales a partir de dichos modelos.

- Propuesta de un algoritmo de posicionamiento para robots móviles que mejora las estimaciones de posición y orientación respecto a la técnicas clásicas basadas en odometría. Se incorpora además en dicho algoritmo nuevos mecanismos de sincronización absoluta.

Otras aportaciones realizadas como resultado de este trabajo de investigación han sido las Publicaciones, el Desarrollo de Aplicaciones, la Tutela de Proyectos, una Patente en proceso de registro y la Revisión de Trabajos Científicos.

Publicaciones

Las publicaciones realizadas durante el desarrollo de esta tesis han sido las siguientes:

Congresos

- Rafael Socas, Sebastián Dormido, and Raquel Dormido. Event-based controller for noisy environments. En *Complex Systems (WCCS), 2014 Second World Conference on,* págs. 280-285. IEEE, 2104.

- Rafael Socas, Sebastián Dormido, Raquel Dormido, and Ernesto Fabregas. 3d positioning algorithm for low cost mobile robots. En *Informatics in Control, Automation and Robotics (ICINCO), 2015 12th International Conference on*, tomo 2, págs. 5-14. IEEE, 2015.
- Rafael Socas, Sebastián Dormido, and Raquel Dormido. Event-based control strategy for the guidance of the aerosonde uav. En *Mobile Robots (ECMR), 2015 European Conference on*, págs. 1-6. IEEE, 2015.

Revistas

- Rafael Socas, Sebastián Dormido, Raquel Dormido, and Ernesto Fabregas. Event-based control strategy for mobile robots in wireless environments. *Sensors*, 15(12):30076-30092, 2015.
- Rafael Socas, Sebastián Dormido, and Raquel Dormido. Optimal threshold setting for event-based control strategies. *IEEE Access*, 2017

Libros

- Rafael Socas, Sebastián Dormido, Raquel Dormido, and Ernesto Fabregas. Improving the 3d positioning for low cost mobile robots. En *Informatics in Control, Automation and Robotics 12th International Conference, ICINCO 2015 Colmar, France, July 21-23, 2015 Revised Selected Papers*, págs. 97-114. Springer, 2016.

Desarrollo de Aplicaciones

Las aplicaciones más relevantes desarrolladas en este trabajo de investigación se pueden resumir en las siguientes:

- Librería gráfica de algoritmos de navegación para la plataforma de robots mOway en el entorno mOway World.
- Software de Control de Navegación para el dominio Discreto en C++.
- Software de Control de Navegación para el dominio de Eventos en C++.
- Plantillas de Análisis de los experimentos de Navegación en el entorno Tracker.
- Módulos software en Matlab para el Análisis de Resultados.

Tutela de Proyectos

Durante el periodo de investigación se ha realizado la tutela de dos proyectos fin de grado en el departamento de Ingeniería Electrónica y Automática de la Universidad de Las Palmas de Gran Canaria. Dicha tutela ha sido co-dirigida por el profesor Dr. Luis Gómez Déniz, Profesor Titular en dicha Universidad.

- Saúl Robaina, Luis Gómez, and Rafael Socas. Diseño de un sistema electrónico para el pesaje automático y estimación del centro de gravedad de aviones ligeros. En *Proyecto Fin de Grado*. Departamento de Electrónica y Automática. Universidad de Las Palmas de Gran Canaria, 2015.
- José Miguel Betancort, Luis Gómez, and Rafael Socas. Modelado y diseño de un sistema de control para la dinámica de vuelo de un quadrotor. En *Proyecto Fin de Grado*. Departamento de Electrónica y Automática. Universidad de Las Palmas de Gran Canaria, 2016.

Patentes

Fruto del proyecto *Diseño de un Sistema Electrónico para el Pesaje Automático y Estimación del Centro de Gravedad de Aviones Ligeros*, se ha comenzado el proceso de patentar las ideas allí desarrolladas. El pasado 08/03/2016 se dio registro en la Oficina Española de Patentes y Marcas la solicitud de modelo de utilidad **Sistema electrónico para estimación de carga y centrado de aeronaves**. En dicha patente el autor de esta tesis posee el $33,3\,\%$ de la propiedad intelectual.

Revisión de Trabajos Científicos

El autor de este trabajo ha sido invitado por la revista *International Journal of Advanced Robotics and Automation (IJARA)* a participar como revisor de sus publicaciones y a ser miembro activo del *Editorial Board*. Los trabajos revisados por el autor durante la elaboración de esta tesis y que se han publicados en dicha revista se detallan a continuación:

- A Pandey and DR Parhi. Multiple mobile robots navigation and obstacle avoidance using minimum rule based anfis network controller in the cluttered environment. En *Volume1-Issue1 - 2016*, págs. 1(1): 1–11. International Journal of Advanced Robotics and Automation, 2016.

- D Flippo and DP Miller. Advantages of anisotropic wheels in skid steer turns. En *Volume1-Issue1 - 2016*, págs. 1(1): 1–5. International Journal of Advanced Robotics and Automation, 2016.

- HW Ka, D Ding, and RA Cooper. Three dimensional computer vision-based alternative control method for assistive robotic manipulator. En *Volume1-Issue1 - 2016*, págs. 1(1): 1–6. International Journal of Advanced Robotics and Automation, 2016.

- X Zhang, W Xie, SV Hoa, and R Zeng. Design and analysis of collaborative automated fiber placement machine. En *Volume1-Issue1 - 2016*, págs. 1(1): 1–14. International Journal of Advanced Robotics and Automation, 2016.

Capítulo 2

Robots Móviles

En este capítulo se hace una revisión de los conceptos y modelos matemáticos que describen el comportamiento de los robots móviles. Se describen los diferentes tipos de robots, profundizando en los robots diferenciales que serán sobre los que se apliquen las ideas y contribuciones desarrolladas en esta tesis. Posteriormente, se hará una revisión de los modelos cinemáticos, dinámicos y de las diferentes técnicas de posicionamiento que se utilizan en este tipo de plataformas. Se continua con una descripción de los sensores más típicos utilizados en este tipo de robots y sus características principales. Finaliza el capítulo describiendo los objetivos del control en este tipo de robots y se analiza la estabilidad de estos sistemas desde el punto de vista de los algoritmos de navegación.

2.1. Conceptos Generales

Los robots móviles son un tipo de robots que pueden moverse de un sitio a otro de forma autónoma. A diferencia de la mayoría de los robots industriales que solo pueden moverse en un entorno de trabajo específico, los robots móviles tienen la característica especial de moverse libremente por entornos que no están definidos previamente para alcanzar los objetivos marcados. Esta capacidad de movimiento permite que se puedan desarrollar gran cantidad de aplicaciones en entornos estructurados y no estructurados.

Los robots móviles terrestres pueden clasificarse en robots móviles con ruedas (*Wheeled Mobile Robots* WMRs) que pueden incluir también un mecanismo de locomoción basados en cadenas y los robots móviles con patas (*Legged Mobile Robots* LMRs). También son considerados dentro de la categoría de robots móvi-

les los vehículos aéreos no tripulados (*Unmanned Aerial Vehicles* UAVs) y los vehículos autónomos acuáticos (*Autonomous Underwater Vehicles* AUVs). En la Figura 2.1 se presenta un esquema general de la clasificación de los robots móviles descritos en este apartado.

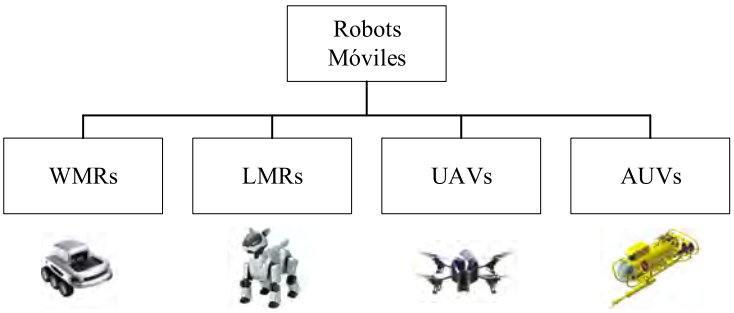

Figura 2.1: Clasificación de los robots móviles.

Este trabajo de investigación estará centrado en los sistemas de control que gobiernan los robots móviles terrestres con ruedas (WMRs). Este tipo de robots son los más populares y se usan en gran número de aplicaciones ya que tienen una complejidad mecánica y un consumo de energía reducida.

2.1.1. Robots Móviles con Ruedas (WMRs)

La maniobrabilidad de los WMRs depende de los motores y del tipo de ruedas que estos incluyen. En general para aplicaciones con movimientos en un plano éstos suelen tener tres grados de libertar (3DOF). Dependiendo del terreno donde vayan a trabajar la suspensión tiene un papel fundamental ya que tiene la misión de asegurar el contacto de las ruedas con el terreno. Como normal general, los principales problemas de diseño para este tipo de robots son: la tracción, la maniobrabilidad, la estabilidad y el control que está estrechamente ligado al tipo de configuración que disponga.

En los robots móviles con ruedas las configuraciones más comunes que se pueden encontrar en la práctica son: movimiento diferencial, triciclo, omnidireccional, movimiento síncrono, ackerman (modelo de automóviles actuales) o tipo oruga, ver Tzafestas [142] para más detalles.

2.1.2. Robots WMRs Diferenciales

Esta tesis se centra en aplicar algoritmos de control basados en eventos a los robots móviles diferenciales los cuales se caracterizan por tener un movimiento con tres grados de libertad (3DOF) sobre un plano. Este tipo de robots están caracterizados geometricamente por el radio r de sus ruedas y la distancia L entre éstas (Figura 2.2).

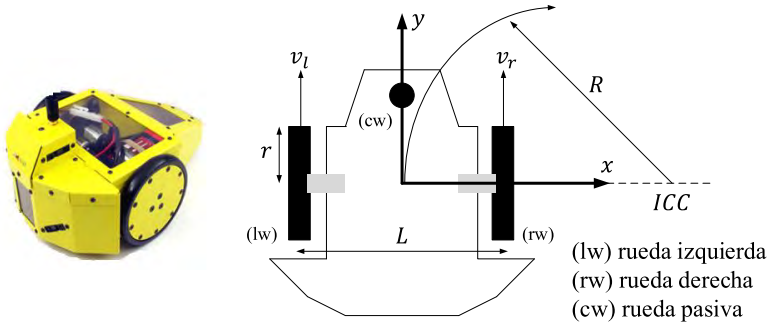

Figura 2.2: Geometría de los robots WMR diferenciales.

Los robots diferenciales contienen dos ruedas motorizadas de forma independiente, una en el lado izquierdo y otra en el lado derecho de la estructura. Disponen además de una rueda pasiva (cw *castor wheel*) para realizar los giros y mantener la estabilidad del robot. Este tipo de vehículos tienen una estructura mecánica muy simple ya que no necesitan rotar el eje de tracción y todos los movimientos los realiza combinando las velocidades de la ruedas motrices. Se resumen a continuación los diferentes movimientos que realizan:

1. Si ambas ruedas (lw y rw) rotan a la misma velocidad, el robot se mueve en línea recta hacia adelante o hacia atrás.

2. Cuando una rueda gira más rápida que la otra, el robot describe una trayectoria circular hacia el interior de la rueda más lenta.

3. Finalmente, si ambas ruedas giran a la misma velocidad pero en dirección opuesta, el robot gira alrededor de su punto medio situado en el centro de sus dos ruedas.

Una vez analizadas las trayectorias que pueden describir este tipo de robots, se puede definir el centro instantáneo de curvatura (*Instantaneous Center of Curvature* ICC) como el centro del círculo cuyo radio R describe el robot en su trayectoria circular. Este radio depende de las velocidades lineales de ambas ruedas (v_l y v_r) según se describe en la siguiente ecuación.

$$R = \frac{L}{2}\frac{(v_l + v_r)}{(v_l - v_r)} \ , \ v_l \geq v_r \tag{2.1}$$

Según la expresión anterior, cuando $v_l = v_r$ (movimiento el línea recta), tenemos un radio $R = \infty$. En cambio, cuando $v_l = -v_r$ (movimiento rotatorio), el radio descrito por el robot tiene un valor de $R = 0$.

2.2. Modelos y Métodos de Localización para Robots Diferenciales

Una vez analizadas las características generales de los robots diferenciales, en las siguientes secciones se describirán en detalle los modelos que describen el comportamiento dinámico de este tipo de robots a la vez que se presentan las técnicas para su localización.

2.2.1. Modelo Cinemático

El análisis cinemático estudia la configuración del robot, su espacio de trabajo, la relación entre sus parámetros geométricos y las restricciones impuestas en su trayectoria. Por tanto, sus ecuaciones cinemáticas dependen de la estructura geométrica del robot. El análisis cinemático es fundamental para el estudio de la dinámica, la estabilidad y el control del robot (Angelo [4], Alexander and Maddocks [3], Phairoh and Williamson [111]).

Si se consideran los robots móviles diferenciales, cuyos parámetros geométricos y cinemáticos se representan en la Figura 2.3, el vector de posición y su velocidad

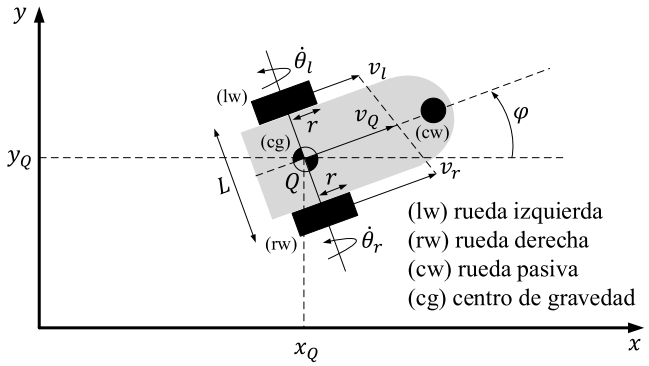

Figura 2.3: Parámetros geométricos y cinemáticos de los robots diferenciales.

pueden representarse según las Ecuaciones 2.2 y 2.3.

$$\mathbf{p} = \begin{pmatrix} x_Q \\ y_Q \\ \varphi \end{pmatrix} \tag{2.2}$$

$$\dot{\mathbf{p}} = \begin{pmatrix} \dot{x}_Q \\ \dot{y}_Q \\ \dot{\varphi} \end{pmatrix} \tag{2.3}$$

En este esquema se definen las posiciones angulares θ_l y θ_r tanto para la rueda izquierda como para la rueda derecha respectivamente, así como sus velocidades angulares $\dot{\theta}_l$ y $\dot{\theta}_r$. Por otro lado, para poder definir las ecuaciones de la dinámica de este robot, se tienen en cuenta las siguientes suposiciones:

1. Las ruedas giran sobre la superficie sin efectos de deslizamiento.
2. El eje de giro del robot es perpendicular al plano Oxy.
3. El punto Q coincide con el centro de gravedad (cg) del robot.

Definiendo v_l la velocidad lineal de la rueda izquierda, v_r la de la rueda derecha y v_q la velocidad del punto medio Q entre las ruedas, se puede relacionar dichas velocidades según la Ecuación 2.4.

$$v_r = v_Q + \frac{L}{2}\dot{\varphi} \quad , \quad v_l = v_Q - \frac{L}{2}\dot{\varphi} \tag{2.4}$$

Si se opera con las expresiones de la Ecuación 2.4, puede obtenerse la expresión para la velocidad lineal y angular del robot según la siguiente expresión.

$$v_Q = \frac{1}{2}(v_r + v_l) \quad , \quad \dot{\varphi} = \frac{(v_r - v_l)}{L} \tag{2.5}$$

Puesto que se ha considerado que no existe efectos de deslizamiento, se satisface que $v_r = r\dot{\theta}_r$ y $v_l = r\dot{\theta}_l$, obteniéndose la velocidad en el eje x como $\dot{x}_Q = v_Q cos(\varphi)$ y para el eje y según la expresión $\dot{y}_Q = v_Q sin(\varphi)$, resultando así las Ecuaciones 2.6 - 2.8.

$$\dot{x}_Q = \frac{r}{2}(\dot{\theta}_r cos(\varphi) + \dot{\theta}_l cos(\varphi)) \tag{2.6}$$

$$\dot{y}_Q = \frac{r}{2}(\dot{\theta}_r sin(\varphi) + \dot{\theta}_l sin(\varphi)) \tag{2.7}$$

$$\dot{\varphi} = \frac{r}{L}(\dot{\theta}_r - \dot{\theta}_l) \tag{2.8}$$

Si se representan estas ecuaciones de forma matricial se obtiene la expresión 2.9 que se corresponde con **el modelo cinemático diferencial directo,**

$$\dot{\mathbf{p}} = \begin{pmatrix} (\frac{r}{2})cos(\varphi) \\ (\frac{r}{2})sin(\varphi) \\ \frac{r}{L} \end{pmatrix} \dot{\theta}_r + \begin{pmatrix} (\frac{r}{2})cos(\varphi) \\ (\frac{r}{2})sin(\varphi) \\ \frac{-r}{L} \end{pmatrix} \dot{\theta}_l \tag{2.9}$$

cuya forma simplificada es,

$$\dot{\mathbf{p}} = \mathbf{J}\dot{\mathbf{q}} \tag{2.10}$$

donde $\dot{\mathbf{p}} = \begin{pmatrix} \dot{x}_Q \\ \dot{y}_Q \\ \dot{\varphi} \end{pmatrix}$, $\dot{\mathbf{q}} = \begin{pmatrix} \dot{\theta}_r \\ \dot{\theta}_l \end{pmatrix}$ y $\mathbf{J}$ es el Jacobiano del robot diferencial dado por:

$$\mathbf{J} = \begin{pmatrix} \frac{r}{2}cos(\varphi) & \frac{r}{2}cos(\varphi) \\ \frac{r}{2}sin(\varphi) & \frac{r}{2}sin(\varphi) \\ \frac{r}{L} & \frac{-r}{L} \end{pmatrix} \tag{2.11}$$

La matriz Jacobiana no es invertible (tiene tres filas y dos columnas), por tanto, para obtener $\dot{\mathbf{q}}$ a partir de la Ecuación 2.10 se tiene que utilizar la expresión $\dot{\mathbf{q}} = \mathbf{J}^{\dagger}\dot{\mathbf{p}}$ donde $\mathbf{J}^{\dagger}$ es la matriz pseudoinversa de $\mathbf{J}$ definida de la forma $\mathbf{J}^{\dagger} = (\mathbf{J}^T\mathbf{J})^{-1}\mathbf{J}^T$. Sin embargo, $\mathbf{J}^{\dagger}$ puede calcularse considerando la Ecuación 2.4 y la relación $v_Q = \dot{x}_Q cos(\varphi) + \dot{y}_Q sin(\varphi)$, obteniéndose la expresión 2.12.

$$\begin{aligned} r\dot{\theta}_r &= \dot{x}_Q cos(\varphi) + \dot{y}_Q sin(\varphi) + \tfrac{L}{2}\dot{\varphi} \\ r\dot{\theta}_l &= \dot{x}_Q cos(\varphi) + \dot{y}_Q sin(\varphi) - \tfrac{L}{2}\dot{\varphi} \end{aligned} \tag{2.12}$$

Seguidamente, si se escriben estas ecuaciones de forma matricial, se obtiene la expresión 2.13 que se corresponde con el **modelo cinemático diferencial inverso,**

$$\begin{pmatrix} \dot{\theta}_r \\ \dot{\theta}_l \end{pmatrix} = \frac{1}{r} \begin{pmatrix} cos(\varphi) & sin(\varphi) & \frac{L}{2} \\ cos(\varphi) & sin(\varphi) & -\frac{L}{2} \end{pmatrix} \begin{pmatrix} \dot{x}_Q \\ \dot{y}_Q \\ \dot{\varphi} \end{pmatrix} \tag{2.13}$$

que de forma compacta se expresa según la Ecuación 2.14

$$\dot{\mathbf{q}} = \mathbf{J}^{\dagger}\dot{\mathbf{p}} \tag{2.14}$$

donde $\mathbf{J}^{\dagger}$ está dado por:

$$\mathbf{J}^{\dagger} = \frac{1}{r}\begin{pmatrix} cos(\varphi) & sin(\varphi) & \frac{L}{2} \\ cos(\varphi) & sin(\varphi) & -\frac{L}{2} \end{pmatrix} \tag{2.15}$$

Finalmente, el radio de curvatura instantáneo R se obtiene mediante:

$$R = \frac{v_Q}{\dot{\varphi}} = \frac{L}{2}\left(\frac{v_r + v_l}{v_r - v_l}\right) \tag{2.16}$$

2.2.2. Modelo Dinámico

Una vez analizado el problema del modelado cinemático de los robots diferenciales, en esta sección se tratará su análisis dinámico. Los modelos dinámicos de estos robots se obtienen considerando los momentos de inercia, la elasticidad y la fricción presente en los mecanismos de los robots. Dichos modelos suelen ser de gran utilidad en los problemas de control y análisis sobre todo cuando existen deslizamientos laterales o longitudinales en el movimiento de sus ruedas (Sidek and Sarkar [125], Williams et al. [149], Stonier et al. [136]).

Las metodologías que se aplican para obtener estos modelos suelen ser el método de Newton-Euler o el método de Lagrange. En este caso, se utilizará el primer método para obtener el modelo dinámico del robot móvil.

Considerando nuevamente la Figura 2.3 donde el punto medio Q coincide con el centro de gravedad del robot, el movimiento de traslación queda descrito mediante la Ecuación 2.17 y el de rotación mediante la Ecuación 2.18.

$$m\dot{\mathbf{v}} = \mathbf{F} \tag{2.17}$$

$$I\dot{\boldsymbol{\omega}} = \mathbf{N} \tag{2.18}$$

En este caso $\mathbf{F}$ es la fuerza total aplicada al centro de gravedad (cg), $\mathbf{N}$ es el momento angular total respecto al cg, m es la masa del robot e I el momento de inercia.Considerando la fuerza que genera el motor derecho ($\mathbf{F_r}$) y el izquierdo ($\mathbf{F_l}$), así como sus respectivos momentos ($\boldsymbol{\tau_r}$ y $\boldsymbol{\tau_l}$) se pueden escribir las relaciones entre las fuerzas y los momentos que actúan sobre el robot tal como se muestra

en la Ecuación 2.19.

$$\mathbf{F} = \mathbf{F_r} + \mathbf{F_l}, \quad \boldsymbol{\tau_r} = \mathbf{r} \cdot \mathbf{F_r}, \quad \boldsymbol{\tau_l} = \mathbf{r} \cdot \mathbf{F_l} \tag{2.19}$$

A partir de esta ecuación y teniendo en cuenta la Figura 2.3, se pueden escribir las Ecuaciones 2.20 y 2.21 que relacionan las fuerzas y los momentos con los parámetros geométricos del robot.

$$\mathbf{F} = \frac{1}{\mathbf{r}}(\boldsymbol{\tau_r} + \boldsymbol{\tau_l}) \tag{2.20}$$

$$\mathbf{N} = (\mathbf{F_r} - \mathbf{F_l})L = \frac{L}{\mathbf{r}}(\boldsymbol{\tau_r} - \boldsymbol{\tau_l}) \tag{2.21}$$

Finalmente, combinando la Ecuación 2.20 con la 2.17 y la Ecuación 2.21 con la 2.18, obtenemos el **modelo dinámico del robot diferencial**, que queda descrito por las Ecuaciones 2.22 y 2.23.

$$\dot{\mathbf{v}} = \frac{1}{m\mathbf{r}}(\boldsymbol{\tau_r} + \boldsymbol{\tau_l}) \tag{2.22}$$

$$\dot{\boldsymbol{\omega}} = \frac{L}{I\mathbf{r}}(\boldsymbol{\tau_r} - \boldsymbol{\tau_l}) \tag{2.23}$$

2.2.3. Modelo de Movimiento

Considérese un sistema de referencia inercial como (x, y) y un sistema de referencia en el centro de gravedad (cg) del robot como (x_R, y_R) (Figura 2.4). Si se toman las velocidades del robot en coordenadas cartesianas respecto al

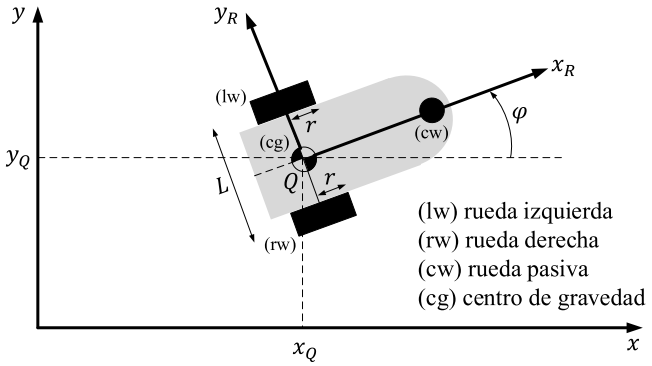

Figura 2.4: Sistema de referencia inercial y sistema de referencia del robot.

sistema de referencia inercial como $\dot{\mathbf{p}}$ (Ecuación 2.3) y las velocidades medidas

en el sistema de referencia del robot según la Ecuación 2.24,

$$\dot{\mathbf{p}}_\mathbf{r} = \begin{pmatrix} \dot{x}_r \\ \dot{y}_r \\ \dot{\varphi}_r \end{pmatrix} \tag{2.24}$$

se pueden relacionar las medidas en ambos sistemas de referencia según la expresión,

$$\dot{\mathbf{p}}_\mathbf{r} = \mathbf{R}(\varphi)\dot{\mathbf{p}} \tag{2.25}$$

donde $\mathbf{R}(\varphi)$ es la matriz de transformación definida por:

$$\mathbf{R}(\varphi) = \begin{pmatrix} cos(\varphi) & sin(\varphi) & 0 \\ -sin(\varphi) & cos(\varphi) & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.26}$$

Cada una de la ruedas contribuye al movimiento del robot y al mismo tiempo imponen restricciones al movimiento de éste. Dado que las ruedas del robot no pueden deslizarse, esto impone la restricción no-holonómica que se expresa según:

$$\dot{x}_Q sin(\varphi) - \dot{y}_Q cos(\varphi) = 0 \tag{2.27}$$

Por otro lado, la distancia recorrida por cada una de las ruedas no es suficiente para estimar la posición final del robot. Es necesario conocer como evoluciona el movimiento a lo largo del tiempo tal como se ilustra en la Figura 2.5 ($S_{1L} = S_{2L}$, $S_{1R} = S_{2R}$ y $S_1 = S_2$, pero $x_1 \neq x_2$ e $y_1 \neq y_2$.).

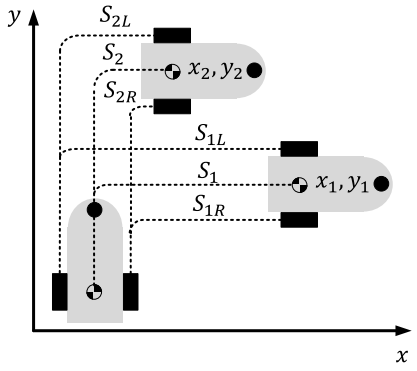

Figura 2.5: Ambigüedad del posicionamiento.

Considérese el robot diferencial en una posición arbitraria A $(x_A, y_A, \varphi)^T$ a una distancia ρ de la posición objetivo B $(x_B, y_B, \beta)^T$ definida con respecto al sistema

de referencia inercial, Figura 2.6. Los parámetros que gobiernan el movimiento

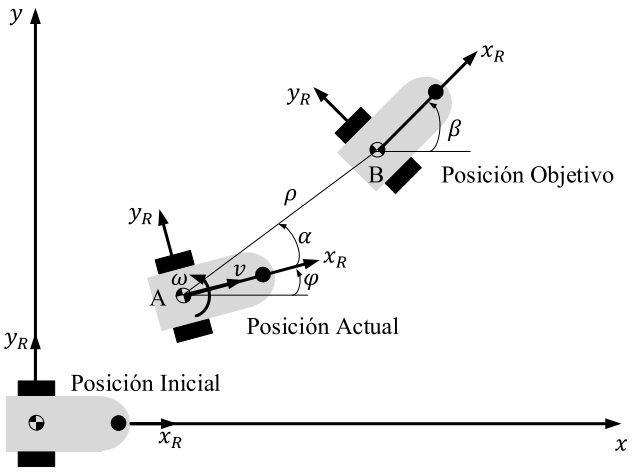

Figura 2.6: Definición del posicionamiento del robot móvil diferencial.

del robot son las velocidades angulares $\dot{\theta}_l$ y $\dot{\theta}_r$. Estas velocidades se relacionan con la velocidad lineal v y angular ω a la que se desplaza el cg del robot mediante las expresiones $v = \frac{r}{2}(\dot{\theta}_r + \dot{\theta}_l)$ y $\omega = \frac{r}{L}(\dot{\theta}_r - \dot{\theta}_l)$. Con esto, pueden escribirse las ecuaciones de posición instantánea del robot en coordenadas cartesianas respecto al sistema de referencia inercial según la expresión 2.28.

$$\begin{aligned} \dot{x} &= vcos(\varphi) \\ \dot{y} &= vsin(\varphi) \\ \dot{\varphi} &= \omega \end{aligned} \tag{2.28}$$

De la misma forma, utilizando las coordenadas polares y considerando la distancia ρ (para $\rho > 0$), las ecuaciones dinámicas del movimiento pueden escribirse de la forma:

$$\begin{aligned} \dot{\rho} &= -vcos(\beta - \varphi) = -vcos(\alpha) \\ \dot{\beta} &= v\frac{sin(\alpha)}{\rho} \\ \dot{\varphi} &= \omega \end{aligned} \tag{2.29}$$

Las ecuaciones dinámicas así descritas en coordenadas polares son la base a la hora de establecer las estrategias de control para alcanzar un punto objetivo en este tipo de robots.

2.2.4. Localización basada en Odometría

La localización es uno de los aspectos fundamentales que deben ser tratados en la ingeniería de robots móviles (Siegwart et al. [126], Burgard et al. [22], Chenavier and Crowley [26]). La cuestión fundamental es conocer donde se encuentra el robot y para ello, debe conocerse en todo momento la localización y orientación de éste. Una vez resuelto el problema del posicionamiento, las diferentes técnicas de control podrán guiar al robot desde su posición actual hasta su destino.

Existen muchas técnicas de localización de robots móviles, siendo una de las más comunes las basadas en odometría. Estas metodologías se basan en utilizar codificadores incrementales que miden la rotación de las ruedas y que mediante una integración del modelo cinemático del robot entre los períodos de muestreo $[t_n, t_{n+1}]$ estiman su posición.

Por tanto, para un robot diferencial, la posición puede ser estimada partiendo de una posición conocida mediante la integración de su movimiento (sumando los incrementos de distancia recorrida). De esta forma, la posición del robot en el instante t_n puede obtenerse mediante la Ecuación 2.30.

$$\begin{pmatrix} x_n \\ y_n \\ \varphi_n \end{pmatrix} = \begin{pmatrix} x_{n-1} \\ y_{n-1} \\ \varphi_{n-1} \end{pmatrix} + \begin{pmatrix} cos(\varphi_{n-1}) & 0 \\ sin(\varphi_{n-1}) & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \Delta S \\ \Delta\varphi \end{pmatrix} \tag{2.30}$$

donde $\Delta S = \frac{r}{2}(\Delta\eta_r + \Delta\eta_l)$ y $\Delta\varphi = \frac{r}{L}(\Delta\eta_r - \Delta\eta_l)$, siendo $\Delta\eta_r$ y $\Delta\eta_l$ el número de rotaciones de cada una de las ruedas medido en el intervalo de muestro T_s.

Como complemento a estas metodologías de localización, se puede consultar los trabajos Socas et al. [130] y Socas et al. [132] donde el autor de esta tesis presenta nuevas ideas para mejorar la localización basada en odometría complementada con técnicas inerciales. En estos trabajos se presentan unas metodologías de localización que aprovechan las medidas del acelerómetro del robot para hacer una estimación 3D de la posición. Al mismo tiempo, y basado en la estimación de la inclinación, se propone un sistema de sincronización para la orientación del robot. Con este nuevo sistema se consigue mejorar de forma considerable las estimaciones de posicionamiento respecto a utilizar técnicas clásicas basadas en odometría.

2.3. Sensores

Hay una amplia variedad de sensores que ayudan a los robots a realizar sus tareas y a reconocer el entorno que los rodea. En esta sección, se hará una descripción somera de los sensores más relevantes dentro del campo de los robots móviles, campo de interés de esta tesis.

Los sensores en robótica se pueden clasificar en: 1) sensores analógicos y 2) sensores digitales. Los primeros proporcionan una salida analógica la cual requiere una conversión analógica-digital (A/D). En cambio, los sensores digitales suelen tener más precisión y suelen tener diferentes tipos de salidas como pueden ser de tipo serie síncrona o paralelo con una codificación que está entre 8 y 16 bits, incluso superiores si son de muy alta precisión. En ambos casos, es deseable que los sensores para este tipo de robots tengan como características principales una alta resolución, un rango de operación elevado, una respuesta rápida, un método de calibración sencillo, alta fiabilidad y bajo coste.

Desde el punto de vista de los sistemas de sensores utilizados para el posicionamiento de robots, éstos pueden clasificarse en: 1) Mecánicos, 2) Acústicos, 3) Electromagnéticos, 4) Magnéticos y 5) Ópticos.

Finalmente, si se considera una clasificación considerando la perspectiva de los robots móviles, éstos pueden clasificarse como: 1) sensores internos y 2) sensores externos. Los sensores internos son aquellos que miden la velocidad de los motores, la carga de las ruedas, los ángulos de los brazos, el voltaje de la batería, etc. En cambio, los externos miden las distancias a los objetos, la amplitud del sonido, la intensidad de la iluminación, etc.

2.3.1. Características de los Sensores

Los sensores utilizados en robótica poseen una amplia variedad de características. Estas pueden variar dependiendo del entorno de trabajo, así que, tendrán prestaciones diferentes según sean entornos controlados (interiores, laboratorios, etc.) o entornos no controlados (exteriores, entorno del mundo real). Las características principales de dichos dispositivos suelen ser:

1. *Rango dinámico.* Define el intervalo entre el valor de entrada mínimo y el máximo con el que el sensor puede trabajar sin problemas de saturación.

Normalmente este rango se expresa en dB según la expresión 2.31.

$$\text{Rango} = 20 \log \left(\frac{\text{entrada máxima}}{\text{entrada mínima}} \right) (dB) \tag{2.31}$$

2. *Resolución*. Contempla la mínima variación en la medida que el sensor es capaz de medir.

3. *Linealidad*. Los sensores tienen un comportamiento lineal cuando cumplen el principio de superposición que engloba las propiedades de proporcionalidad y aditividad. Si un sensor tiene como salida y_1 para la entrada x_1 e y_2 para la entrada x_2, el principio de superposición establece que para la entrada $\alpha x_1 + \beta x_2$ el sensor tendrá una salida igual a $\alpha y_1 + \beta y_2$.

4. *Ancho de banda*. El ancho de banda establece el rango de frecuencias que puede tener la magnitud a medir y que el sensor puede detectar.

5. *Sensibilidad*. Es la razón de cambio de la salida frente a cambios en la entrada.

6. *Precisión*. Mide el grado en que la medida obtenida por el sensor coincide con la real. Esta precisión puede medirse de forma genérica según la expresión:

$$\text{Precisión} = 1 - \frac{|e|}{x} \tag{2.32}$$

donde e es el error definido como $e = y - x$, siendo y la medida del sensor y x el valor real. Cuando se trata de errores aleatorios, ésta puede calcularse mediante,

$$\text{Precisión} = \frac{\text{Rango}}{\sigma} \tag{2.33}$$

donde σ es la desviación estándar del error.

2.3.2. Posición y Velocidad

Los sensores de posición son utilizados para determinar cuanto se han movido los ejes motrices del robot (de forma lineal o rotacional). De forma similar, los sensores de velocidad miden la velocidad lineal/angular de estos ejes.

Dentro de los sensores de posición se pueden encontrar los potenciómetros, los *resolvers* y los codificadores, siendo estos últimos los más frecuentes.

Los codificadores de posición se clasifican en incrementales y absolutos. Estos sensores se suelen fabricar mediante dos técnicas diferentes: los sensores de efecto Hall (sensores magnéticos) y los ópticos que disponen un disco con segmentos blancos y negros junto con un LED y un fotodiodo (Figura 2.7 a)).

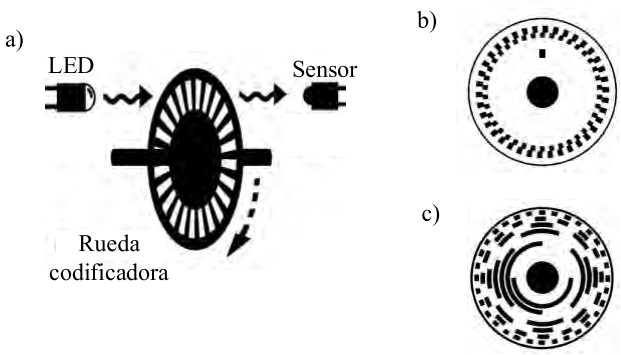

Figura 2.7: Codificador de posición óptico.

El fotodiodo detecta la luz cuando existe un segmento blanco y deja de recibirla cuando coincide con un segmento negro, de esta forma va recibiendo los pulsos a medida que avanza el robot a consecuencia de que va girando el disco. Un codificador incremental posee una segunda pista que actúa como índice e indica que se ha producido una revolución completa (Figura 2.7 b)). Finalmente, en el codificador absoluto el disco tiene codificada la palabra (byte o bytes) que indica en que posición se encuentra el disco en cada momento (Figura 2.7 c)).

Para los medidores de velocidad suelen utilizarse tacómetros, aunque es práctica habitual utilizar también integración numérica de la señal de los medidores de posición. En robots móviles y en robótica en general, los medidores de velocidad más ampliamente usados son los tacómetros de corriente continua. Éstos generan un voltaje de salida que es proporcional a la velocidad del motor conectado a este tipo de sensor.

2.3.3. Distancia

Dentro de los sensores utilizados para medir distancias en el campo de robótica podemos encontrar: 1) Sonar, 2) Láser y 3) Infrarrojos.

Dado que los robots utilizados en los experimentos de este trabajo de investigación, utilizan sensores de infrarrojos para medir distancias, se describirán someramente solo éstos últimos.

La luz infrarroja puede ser generada mediante un LED, esta señal tiene una longitud de onda λ que oscila entre los 820 y 880 nm. La mayoría de las superficies tienen una rugosidad superior a esta longitud de onda, por tanto, la luz infrarroja que incide sobre estas es reflejada de forma isotrópica. La componente de este rayo reflejado, que coincide en la dirección de la apertura del sensor, puede utilizarse para medir la distancia D a los objetos que están enfrente del sensor (Figura 2.8 a)).

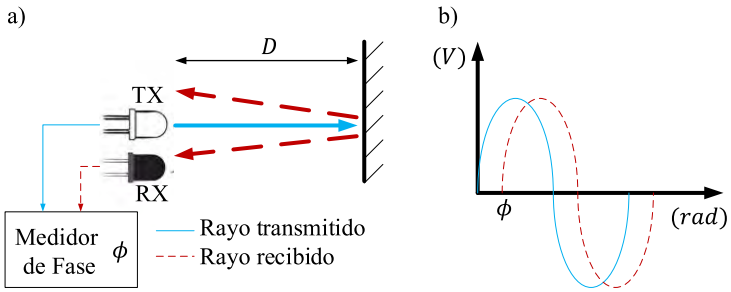

Figura 2.8: Sensor de distancia infrarrojo.

El sensor envía una luz infrarroja a una frecuencia f (cuya longitud de onda es λ), luego el medidor de fase calcula la diferencia de fases ϕ entre el rayo transmitido y el recibido (véase Figura 2.8 b)). Finalmente, una vez conocida esta diferencia de fase, se determina la distancia al objeto según la siguiente ecuación:

$$D = \left(\frac{\lambda}{4\pi}\right)\phi \tag{2.34}$$

2.3.4. Aceleración

Para medir las aceleraciones en aplicaciones de robótica, se han desarrollado una serie de sensores inerciales denominados acelerómetros que son capaces de medir la aceleración (gravitatoria, centrípeta, lineal, etc.) que afecta a un robot sobre un sistema de referencia dado. Existen varios tipos de acelerómetros entre los que se encuentran los mecánicos, capacitivos, pizoeléctricos, pizoresistivos, etc. La plataforma de robots móviles utilizada en los experimentos de este trabajo utiliza acelerómetros capacitivos por lo que se hará solo una breve reseña a este tipo de sensores.

Los acelerómetros capacitivos modifican la posición relativa de las placas de un microcondensador cuando éste está sometido a una aceleración. Por tanto, el movimiento paralelo de una de las placas del condensador hace variar su capaci-

dad. Como consecuencia, los acelerómetros capacitivos basan su funcionamiento en la variación de la capacidad entre dos ó más conductores entre los que se encuentra un dieléctrico, en respuesta a la variación de la aceleración (véase Figura 2.9 a)).

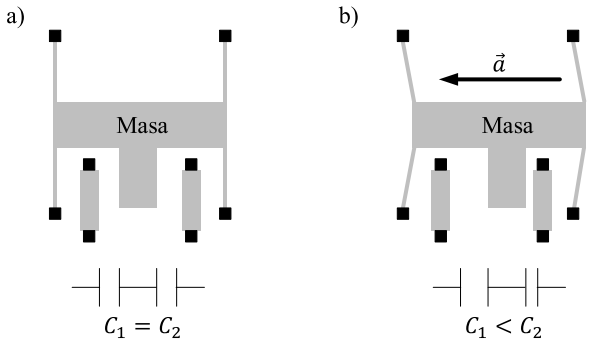

Figura 2.9: Acelerómetro capacitivo.

Estos sensores micromecanizados (MEMS) con forma de H tienen sus delgados y largos brazos sujetos al substrato, el resto de elementos se pueden mover libremente. Este dispositivo lo forman una serie de filamentos finos, con una masa central, donde cada uno actúa como una placa de un condensador variable de placas paralelo. La aceleración o desaceleración en el eje sensor, ejerce una fuerza sobre la masa central. Al moverse libremente, la masa desplaza las minúsculas placas del condensador, provocando un cambio de capacidad y detectando así los cambios de aceleración (Figura 2.9 b)).

2.3.5. Otros Sensores de la Plataforma de Experimentación

Los robots utilizados en los experimentos desarrollados en esta tesis disponen de otros sensores internos como son el medidor de batería, micrófono, medidor de temperatura, etc. Dado que estos sensores no han sido tratados de manera especial, en este documento no se hará ninguna descripción de los mismos.

2.4. Control y Estabilidad de los Robots WMR Diferenciales

Los algoritmos de control deben dirigir el robot desde la posición actual (x_c, y_c, φ) a la posición objetivo (x_g, y_g, β) tal como se muestra en la Figura 2.10.

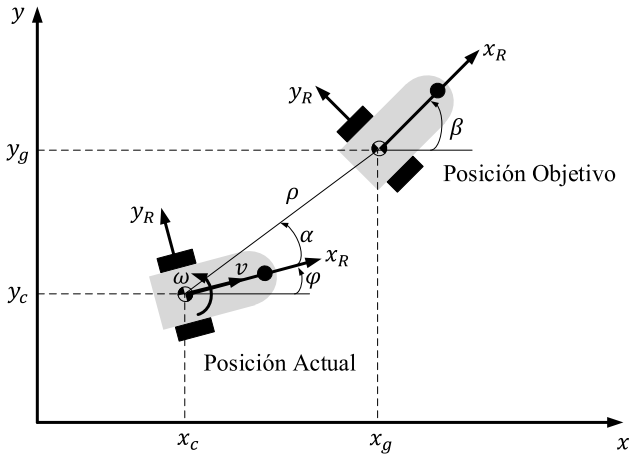

Figura 2.10: Definición del posicionamiento actual y objetivo.

El objetivo de la ley de control es encontrar $u = (v, \omega)$ que permita al robot alcanzar el punto objetivo en un intervalo de tiempo finito. Considerando el modelo dinámico del robot diferencial en coordenadas polares como se definió anteriormente, la ley de control debe depender del estado del robot $(v, \omega) = f(\rho, \alpha, \varphi)$ y debe garantizar que dicho estado tenderá asintóticamente a $(0, 0, \beta)$ en un tiempo finito (Malu and Majumdar [81]).

Para estudiar el comportamiento asintótico de este robot, se utilizará la teoría de estabilidad de Lyapunov que es una de las metodologías más comúnmente usadas en este campo. Se considera una función de Lyapunov cuadrática definida positiva de la forma:

$$V = V_1 + V_2 = \frac{1}{2}\rho^2 + \frac{1}{2}\alpha^2 \tag{2.35}$$

Donde los términos V_1 y V_2 representan una ponderación de las normas al cuadrado del vector error distancia ρ y del error de orientación α. Estos parámetros de error representan la diferencia entre la posición inicial y final del robot con respecto al sistema de referencia inercial. La derivada de la función de Lyapunov (2.35) puede expresarse como:

$$\dot{V} = \dot{V}_1 + \dot{V}_2 = \dot{\rho}\rho + \alpha\dot{\alpha} \tag{2.36}$$

sustituyendo la ecuación cinemática 2.29 en la 2.36, se obtiene:

$$\dot{V} = \rho\,(-v cos(\alpha)) + \alpha(-\omega + \frac{v}{\rho} sin(\alpha)) \tag{2.37}$$

Si la velocidad lineal v se escribe de la forma:

$$v = K_\rho \rho cos(\alpha) \quad , \quad K_\rho > 0 \tag{2.38}$$

donde K_ρ pondera la velocidad lineal en función del error de orientación α, por tanto, $\dot{V}_1$ puede escribirse según la expresión:

$$\dot{V}_1 = \rho(-K_\rho \rho cos^2(\alpha)) = -K_\rho \rho^2 cos^2(\alpha) \leq 0 \tag{2.39}$$

siendo $\dot{V}_1$ una función definida negativa.

De forma similar, si la velocidad angular se define como:

$$\omega = K_\rho sin(\alpha)cos(\alpha) + K_\alpha \alpha \quad , \quad K_\alpha > 0 \tag{2.40}$$

donde K_α pondera en este caso la velocidad angular junto con K_ρ, entonces el término $\dot{V}_2$ queda descrito por la expresión:

$$\dot{V}_2 = \alpha \left(-K_\rho sin(\alpha)cos(\alpha) - K_\alpha \alpha + \frac{K_\rho \rho sin(\alpha)cos(\alpha)}{\rho} \right) = -K_\alpha \alpha^2 \leq 0 \tag{2.41}$$

siendo $\dot{V}_2$ también una función definida negativa.

Finalmente, considerando las expresiones 2.39 y 2.41, se puede reescribir la derivada de la función de Lyapunov $\dot{V}$ (2.37) según la expresión:

$$\dot{V} = \dot{V}_1 + \dot{V}_2 = -K_\rho \rho^2 cos^2(\alpha) - K_\alpha \alpha^2 \leq 0 \tag{2.42}$$

Como resultado, la función $\dot{V}$ dada por la expresión 2.42 es una función semi-definida negativa. Aplicando el lema de Barbalat se garantiza que $\dot{V}$ necesariamente converge a cero a lo largo del tiempo. Esto implica que el robot partiendo de la posición (x_c, y_c, φ) converge al destino (x_g, y_g, β) gracias a las leyes de control $v = K_\rho \rho cos(\alpha)$ y $\omega = K_\rho sin(\alpha)cos(\alpha) + K_\alpha \alpha$ con un movimiento suave y estable, lo que en otras palabras significa que el robot converge de forma asintótica al destino.

2.5. Conclusiones

En este capítulo se han estudiado con cierta profundidad los robots móviles. Estos sistemas serán las plataformas sobre las que se desarrollarán y analizarán

las ideas planteadas en esta tesis. Se han analizado tanto los elementos que los componen como sus modelos matemáticos. Se ha hecho una revisión de los sensores que contienen y cuáles son sus características principales. Por otro lado, se han sentado las bases de cuales son los criterios que deben cumplir los algoritmos de control que los gobiernan para garantizar su estabilidad.

Capítulo 3

Sistemas de Control basados en Eventos para Robots Móviles

En este capítulo se presentan las estructuras de control para robots móviles en entornos inalámbricos. En este ámbito, se propone una arquitectura de control basada en eventos que permite implementar los algoritmos de navegación típicos en aplicaciones de robots móviles. Posteriormente, se analizan y se proponen los algoritmos de navegación sobre la arquitectura propuesta, detallándose los parámetros que los definen y la forma de sintonizar dichos algoritmos. El capítulo finaliza con un estudio sobre la eficiencia en el uso de los recursos de comunicación, computación y consumo energético que la estructura propuesta obtiene respecto a las soluciones clásicas basadas en control discreto.

3.1. Introducción

Si se hace una revisión de la literatura sobre los sistemas de control aplicados a robots móviles, se puede encontrar un amplio abanico de estructuras y algoritmos que evolucionan día a día con nuevas aportaciones en este campo de investigación. Estos algoritmos pueden clasificarse según las siguientes categorías:

1. *Métodos de Control basados en Lyapunov* (Velagic et al. [145], Ramaswamy and Balakrishnan [113], Aicardi et al. [2]).

2. *Métodos basados en Sistemas Afines e Invariantes Múltiples* (Astolfi [10], Watanabe et al. [147], DeVon and Bretl [31]).

3. *Control Adaptativo y Robusto* (Marino and Tomei [82], Pourboghrat and Karlsson [112], Solea et al. [133]).

4. *Control basado en Lógica Difusa y Redes Neuronales* (Moustris and Tzafestas [97], Tzafestas et al. [143]).

5. *Control basado en Visión* (Geyer [48], Gong [49], Barreto and Araujo [16], Yang and Wang [152], Liang and Wang [74]).

Por otro lado, desde un punto de vista temporal, los sistemas de control periódicos o activados por tiempo han dominado casi de manera exclusiva la investigación en ingeniería de control. El control basado en eventos es una alternativa muy prometedora particularmente cuando se consideran sistemas con capacidades reducidas de computación y de comunicación. En un sistema de control basado en eventos es la ocurrencia de un evento, en lugar del paso del tiempo, la que decide cuándo se debe efectuar el muestreo. Basado en esta idea, en el presente capítulo se desarrollan nuevas estructuras de control basadas en eventos aplicadas a robots móviles. La hipótesis sobre la que se sustenta el desarrollo de estas estructuras es conseguir un control similar al tradicional (continuo o discreto) pero que haga un uso más eficiente de los recursos de comunicación, computacionales y de consumo de energía.

3.2. Arquitecturas de Control para Robots Móviles en Entornos Inalámbricos

De manera genérica los robots móviles operan en entornos inalámbricos debido a que es el mejor medio de comunicación que los dota de autonomía y facilidad de movimiento. Por otro lado, desde un punto de vista tecnológico y gracias al desarrollo de la tecnología digital, la gran mayoría de esta soluciones trabajan a día de hoy con sistema de control discreto. Es por tanto, que las aportaciones de este trabajo se centrarán en cambiar este paradigma y proponer soluciones para estos entornos desarrollas en control basado en eventos. Se persigue en estas nuevas arquitecturas conseguir una respuesta similar a las clásicas de tiempo discreto complementadas con las ventajas intrínsecas del control basado en eventos.

3.2.1. Control en Tiempo Discreto

En los entornos inalámbricos, la arquitectura típica para robots móviles son sistemas de control de tiempo discreto como el mostrado en la Figura 3.1

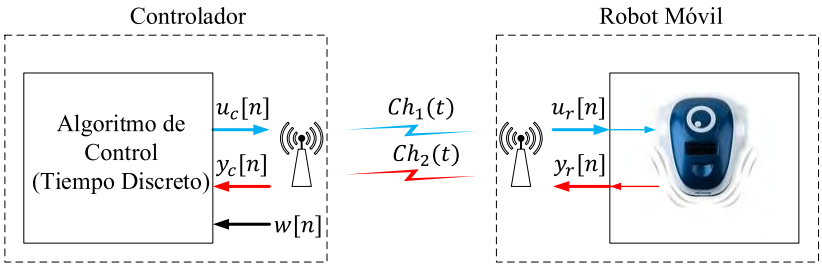

Figura 3.1: Sistema de control discreto inalámbrico.

Los elementos de este sistema trabajan en tiempo discreto; las señales de control ($u_c[n]$ y $u_r[n]$) y la señales de los sensores ($y_r[n]$ e $y_c[n]$) tienen un período de muestreo $1/f_s$, donde f_s es la frecuencia de muestreo del sistema. Los canales de radio frecuencia (RF) $Ch_1(t)$ y $Ch_2(t)$ pueden usar modulaciones analógicas o digitales para transportar la información entre los elementos. En ambos casos, este interfaz de RF trabaja en el dominio de tiempo continuo.

En este esquema, el controlador envía las señales de control $u_c[n]$ sobre el canal de comunicación $Ch_1(t)$; esta información es recibida en el robot $u_r[n]$ para controlar los actuadores. La diferencia entre las señales $u_c[n]$ y $u_r[n]$ es el ruido y las perturbaciones presentes en el canal de comunicaciones $Ch_1(t)$. La señales del sensor $y_r[n]$ son enviadas hacia el controlador sobre el otro canal de comunicación $Ch_2(t)$. Finalmente, el controlador recibe las señales del sensor $y_c[n]$ y calcula la información de control $u_c[n]$ considerando tanto la señal de referencia $w[n]$ como la información procedente de los sensores. Al igual que el canal $Ch_1(t)$, el ruido y las perturbaciones en el canal $Ch_2(t)$ hace que las señales transmitidas $y_r[n]$ sean diferentes a las señales recibidas $y_c[n]$. El sistema intercambia información entre el controlador y el robot móvil cada $1/f_s$ segundos, donde f_s está expresado en Hz.

En esta estructura, los canales de comunicación $Ch_1(t)$ y $Ch_2(t)$ están ocupados cada $1/f_s$, esto se debe a que el sistema de forma periódica intercambia información entre el controlador y el robot. Cuando el robot se encuentra en estado estacionario, no debería ser necesario intercambiar información entre estos dos elementos, pero tanto los canales de comunicación como el resto de elementos del sistema están ocupados cada periodo de tiempo $1/f_s$. Por otro lado, cuan-

do el canal de comunicación para conectar los diferentes elementos del sistema está implementado en tecnología inalámbrica, es deseable utilizar los canales de comunicación solo para enviar información crítica ya que este tipo de sistema de comunicación tiene los recursos muy limitados. Estos canales suelen tener un reducido ancho de banda y el número de elementos que pueden transmitir información al mismo tiempo suele ser reducido. Por tanto, desde un punto de vista de recursos tanto inalámbricos, computacionales como energéticos, se deberían estudiar formas alternativas de implementar las estrategias de control con un uso más eficiente de este tipo de recursos.

3.2.2. Nueva Arquitectura de Control basada en Eventos.

Como se ha mencionado anteriormente, si el sistema se encuentra en estado estacionario, las estrategias de control basadas en eventos tienen dos ventajas principales:

1. Los recursos de transmisión pueden quedar libres para transmitir otras informaciones diferentes a las del control.

2. El controlador no necesita calcular nuevas señales de control hasta que el robot abandone el estado estacionario.

Teniendo en cuenta estas consideraciones, en el presente trabajo se sientan las bases para desarrollar un control basado en eventos de aplicación en robots móviles. Apoyado en algunas de las ideas desarrolladas en Lunze and Lehmann [79], se plantea una arquitectura de control para robots móviles basada en eventos para entornos inalámbricos como la mostrada en la Figura 3.2.

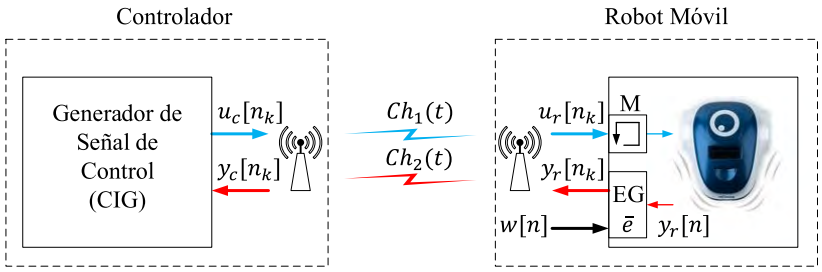

Figura 3.2: Sistema de control basado en eventos inalámbrico propuesto.

En el robot, un generador de eventos (EG) compara la señales del sensor $y_r[n]$ con la señal de referencia $w[n]$. Si la diferencia entre ambas señales supera el

event threshold $\bar{e}$, se genera un evento k y la señal $y_r[n_k]$ se envía al controlador a través del interfaz radio. Se pueden usar diferentes metodologías para generar eventos, en Dormido et al. [36] se realiza una amplia revisión de estas técnicas. En sentido general, el *event threshold* se define como un valor constante que debe ser seleccionado de forma muy cuidadosa porque impacta de forma directa en las prestaciones y estabilidad del sistema de control. En otras ocasiones, este puede ser definido como función del ruido del sistema o como función de otras variables para tener una mayor precisión a la hora de generar los eventos, véase Dormido et al. [36] y Socas et al. [127] para más detalles. La señal $y_r[n_k]$ llega al controlador a través del canal de comunicaciones $Ch_2(t)$. En este caso, el canal está ocupado solamente cuando el EG genera eventos. Cuando el robot está en un estado estacionario, las señales $w[n]$ e $y_r[n]$ son muy parecidas y no se genera ningún evento. En este caso, tanto el canal de comunicación $Ch_1(t)$ como $Ch_2(t)$ están libres. Cada vez que se genera un evento y la señal $y_r[n_k]$ se envía al controlador, el robot recibe la señales de control $u_r[n_k]$ y estas señales son almacenadas en la memoria M. Por tanto, siempre que se genera un evento k, la memoria se actualiza con estos nuevos valores, mientras que en los periodos entre eventos, las señales almacenadas en la memoria son usadas para actuar sobre el robot.

Por otro lado, siempre que se genera un evento en el sistema, el generador de señal de control (CIG) recibe la señal $y_c[n_k]$ y se calcula la señal de control $u_c[n_k]$, seguidamente, esta información $u_c[n_k]$ se envía al robot a través del canal de comunicación $Ch_1(t)$. Los canales de comunicación $Ch_1(t)$ y $Ch_2(t)$ están ocupados solo cuando se generan eventos, en otros periodos de tiempo, estos medios de comunicación están libres. Al igual que en la arquitectura de tiempo discreto, las diferencias entre las señales de los sensores ($y_r[n_k]$ y $y_c[n_k]$) y las diferencias entre las señales de control ($u_c[n_k]$ y $u_r[n_k]$) son debidas al ruido y las perturbaciones presentes en los canales de comunicación $Ch_1(t)$ y $Ch_2(t)$.

Si la arquitectura propuesta se compara con la estrategia de tiempo discreto, ésta tiene dos ventajas fundamentales:

1. Los canales de comunicación están ocupados solamente cuando en el sistema se generan eventos.

2. El controlador no necesita calcular señales de control mientras el robot está en estado estacionario.

En esta arquitectura basada en eventos, la señal de referencia $w[n]$ es proce-

sada en el generador de eventos (EG). Por esta razón, el canal de comunicaciones $Ch_1(t)$ se usa para enviar esta información. De forma general, esta información de referencia suele ser constante o sus cambios suelen ser de baja frecuencia. Por esta razón, el uso de los recursos radio para enviar la señal de referencia tiene poco impacto.

Las ideas aquí planteadas suponen de forma general que el sistema trabaja de forma estricta en el dominio de eventos. Esto quiere decir que, un evento se puede generar en cualquier instante de tiempo y que no está condicionado a un reloj que marque los intervalos de ejecución. En este caso, para evitar los fenómenos tipo Zeno habría que incluir en los algoritmos de control un parámetro T_{min} que marque el tiempo mínimo entre eventos (Lunze and Lehmann [79]) y que evite así estos fenómenos indeseados en el sistema. Fijar este parámetro supone buscar un compromiso entre las prestaciones del sistema y el poder evitar los efectos de *chattering*.

La plataforma de experimentación desarrollada en esta tesis trabaja de forma nativa en el dominio discreto. Por tanto, la implementación de las soluciones basadas en eventos ya llevan implícito un tiempo mínimo entre eventos que lo marca el periodo de muestreo del sistema $1/f_s$. En este caso, los algoritmos planteados en este trabajo ya están protegidos del efecto Zeno siempre y cuando se ejecuten en la plataforma aquí desarrollada. Por otro lado, si se exportan estas ideas a otras plataformas que no tengan esta limitación, se debe contemplar este mecanismo de protección en los mencionados algoritmos para evitar estos efectos indeseados.

Por tanto, la plataforma desarrollada para este trabajo es estrictamente hablando un sistema de control basado en eventos periódico (Heemels and Donkers [54]), que implícitamente es inmune al efecto Zeno. Además tiene una frecuencia de muestreo elevada en relación a la dinámica del robot, con lo que se garantizan también las prestaciones del sistema de control. El autor de esta tesis también ha desarrollado arquitecturas de control basadas en estas mismas ideas en plantas más complejas como los UAVs (Socas et al. [128]).

3.3. Estabilidad de los Algoritmos de Navegación

Para comprobar la estabilidad de la solución propuesta, se tendrán en cuenta los algoritmos de navegación típicos que utilizan este tipo de robots. Los robots móviles diferenciales suelen estar controlados por un sistema híbrido el cual aplica un algoritmo de navegación u otro en función de la información que obtenga del entorno. Así, los algoritmos más generales de navegación se pueden englobar en una de estas tres categorías (Siegwart et al. [126], Nehmzow [101], Nehmzow [102]):

1. Directo al Destino (DD) - *Go to Goal*.
2. Evitar Obstáculos (EO) - *Avoid Obstacles*.
3. Seguimiento de Paredes (SP) - *Wall Following*.

Todos estos algoritmos de navegación tienen como objetivo el determinar la velocidad lineal v y angular angular ω que permita al robot seguir la trayectoria adecuada en cada una de las situaciones anteriores. Por tanto, la dinámica de dichos robots queda descrita según la Ecuación 3.1.

$$\begin{aligned} \dot{x} &= v cos(\varphi) \\ \dot{y} &= v sin(\varphi) \\ \dot{\varphi} &= \omega \end{aligned} \tag{3.1}$$

Desde un punto de vista práctico, el robot tiene que ser guiado por las velocidades angulares que controlan cada motor, es decir, ω_r para la rueda derecha y ω_l para la rueda izquierda. Por tanto, una vez que el sistema de control determina la velocidad lineal v y angular ω del robot, éstas deben ser traducidas a las velocidades de cada una de las ruedas teniendo en cuenta los parámetros geométricos del robot (r radio de la rueda y L distancia entre las rudas) según la Ecuación 3.2.

$$\begin{aligned} \omega_r &= \frac{2V+\omega L}{2r} \\ \omega_l &= \frac{2V-\omega L}{2r} \end{aligned} \tag{3.2}$$

3.3.1. Estabilidad de los Algoritmos Directo al Destino - DD

Analizando la Figura 3.3 se observa que algoritmo DD tiene como objetivo encontrar la velocidad lineal v y angular w del robot de tal modo que la posición del robot converja hacia el punto de destino (x_g, y_g) de forma estable y en un tiempo finito.

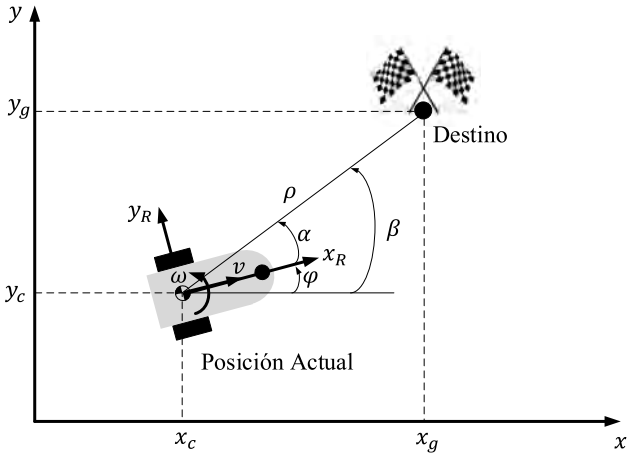

Figura 3.3: Posición y orientación para el algoritmo DD.

En este contexto, si se define una velocidad lineal v según la Ecuación 3.3, una velocidad angular ω según la Ecuación 3.4 y el sistema trabaja en el dominio continuo, se tiene una arquitectura estable según Lyapunov tal como quedó demostrado en la sección 2.4.

$$v = K_\rho \rho cos(\alpha) \quad , \quad K_\rho > 0 \tag{3.3}$$

$$\omega = K_\rho sin(\alpha)cos(\alpha) + K_\alpha \alpha \quad , \quad K_\alpha > 0 \tag{3.4}$$

La estabilidad de este algoritmo está garantizada según Lyapunov siempre y cuando el sistema trabaje en tiempo continuo. Para garantizar que este algoritmo sigue siendo estable en tiempo discreto, basta con elegir una frecuencia de muestreo f_s lo suficientemente alta en relación a la dinámica del robot y así conseguir que dicho sistema de control sea estable.

Para analizar el comportamiento de esta solución tanto en el dominio discreto como en la arquitectura basada en eventos propuesta en este trabajo, se hará un análisis basado en el esquema mostrado en la Figura 3.4. En este caso, se requiere por ejemplo que el robot parta de una posición inicial $(0, 0)$ cm y alcance

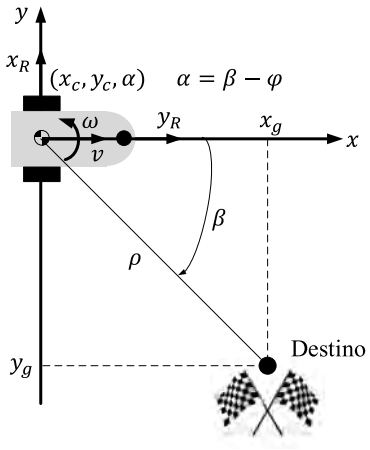

Figura 3.4: Esquema para el análisis del algoritmo DD.

el punto $x_g = 50$ cm e $y_g = -50$ cm de forma estable. Se considerarán diferentes frecuencias de muestreo f_s para el caso discreto y un nuevo algoritmo de control, que será una de las propuestas de este trabajo, para la solución basada en eventos. En cuanto a los parámetros K_ρ y K_α se analizarán diferentes conjuntos de valores obteniendo así los diferentes escenarios de análisis mostrados en la Tabla 3.1. En esta tabla el campo color representativo se utiliza para diferenciar cada uno de los escenarios en las gráficas de los resultados.

Tabla 3.1: Escenarios de análisis para el algoritmo DD.

Escenario	Color Representativo	K_ρ(1/s)	K_α(1/s)
1	azul	0,1	0,5
2	rojo	0,1	0,2
3	verde	0,1	0,1

Algoritmo DD en Control Discreto

En un sistema discreto, el modelo dinámico del robot diferencial está descrito por las siguientes ecuaciones en diferencias:

$$\begin{aligned} x[n+1] &= x[n] + v[n]\Delta t cos(\varphi[n]) \\ y[n+1] &= y[n] + v[n]\Delta t sin(\varphi[n]) \\ \varphi[n+1] &= \varphi[n] + \omega[n]\Delta t \end{aligned} \tag{3.5}$$

donde v es la velocidad lineal, ω la velocidad angular y Δt el periodo de muestreo del sistema. De la misma forma, las leyes de control para la velocidad lineal y

angular deben ser reescritas en forma discreta según las expresiones:

$$v[n] = K_\rho \rho[n] cos(\alpha[n]) \quad , \quad K_\rho > 0 \tag{3.6}$$

$$\omega[n] = K_\rho sin(\alpha[n]) cos(\alpha[n]) + K_\alpha \alpha[n] \quad , \quad K_\alpha > 0 \tag{3.7}$$

El sistema de control en tiempo discreto aquí presentado funciona de la siguiente forma (ver Figura 3.1):

1. El robot en cada periodo de muestreo envía al controlador por el enlace radio la información de los sensores de distancia recorrida de cada una de las ruedas.

2. El controlador, mediante odometría estima la posición actual del robot (x_c, y_c, φ) y la compara con la señal de referencia (x_g, y_g).

3. De la comparación anterior obtiene las variables $\rho[n]$ y $\alpha[n]$ y junto con los parámetros de control K_ρ y K_α, se obtienen las velocidades $v[n]$ y $\omega[n]$ mediante las Ecuaciones 3.6 y 3.7.

4. Finalmente, mediante el enlace radio, el controlador envía estas velocidades al robot para modificar su trayectoria.

Para estudiar la estabilidad de este sistema discreto en un entorno de simulación, se han analizado los tres escenarios de la Tabla 3.1. A su vez, se han hecho modificaciones en el periodo de muestreo Δt. Los resultados obtenidos considerando dos frecuencias de muestreo diferentes se muestran en las siguientes figuras. En la Figura 3.5 para una f_s=10 Hz y en la Figura 3.6 para una f_s=0,3 Hz.

Tal como se observa en la Figura 3.5, para una frecuencia de muestreo alta, el sistema tiene un comportamiento totalmente estable (véase Figura 3.5 a) y 3.5 b)). En todos los escenarios, el robot llega a su destino aproximadamente al mismo tiempo y con unas velocidades que tienden asintóticamente a 0 a medida que el robot se aproxima al destino (Figura 3.5 c) y 3.5 d)). Por otro lado, los parámetros K_ρ y K_α modulan la curvatura de la trayectoria de aproximación al destino. Estos parámetros son fundamentales para no sobrepasar las limitaciones mecánicas del robot en cuanto a velocidades lineales y angulares máximas y por tanto añaden dos grados de libertad al sistema para hacer un ajuste más preciso de éste.

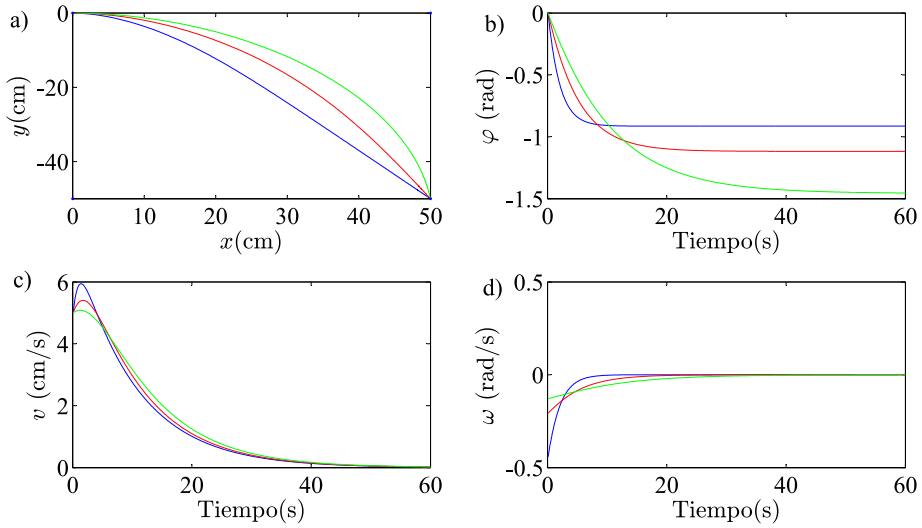

Figura 3.5: Respuesta del algoritmo DD discreto con $f_s = 10$ Hz.

Por otro lado, como era de esperar, a medida que se reduce la frecuencia de muestreo f_s el control del robot en este tipo de algoritmos empieza a manifestar comportamientos inestables tal como se muestra en la Figura 3.6. Un hecho relevante de este análisis, aunque no es objeto de esta tesis, muestra que si se tienen sistemas con frecuencias de muestreo bajas, el reducir los parámetros K_ρ y K_α, reduce la frecuencia de la dinámica del sistema y como resultado aumenta el rango de estabilidad del sistema global (Escenarios 2 y 3 en la Figura 3.6).

3.3.2. Adaptación del Algoritmo DD al Control basado en Eventos

El algoritmo de navegación DD descrito anteriormente, se implementará ahora en la arquitectura de control basada en eventos propuesta en este trabajo de investigación. Una de las condiciones que se imponen a esta solución es que siga siendo estable tal como lo son las versiones en tiempo continuo y discreto analizadas previamente. El criterio de estabilidad establece que el robot partiendo de una posición inicial (x_c, y_c, φ) converge de forma homogénea al punto destino (x_g, y_g) en un tiempo finito. En la Figura 3.7 se presenta el algoritmo propuesto desarrollado en control basado en eventos que debe cumplir estos requisitos. La filosofía de este algoritmo se basa en los siguientes postulados:

1. El controlador envía al robot las velocidades iniciales $v[k]$ y $\omega[k]$. Estas

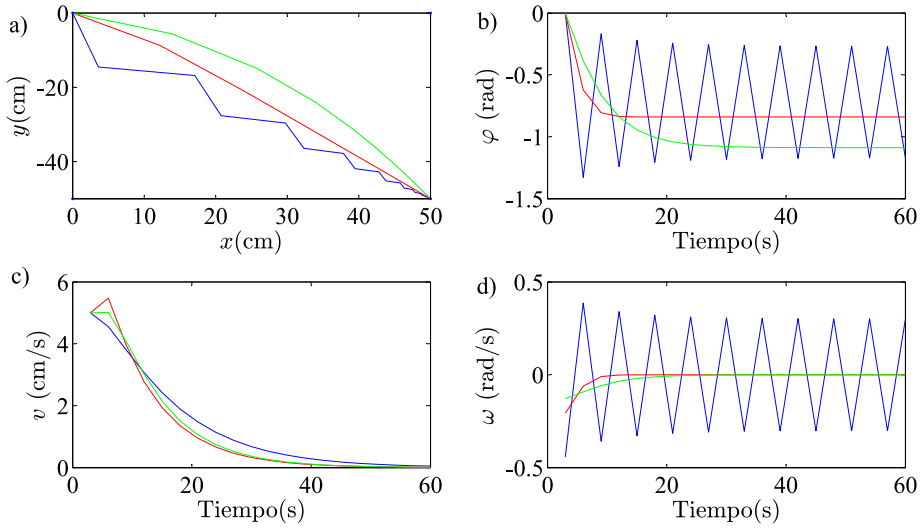

Figura 3.6: Respuesta del algoritmo DD discreto con $f_s = 0,3$ Hz.

velocidades las obtiene el controlador a partir de la posición inicial del robot (x_c, y_c, φ), el punto destino (x_g, y_g) y estimando los valores iniciales de la distancia al destino ρ_0 y el error inicial en el ángulo de orientación α_0.

2. El robot mediante odometría va estimando su posición y obteniendo los parámetros ρ y α. Para ello se hace uso de las medidas de distancia recorrida por cada una de sus ruedas (d_l, d_r) y las señales de referencia (x_g, y_g).

3. Si el parámetro ρ es inferior a la cota definida por e_ρ (condición de parada), significa que el robot ha llegado a su destino y se detiene.

4. En el caso de que aún no haya llegado al destino, se evalúa el parámetro α. Si esta variable es superior al *event threshold* $\bar{e}_\alpha$ (condición de evento $|\alpha| > \bar{e}_\alpha$), se produce un evento en el sistema y se solicita al controlador un nuevo valor para las velocidades v y ω. El controlador calculará estas velocidades según las Ecuaciones 3.8 y 3.9, donde k es el instante cuando se produce el evento. Estas nuevas velocidades son transmitidas al robot para que modifique su trayectoria.

$$v[k] = K_\rho \rho[k] cos(\alpha[k]) \quad , \quad K_\rho > 0 \tag{3.8}$$

$$\omega[k] = K_\rho sin(\alpha[k]) cos(\alpha[k]) + K_\alpha \alpha[k] \quad , \quad K_\alpha > 0 \tag{3.9}$$

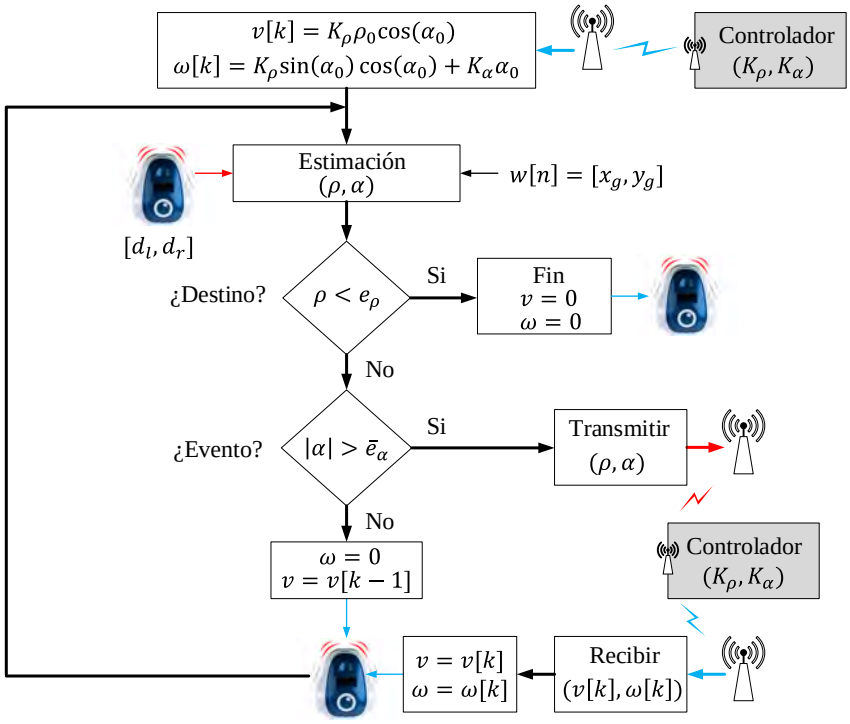

Figura 3.7: Algoritmo DD propuesto para el sistema de control basado en eventos.

5. Por último, mientras no se produzca la condición de evento, lo cual sucede cuando el robot está bien orientado hacia el destino ($\alpha \approx 0$), el robot mantiene la última velocidad lineal enviada por el controlador $v = v[k-1]$ y una velocidad angular nula ($\omega = 0$).

En el algoritmo propuesto (véase Figura 3.7) el controlador envía al robot las velocidades $v[k]$ y $\omega[k]$. La mayoría de los robots diferenciales no aceptan como entradas la velocidad lineal y angular, y como tal hay que enviarles las velocidades que deben dirigir cada una de sus ruedas, bien sean velocidades angulares ($\omega_l[k]$ y $\omega_r[k]$) o lineales ($v_l[k]$ y $v_r[k]$). Por tanto, desde un punto de vista práctico, a la hora de implementar estos algoritmos hay que hacer la transformación a estas velocidades ($v[k]$ y $\omega[k]$) a ($v_l[k]$ y $v_r[k]$) en el controlador. En el presente trabajo se ha hecho un proceso de identificación y calibrado de estas velocidades para que puedan realizarse estas transformaciones en el sistema de experimentación (véase Apéndice C).

Analizando con detalle el algoritmo de navegación propuesto se obtienen las siguientes conclusiones:

1. La convergencia del algoritmo está garantizada ya que si el robot no tiene

la orientación correcta, tanto la velocidad lineal v como la angular ω se calculan con las expresiones 3.8 y 3.9. Estas leyes de control mantienen la estabilidad según Lyapunov tal como se demostró anteriormente.

2. Una vez que el robot tiene la orientación correcta ($\alpha \approx 0$), se mantiene una velocidad lineal $v = v[k-1]$ que ya no depende de ρ y que por tanto será más alta que las obtenidas con los algoritmos tradicionales (continuo o discreto). Por tanto, este algoritmo tiene un tiempo de convergencia menor.

3. Por último, gracias a la condición de evento ($|\alpha| > \bar{e}_\alpha$), mientras no existan eventos en el sistema, tanto el controlador como el canal de comunicaciones radio no están consumiendo recursos. Por tanto, esta solución tiene mayor eficiencia en el uso de recursos que los controladores tradicionales.

Para analizar en un entorno de simulación como se comporta el algoritmo propuesto, se han analizado los mismos escenarios estudiados anteriormente con el sistema discreto (véase Figura 3.4 y Tabla 3.1). En este caso, el sistema tiene dos nuevos parámetros, el *event threshold* $\bar{e}_\alpha$ y la condición de parada e_ρ. Con estos nuevos elementos, fijando el *event threshold* $\bar{e}_\alpha$ a 0,01 rad y la condición de parada en e_ρ a 0,5 cm, los escenarios analizados se resumen en la Tabla 3.2. El campo çolor representativo"se utiliza para diferenciar cada uno de los escenarios en las gráficas de los resultados.

Tabla 3.2: Escenarios de análisis del algoritmo DD para el sistema de control basado en eventos.

Escenario	Color Representativo	K_ρ(1/s)	K_α(1/s)	$\bar{e}_\alpha$ (rad)	e_ρ (cm)
1	azul	0,1	0,5	0,01	0,5
2	rojo	0,1	0,2	0,01	0,5
3	verde	0,1	0,1	0,01	0,5

En la Figura 3.8 se presentan los resultados obtenidos en estas simulaciones.

En estas gráficas puede observarse que el algoritmo tiene un comportamiento estable y alcanza el objetivo en un tiempo finito como impone la condición de estabilidad. Al igual que sucedía con el algoritmo discreto, los parámetros K_ρ y K_α determinan la trayectoria y el tiempo de convergencia del algoritmo. A medida que K_α aumenta, disminuye el tiempo de convergencia, aunque este parámetro está limitado por la máxima velocidad angular ω que puede alcanzar el robot.

Tal como se mencionó en las propiedades que posee esta nueva estrategia de control, este sistema basado en eventos tiene un tiempo de convergencia (T_c)

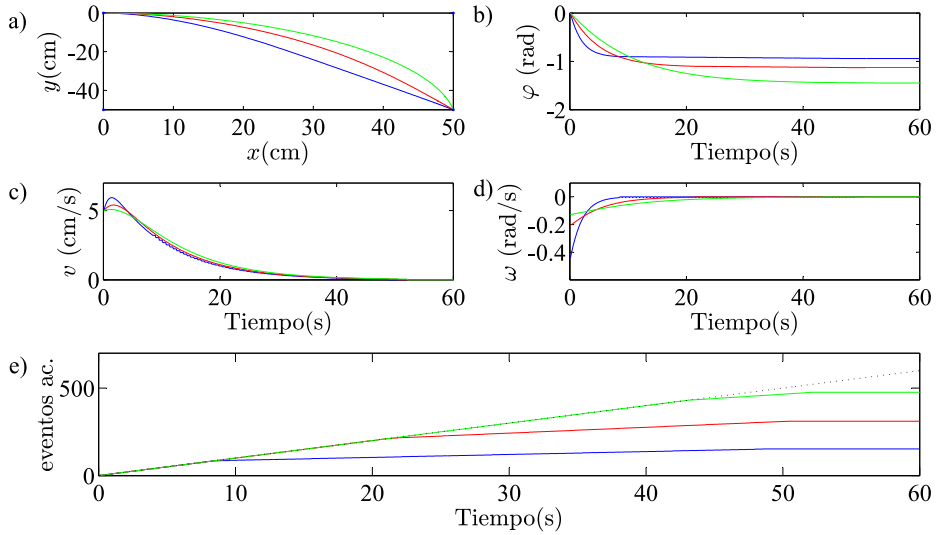

Figura 3.8: Respuesta del algoritmo DD en el sistema de control basado en eventos.

menor al mantener una velocidad lineal v más alta cuando el robot está orientado hacia el objetivo ($\alpha \approx 0$). Si se compara el tiempo de convergencia del controlador discreto (con f_s=10 Hz Figura 3.5) con el propuesto en este trabajo (Figura 3.8), considerando que se ha llegado al destino cuando $\rho \leq 0,5$ cm, se obtienen los tiempos mostrados en la Tabla 3.3.

Tabla 3.3: Tiempo de convergencia de los diferentes escenarios.

Escenario	T_c discreto (s)	T_c eventos (s)	Mejora (%)
1	50,0	48,9	2
2	50,8	50,5	1
3	52,2	52,2	0

Como se observa en dicha tabla, en los escenarios donde K_α es pequeña, ambos sistemas tienen un tiempo de convergencia similar. A medida que este parámetro aumenta el sistema basado en eventos mejora los tiempos de convergencia respecto al sistema discreto.

Respecto al grado de ocupación del canal de comunicación, los escenarios implementados con la propuesta basada en eventos siempre mejoran el consumo de recursos de transmisión respecto a las soluciones basadas en tiempo discreto. En el sistema discreto hay una comunicación periódica marcada por la frecuencia de muestreo. En cambio, en la solución basada en eventos solo se transmite cuando hay un evento en el sistema. Estas eficiencias en comunicaciones se muestran en

la Figura 3.8 e) (el trazo de puntos se corresponde con las muestras acumuladas del sistema discreto). Además, considerando los parámetros del controlador se observa que a medida que el parámetro K_α se incrementa, el control basado en eventos ocupa menos tiempo el canal de comunicaciones. En el trabajo Socas et al. [129] se desarrollan con más detalles estas ideas.

3.3.3. Estabilidad de los Algoritmos para Evitar Obstáculos (EO) y Seguimiento de Paredes (SP)

En esta sección se estudiará la filosofía general que rigen los algoritmos para evitar obstáculos y de seguimiento de paredes, así como su comportamiento. Posteriormente, se sentarán las bases de su implementación en el sistema de control basado en eventos propuesto en este trabajo (véase Socas et al. [131]).

Tanto los algoritmos EO como SP se apoyan en los sensores infrarrojos que estiman a qué distancia se encuentran los obstáculos. Previo a analizar estos mecanismos de navegación, se hará un breve análisis de cómo procesar la información obtenida en los mencionados sensores. Dado que estos sensores miden las distancias respecto a donde se encuentre situado dicho sensor en el robot, se necesitará un mecanismo para transformar esta información al sistema de referencia global y poder utilizarla en los mencionados algoritmos de navegación. En la Figura 3.9 se representa el caso general de un robot diferencial con sensores de obstáculos basados en infrarrojos.

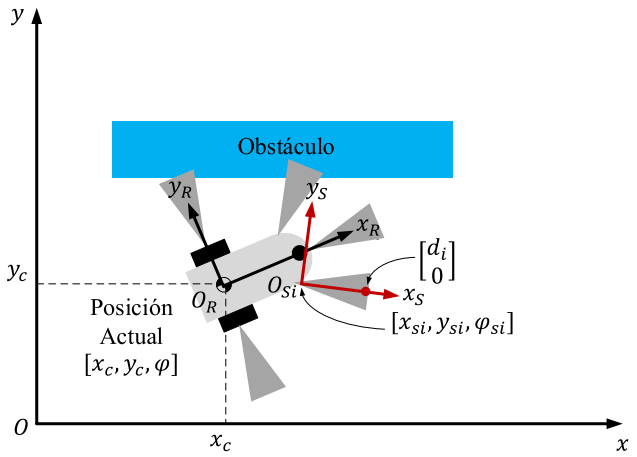

Figura 3.9: Posición de los sensores de obstáculos respecto al sistema inercial.

Como se observa en la Figura 3.9 hay definidos tres sistemas de referencia:

1. Sistema de referencia inercial O sobre el que se referencian los algoritmos de navegación.

2. Sistema de referencia en el cg del robot O_R.

3. Finalmente, un sistema de referencia O_{Si} situado en la posición de cada sensor i y orientado según esté éste respecto al cg del robot.

Cada sensor i mide la distancia $\mathbf{d_i}_{O_{Si}}$ a la que se encuentra un obstáculo respecto a su sistema de referencia O_{Si}, siendo esta medida un valor que solo tiene componente en el eje x_s y que puede escribirse según la expresión:

$$\mathbf{d_i}_{O_{Si}} = \begin{pmatrix} d_i \\ 0 \\ 1 \end{pmatrix} \tag{3.10}$$

Por otro lado, cada sensor i tiene una posición y una orientación respecto al sistema de referencia del robot O_R. Para referenciar las medidas de los sensores respecto al sistema de referencia del robot necesitamos las matrices de transformación $\mathbf{R}_{SR}(x_{si}, y_{si}, \varphi_i)$ de cada sensor i tal como se muestra en la siguiente expresión:

$$\mathbf{R}_{SR}(x_{si}, y_{si}, \varphi_i) = \begin{pmatrix} cos(\varphi_i) & -sin(\varphi_i) & x_{si} \\ sin(\varphi_i) & cos(\varphi_i) & y_{si} \\ 0 & 0 & 1 \end{pmatrix} \tag{3.11}$$

Como siguiente paso, se necesita una segunda matriz, según se muestra en la Ecuación 3.12, para transformar las medidas de los sensores obtenidas respecto al cg del robot y referenciarlas al sistema inercial O.

$$\mathbf{R}_{RO}(x_c, y_c, \varphi) = \begin{pmatrix} cos(\varphi) & -sin(\varphi) & x_c \\ sin(\varphi) & cos(\varphi) & y_c \\ 0 & 0 & 1 \end{pmatrix} \tag{3.12}$$

Finalmente, las medidas obtenidas por los sensores $\mathbf{d_i}_{O_{Si}}$ deben ser transformadas según la Ecuación 3.13 para obtener esa misma información referenciada al sistema universal $\mathbf{d_i}_{O}$ para que puedan ser tratadas de forma homogénea en los algoritmos de navegación.

$$\mathbf{d_{i}}_{O} = \begin{pmatrix} x_{di} \\ y_{di} \\ 1 \end{pmatrix} = \mathbf{R}_{RO}(x_c, y_c, \varphi) \cdot \mathbf{R}_{SR}(x_{si}, y_{si}, \varphi_i) \cdot \begin{pmatrix} d_i \\ 0 \\ 1 \end{pmatrix} \tag{3.13}$$

Algoritmos EO

Un algoritmo para Evitar Obstáculos (EO) tiene como objetivo que el robot alcance el destino sin ninguna colisión. El planteamiento de este algoritmo se muestra en la Figura 3.10. Tal como se observa en la figura, cuando los sensores

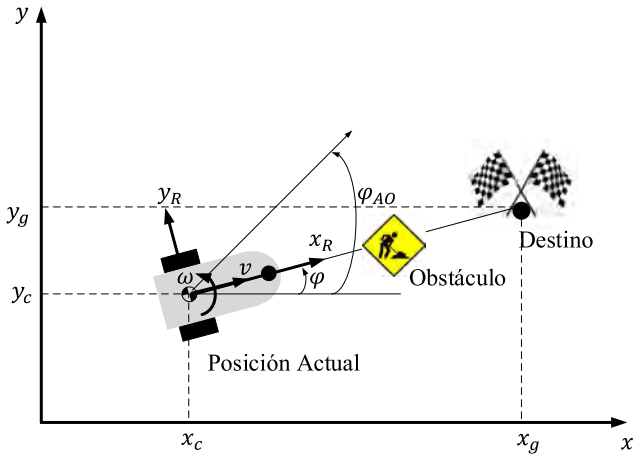

Figura 3.10: Parámetros del algoritmo EO.

de obstáculos detectan un objeto que interrumpe la trayectoria hacia el destino, el algoritmo de control debe modificar la orientación del robot para evitarlo. Si el robot tiene un orientación φ para alcanzar el destino, cuando detecta un obstáculo, esta orientación debe ser modificada a un nuevo valor φ_{AO} de tal forma que evite el obstáculo.

En este caso, existen diferentes planteamientos para obtener la nueva orientación del robot φ_{AO}. Los métodos más comunes se pueden clasificar en:

1. *Conmutación Estricta* (*Hard Switches*): Estos métodos introducen un ángulo fijo ($\frac{\pi}{2}$ o π) en la orientación del robot independientemente de la distancia a la que se encuentre el obstáculo, es decir:

$$\varphi_{AO} = \varphi \pm \gamma \quad \text{donde } \gamma \text{ puede ser } \pi \text{ o } \frac{\pi}{2} \tag{3.14}$$

2. *Combinados* (*Blending*): La filosofía de estos algoritmos se basan en considerar la distancia a la que se encuentra el obstáculo y en función de ésta

aplicar un cambio de dirección proporcional a esta distancia. La idea general de estos algoritmos se presenta en la Figura 3.11.

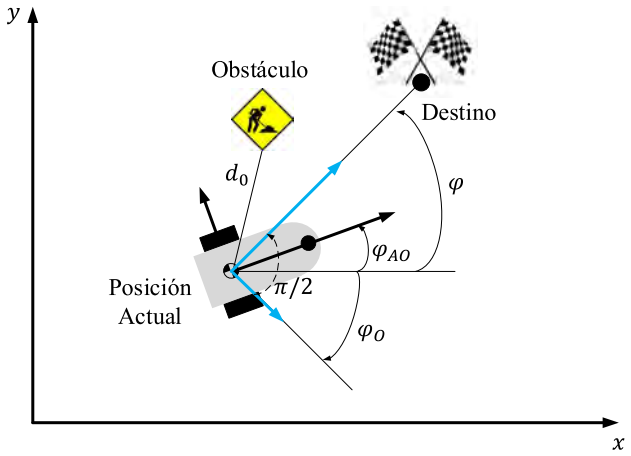

Figura 3.11: Esquema general del algoritmo EO combinado *Blending*.

Analizando dicha figura, si se define la orientación del robot para evitar obstáculos como $\varphi_O = \varphi - \pi/2$ y una función *blending* $\sigma()$ (véase Figura 3.12) que dependa de la distancia al obstáculo d_0 según la Ecuación 3.15.

$$\sigma(d_0) = 1 - e^{d_s d_0} \tag{3.15}$$

Con esta nueva función se puede modular según el alcance que tengan los sensores del robot mediante el parámetro d_s. La elección idónea de d_s debe ser tal que la función $\sigma(d_0)$ esté muy próxima a 1 cuando d_0 esté cercano al máximo alcance del sensor. Tal como se muestra en la Figura 3.12 para un sensor con un alcance de 2 cm el valor del parámetro d_s más adecuado es $2,0$ (trazo verde), para uno de 5 cm d_s debe ser $1,0$ (trazo rojo) y finalmente para un alcance de 10 cm d_s debe ser $0,5$ (trazo azul).

Por tanto, se puede definir el algoritmo EO según la Ecuación 3.16.

$$\varphi_{AO} = \sigma(d_0)\varphi + (1 - \sigma(d_0))\varphi_O \tag{3.16}$$

Tanto en los algoritmos *Hard Switch* como en los *Blending*, una vez se ha modificado la trayectoria del robot para evitar el obstáculo, la velocidad angular ω se pone a cero y la velocidad lineal v permanece constante.

Tal como se observa en la forma de trabajar de los algoritmos EO, la convergencia de estos algoritmos está asegurada. La razón de que no haya problemas

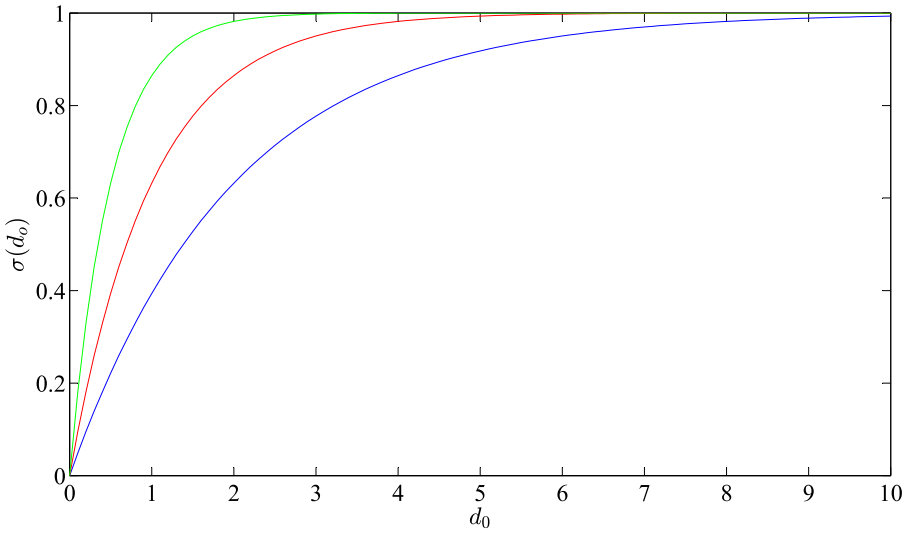

Figura 3.12: Función *Blending* $\sigma(d_0)$ para $d_s = 0,5$ en azul, $d_s = 1,0$ en rojo y $d_s = 2,0$ en verde.

de convergencia se debe a que el algoritmo solo debe modificar la orientación del robot mientras exista un obstáculo entre el robot y el destino sin que exista ningún requisito de tiempo para ello. Una vez el obstáculo ha sido superado, nuevamente el algoritmo DD es quien dirige al robot por lo que está garantizada la convergencia hacia el destino.

Algoritmos SP

El algoritmo de seguimiento de paredes SP es uno de los algoritmos de navegación más usados en aplicaciones de robots móviles. La idea general del algoritmo es la siguiente:

1. Una vez se detecta una pared (obstáculo), el robot se mueve hacia ella hasta alcanzar un distancia determinada d_w.

2. Alcanzada la distancia d_w, el robot gira y va navegando paralelo a la pared a esa distancia.

3. Finalmente, cuando se haya rebasado la pared, se aplica nuevamente el algoritmo DD.

Los algoritmos de seguimiento de paredes suelen tener dos variantes: los que hacen el seguimiento de la pared en el sentido de las agujas del reloj (*clockwise*) o los que lo hacen en sentido contrario (*counter-clockwise*).

Un algoritmo típico de seguimiento de paredes se puede formular considerando los elementos mostrados en la Figura 3.13. Los vectores $\mathbf{p_1}$ y $\mathbf{p_2}$ son las medidas de

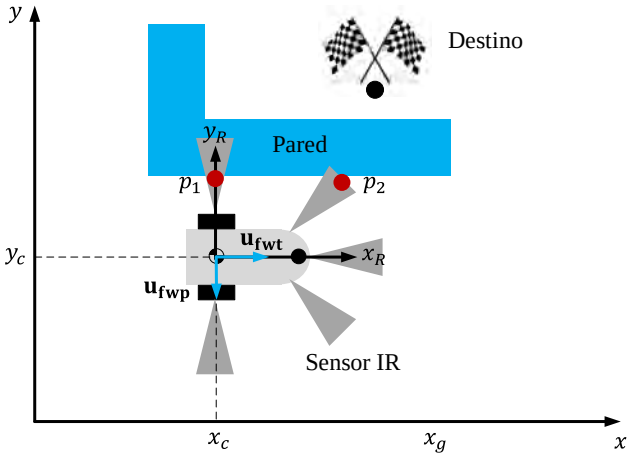

Figura 3.13: Elementos del algoritmo SP.

los sensores de obstáculos laterales del robot referenciados al sistema de referencia universal O. Con dichos puntos se puede obtener el vector unitario paralelo a la pared $\mathbf{u_{fwt}}$ mediante la expresión:

$$\mathbf{u_{fwt}} = \frac{\mathbf{p_2} - \mathbf{p_1}}{\|\mathbf{p_2} - \mathbf{p_1}\|} \tag{3.17}$$

La posición del robot se expresa mediante el vector $\mathbf{p_c} = (x_c, y_c)^T$. Con $\mathbf{p_c}$ y $\mathbf{u_{fwt}}$, se puede obtener el vector $\mathbf{v_{rw}}$ que apunta desde el centro de referencia del robot al punto más cercano de la pared. Este nuevo vector puede obtenerse mediante la Ecuación 3.18.

$$\mathbf{v_{rw}} = (\mathbf{p_1} - \mathbf{p_c}) - ((\mathbf{p_1} - \mathbf{p_c}) \cdot \mathbf{u_{fwt}}) \cdot \mathbf{u_{fwt}} \tag{3.18}$$

Por tanto, un vector en dirección opuesta al punto más cercano a la pared y ponderado por la distancia d_w a la que se quiere que el robot se mantenga con respecto a esta, puede escribirse utilizando la expresión siguiente:

$$\mathbf{u_{fwp}} = \mathbf{v_{rw}} - d_w \frac{\mathbf{v_{rw}}}{\|\mathbf{v_{rw}}\|} \tag{3.19}$$

Finalmente, se puede fijar el rumbo del robot para hacer el seguimiento de paredes mediante el vector $\mathbf{u_{fw}}$ según la Ecuación 3.20. Esta orientación se obtiene de la composición de $\mathbf{u_{fwt}}$ y $\mathbf{u_{fwp}}$ con dos grados de libertad según los parámetros k_1

y k_2.

$$\mathbf{u_{fw}} = k_1 \cdot \mathbf{u_{fwt}} + k_2 \cdot \mathbf{u_{fwp}} \tag{3.20}$$

Al igual que sucedía en los algoritmos de EO, los algoritmos SP no presentan problemas de estabilidad. La razón principal es que en este tipo de algoritmos no existe ninguna condición de convergencia a un punto en un tiempo finito y por tanto es un aspecto que simplifica el diseño de estos controladores.

Estructura General de los Algoritmos EO y SP en el Sistema de Control basado en Eventos

La forma de implementar tanto los algoritmos EO como SP en el sistema de control basado en eventos propuesto en este trabajo sigue la filosofía presentada en la Figura 3.14. Tal como se observa en el esquema, el robot está navegando

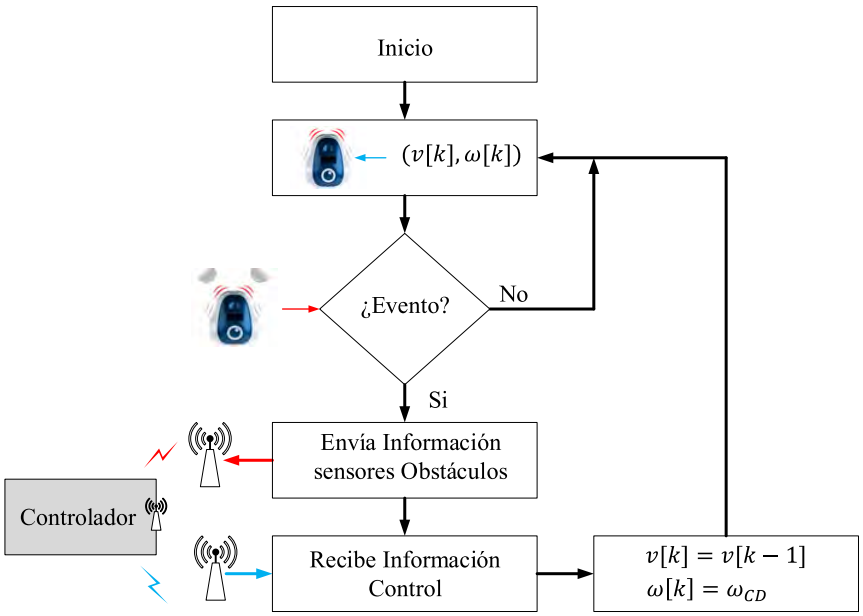

Figura 3.14: Arquitectura de los algoritmos EO y SP en el sistema de control basado en eventos.

hacia el destino con una velocidad lineal $v[k]$ y $\omega[k]$ que ha sido determinada por el controlador para alcanzar un destino concreto o para mantener la trayectoria deseada. Si durante el trayecto los sensores de obstáculos detectan algún elemento que se interpone en su camino, se genera un evento en el sistema y se informa de este hecho al controlador. El robot envía al controlador la información de los sensores de obstáculos y adicionalmente su posición. El controlador determina la nueva velocidad angular ω_{CD} que provoca un cambio de dirección para evitar

el obstáculo o seguir la pared según sea el caso. Al mismo tiempo, se mantiene la misma velocidad lineal $v[k-1]$ que había antes de producirse el evento ya que lo que se persigue ahora es cambiar solo la dirección del robot. Con estas nuevas velocidades, el robot sigue una nueva trayectoria, volviéndose a evaluar la información de los sensores para informar de cuando se produce nuevamente un evento. La forma de detectar los eventos y las leyes de control que aplica el controlador dependerán del algoritmo que se esté ejecutando en cada momento tal como se describirá a continuación.

Sensores de Obstáculos

Los algoritmos DD definidos anteriormente tanto en el dominio discreto como en el dominio de eventos, son aplicables a cualquier robot diferencial. En cambio, los algoritmos EO y SP están fuertemente condicionados a como sea la disposición de los sensores de obstáculos en el robot en el que se pretenda implementar la mencionada solución. Por tanto, y antes de explicar los algoritmos propuestos en el sistema basado en eventos, se describirán brevemente como son los sensores utilizados en el sistema de experimentación desarrollado en esta tesis.

En la Figura 3.15 se muestra un esquema simplificado de las plataformas de robots móviles utilizadas para realizar los experimentos en este trabajo. Como se

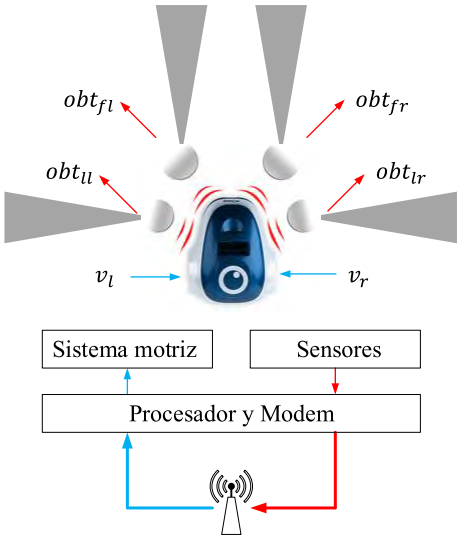

Figura 3.15: Esquema simplificado de la plataforma de robots.

observa los robots disponen de cuatro sensores de obstáculos, dos laterales obt_{ll} y obt_{lr} y dos frontales obt_{fl} y obt_{fr}. El primer subíndice indica su posición en el eje

longitudinal (l indica lateral y f frontal). El segundo subíndice indica la posición en el eje horizontal (l izquierda y r derecha).

Por otro lado, cada uno de estos sensores tienen una respuesta no lineal que ha sido obtenida mediante un proceso de identificación y calibración que se muestra en el Apéndice C.

Una vez analizadas las posiciones de los sensores de nuestra plataforma de experimentación, en las siguientes secciones se mostrarán las implementaciones propuestas para los algoritmos EO y SP en la arquitectura de control basada en eventos desarrollada en este trabajo.

3.3.4. Adaptación del Algoritmo EO al Control basado en Eventos

Siguiendo los esquemas mostrados en las Figuras 3.14 y 3.15, para el caso del algoritmo EO la condición de evento propuesta es la siguiente:

$$\begin{gathered}\textbf{if } [(obt_{fl} - w > \bar{e}_{EO}) \textbf{ OR } (obt_{ll} - w > \bar{e}_{EO}) \textbf{ OR } ... \\ ...(obt_{fr} - w > \bar{e}_{EO}) \textbf{ OR } (obt_{lr} - w > \bar{e}_{EO})] \\ \{evento = true\} \\ \textbf{else } \{evento = false\}\end{gathered} \tag{3.21}$$

Tal como se desprende de la Ecuación 3.21, los eventos se producirán siempre y cuando cualquiera de los cuatro sensores de obstáculos supere la señal de referencia w con un valor superior al *event threshold* $\bar{e}_{EO}$. Este proceso es el más relevante desde un punto de vista del control basado en eventos y será el que controle la actividad tanto del controlador como del uso del canal de comunicaciones. Como es obvio, tanto el controlador como el canal de comunicaciones estarán en reposo mientras no se cumpla esta condición de evento.

En el otro extremo, el controlador aplicará una ley que evite los obstáculos. En este caso el algoritmo de control EO propuesto se representa de forma esquemática en la Figura 3.16.

En este caso el algoritmo EO actúa según los siguientes principios:

1. Cada vez que se produce un evento (Ecuación 3.21), el robot envía al controlador a través de canal de comunicación radio la información de los sensores de obstáculos obt_{fl}, obt_{fr}, obt_{ll} y obt_{lr}.

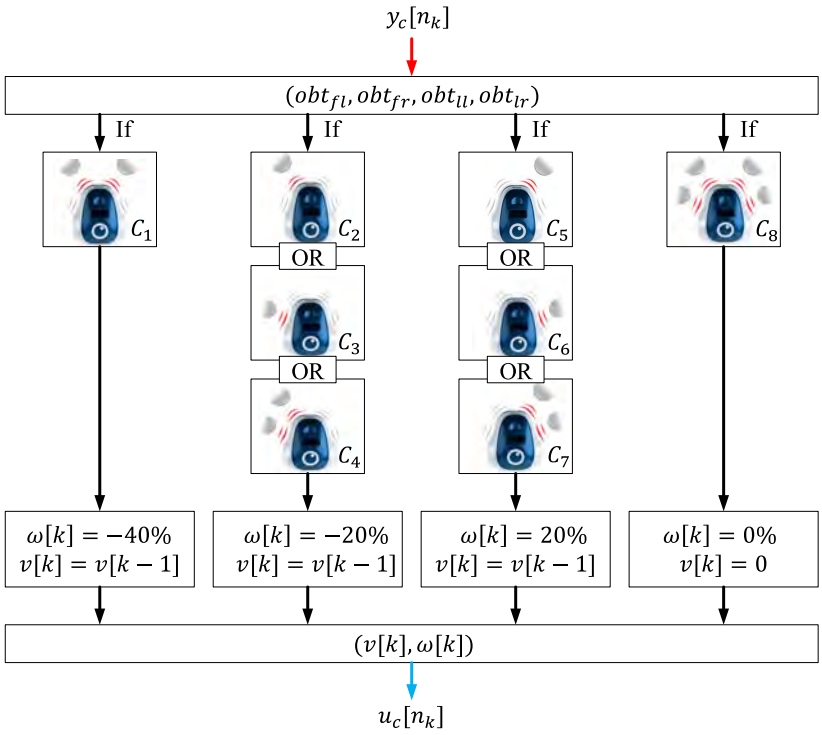

Figura 3.16: Algoritmo de control EO.

2. El controlador con la información de estos sensores determina en qué posición está el obstáculo o los obstáculos que impiden el paso del robot. Las condiciones C_1 - C_8 según las Ecuaciones 3.22 - 3.29 determinan las situaciones posibles de la posición del robot frente a los obstáculos.

$$\textbf{if } [(obt_{fl} > 0) \textbf{ AND } (obt_{fr} > 0)] \{C_1 = true\} \tag{3.22}$$

$$\textbf{if } (obt_{fl} > 0) \{C_2 = true\} \tag{3.23}$$

$$\textbf{if } (obt_{ll} > 0) \{C_3 = true\} \tag{3.24}$$

$$\textbf{if } [(obt_{fl} > 0) \textbf{ AND } (obt_{ll} > 0)] \{C_4 = true\} \tag{3.25}$$

$$\textbf{if } (obt_{fr} > 0) \{C_5 = true\} \tag{3.26}$$

$$\textbf{if } (obt_{lr} > 0) \{C_6 = true\} \tag{3.27}$$

$$\textbf{if } [(obt_{fr} > 0) \textbf{ AND } (obt_{lr} > 0)] \{C_7 = true\} \tag{3.28}$$

$$\begin{gathered}\textbf{if } [(obt_{fl} > 0) \textbf{ AND } (obt_{ll} > 0) \textbf{ AND } ... \\ ...(obt_{fr} > 0) \textbf{ AND } (obt_{lr} > 0)] \\ \{C_8 = true\}\end{gathered} \tag{3.29}$$

3. Finalmente, en función de estas condiciones C_1 - C_8 se aplica la ley de control según la Ecuación 3.30 y se obtienen las velocidades $v[k]$ y $\omega[k]$ que posteriormente son enviadas al robot.

$$\begin{array}{c} \textbf{if } (C_1)\ \{\omega[k] = -40\,\%, v[k] = v[k-1]\} \\ \textbf{else if } [(C_2)\ \textbf{OR}\ (C_3)\ \textbf{OR}\ (C_4)]\ \{\omega[k] = -20\,\%, v[k] = v[k-1]\} \\ \textbf{else if } [(C_5)\ \textbf{OR}\ (C_6)\ \textbf{OR}\ (C_7)]\ \{\omega[k] = 20\,\%, v[k] = v[k-1]\} \\ \textbf{else } (C_8)\ \{\omega[k] = 0\,\%, v[k] = 0\} \end{array} \tag{3.30}$$

Tal como se observa en la Figura 3.16 y en la Ecuación 3.30, en el algoritmo propuesto las velocidades angulares aplicadas dependen del número de sensores de obstáculos activos en cada momento. En esta solución se han calculado unas velocidades proporcionales a las velocidades angulares máximas del robot, éstas se hacen variar entre el 20 % y el 40 % de su valor máximo. Como solución a este problema de navegación, también pueden aplicarse otras velocidades angulares siguiendo una filosofía similar y resolver el problema de forma satisfactoria.

Por último, destacar que esta arquitectura de control dispone de varios parámetros que determinan el comportamiento final del sistema. Por un lado, el parámetro $\bar{e}_{EO}$ que determina el número de eventos del sistema y cuando se deben aplicar las leyes de control en función de la cercanía al obstáculo. Y por otro lado, las velocidades angulares a aplicar en el controlador dependiendo de la situación de los obstáculos en cada momento. Para este último caso, aunque hay ocho situaciones diferentes, C_1 - C_8, basta con fijar cuatro velocidades angulares diferentes para resolver el problema de navegación (Figura 3.16).

3.3.5. Adaptación del Algoritmo SP al Control basado en Eventos

La solución basada en eventos para el algoritmo de seguimiento de paredes SP se apoyará también en los mismos esquemas de las Figuras 3.14 y 3.15. Tal como se mostró en la sección 3.3.3, los algoritmos de seguimiento de paredes suelen tener dos variantes: los que hacen el seguimiento de la pared en el sentido de las agujas del reloj (*clockwise*) o los que lo hacen en sentido contrario (*counter-clockwise*). La solución propuesta y analizada en este trabajo ha sido la segunda *counter-clockwise* tal como se describirá a continuación.

La condición de evento propuesta es la siguiente:

$$\begin{gathered}\textbf{if } [(obt_{fl} > 0) \textbf{ OR } (abs(obt_{ll} - w) > \bar{e}_{SP})] \\ \{evento = true\} \\ \textbf{else } \{evento = false\}\end{gathered} \tag{3.31}$$

Según esta condición, se producirá un evento en el sistema cuando se cumpla alguna de las siguientes condiciones: 1) el sensor de obstáculos frontal delantero obt_{fl} esté detectando la pared o 2) el obstáculo lateral izquierdo obt_{ll} se encuentre a un distancia w de la pared con un margen marcado por el *event threshold* $\bar{e}_{SP}$.

En cuanto al controlador, la ley de control propuesta a aplicar en este algoritmo se resumen en la Ecuación 3.32.

$$\begin{gathered}\textbf{if } (obt_{fl} > 0) \\ \{v_r[k] = 0 \ , \ v_l[k] = v[k-1]\} \\ \textbf{else} \\ \{v_r[k] = v[k-1] \ , \ v_l[k] = (w - obt_{ll})\, v_w + v[k-1]\}\end{gathered} \tag{3.32}$$

En este caso, la ley descrita en la Ecuación 3.32 ha sido definida en función de las velocidades lineales de cada una de las ruedas del robot para entender mejor el funcionamiento del algoritmo. La velocidad $v_l[k]$ corresponde a la velocidad lineal de la rueda izquierda y $v_r[k]$ es la de la rueda derecha. Estas velocidades, una vez determinadas por el controlador, se transforman en la velocidad lineal $v[k]$ y la angular $\omega[k]$ global del robot según la Ecuación 3.33. De esta forma, estas magnitudes se corresponden con los esquemas generales para estos algoritmos de navegación mostrados en las Figuras 3.14 y 3.15.

$$\begin{gathered}v[k] = \frac{(v_r[k]+v_l[k])}{2} \\ \omega[k] = \frac{(v_r[k]-v_l[k])}{L}\end{gathered} \tag{3.33}$$

Volviendo a la ley de control propuesta en la Ecuación 3.32, ésta actúa de la forma siguiente:

1. Si el robot en su camino tiene un objeto que le impide su paso ($obt_{fl} > 0$), este trata de evitarlo frenando su rueda derecha $v_r[k] = 0$, manteniendo la velocidad lineal solo en la rueda izquierda $v_l[k] = v[k-1]$. De esta forma, el robot hace un giro pronunciado a la izquierda.

2. Cuando el robot está libre de obstáculos, la velocidad de la rueda derecha permanece constante $v_r[k] = v[k-1]$ y la de la izquierda se modula según la distancia a la que se encuentre el robot de la pared.

3. La distancia a la pared está determinada por el parámetro w. Este sirve de consigna para marcar a qué distancia debe mantenerse el robot respecto a la pared. En el caso de que el robot se encuentre a una distancia de la pared igual a w (objetivo del algoritmo), el término $(w - obt_{ll})\, v_w$ es cero y hará que el robot navegue con velocidad angular cero ($v_r[k] = v_l[k]$) y se desplace paralelo a la pared. Si el robot se encuentra a una distancia superior a la marcada por la referencia w ($obt_{ll} > w$) la velocidad de la rueda izquierda $v_l[k]$ se aminora por el factor v_w (velocidad de aproximación a la pared) y como consecuencia hará un leve giro a la izquierda. En el sentido contrario, cuando se cumpla que el robot esté más cerca que la referencia ($obt_{ll} < w$), la velocidad de la rueda izquierda será aumentada ponderándose por ese mismo factor v_w, forzando en este caso un giro del robot hacia la derecha.

Como resumen, en el algoritmo de seguimiento de paredes propuesto para el sistema de control basado en eventos intervienen principalmente dos parámetros:

1. El *event threshold* $\bar{e}_{SP}$ que determina la condición de evento.

2. El parámetro v_w que marca la velocidad de aproximación a la pared cuando el robot no está en un estado estacionario respecto a ésta.

3.4. Eficiencia en el Uso de los Recursos

Tal como se comentó en la Sección 3.2.2 las arquitecturas de control basado en eventos tienen una serie de ventajas respecto a los esquemas tradicionales como son:

1. Eficiencia en comunicaciones.

2. Ahorro en recursos de computación.

3. Menor consumo energético.

Por el contrario, este tipo de estrategias carecen de una metodología matemática sólida (aún en desarrollo) que garantice de forma directa la estabilidad de los lazos de control y que permitan hacer un desarrollo sistemático de aplicaciones basadas en estas ideas.

En la Sección 3.3 se han sentado las bases para garantizar la estabilidad de estas estructuras de control basadas eventos en algoritmos de navegación para robots móviles. También se ha propuesto la forma de implementar estos algoritmos con esta nueva estrategia de control a la vez que se han definidos los parámetros que gobiernan las prestaciones de estos sistemas. Por tanto, una vez garantizada la aplicabilidad de estas soluciones, resta por analizar las mejoras de eficiencia tanto en recursos de comunicaciones, computacionales y de consumo energético que esta nuevas estructuras proporcionan a los sistemas que las implementen.

Si comparamos la arquitectura de control basado en eventos desarrollada en este trabajo de investigación con su equivalente en el dominio discreto (véase Figura 3.17), encontramos las siguientes diferencias:

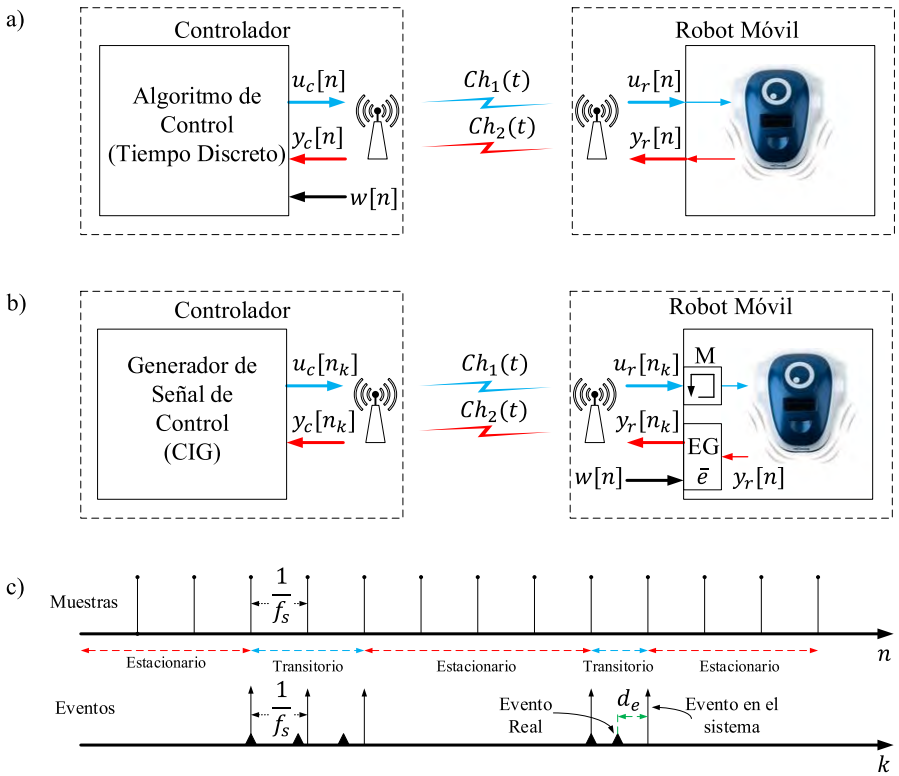

Figura 3.17: a) Arquitectura de control discreto. b) Arquitectura de control propuesta basada en eventos. c) Diagramas temporales de ambas soluciones.

1. El lazo de control discreto de la Figura 3.17 a) trabaja con una frecuencia de

muestro f_s que obliga al controlador a comunicarse de forma periódica y en ambas direcciones con el robot. Esta comunicación en la solución discreta es independiente de si el sistema está en estado transitorio o estacionario. En cambio, para el lazo basado en eventos de la Figura 3.17 b), la comunicación solo se produce en los instantes en que el sistema se encuentre en estado transitorio. En la Figura 3.17 c) se muestra como el sistema discreto actúa cada $1/f_s$, en cambio, el sistema basado en eventos solo tiene actividad cuando se cumple la condición que genera un evento en el sistema.

2. Aunque estrictamente hablando, un sistema de control basado en eventos debería ser capaz de producir un evento en cualquier instante, en el sistema desarrollado para este trabajo, un evento siempre se producirá en un instante que coincide con el periodo de muestreo. Tal como se explicó cuando se planteó la arquitectura, esto se debe a que la plataforma de robots móviles seleccionada trabaja en el dominio discreto de forma nativa y por tanto no se puede forzar a generar un evento en un instante de tiempo arbitrario.

3. El que los eventos solo se puedan generar con el intervalo del periodo de muestreo $\frac{1}{f_s}$ tiene dos implicaciones: La primera es que el sistema es inmune al fenómeno Zeno (*chattering*) ya que no se pueden producir un número infinitos de eventos en una situación transitoria extrema. La segunda es que existe un determinado retardo d_e entre el instante en que se produce el evento y cuando lo registra el sistema. Para este segundo aspecto, si se elige un periodo de muestreo $\frac{1}{f_s}$ suficientemente pequeño, se garantiza la estabilidad del sistema y se consigue que el efecto del retardo d_e sea despreciable.

4. Tal como se observa en la Figura 3.17, la actividad del sistema (envío de información desde el robot al controlador, utilizar el canal de comunicación, calcular las leyes de control en el controlador y enviar esta información al robot) está marcada por la sincronización de éste. En el caso discreto esta sincronización la marca el periodo de muestreo y para el caso del sistema basado en eventos, los instantes en que se producen éstos.

Desde un punto de vista de las prestaciones, los sistemas de control suelen medirse mediante el criterio Integral del Error Absoluto IAE (*Integral Absolute Error*). Este criterio es ampliamente usado tanto en sistema de control continuo

como en sistema de control discreto. La idea del IAE es medir la diferencia entre la salida del sistema ($y(t)$ para el continuo e $y[n]$ para el discreto) y la señal de referencia ($w(t)$ o $w[n]$ respectivamente). Este criterio puede expresarse de la manera siguiente:

$$\begin{aligned} IAE &= \int_{t_0}^{t} |w(t) - y(t)| dt \\ IAE &= \sum_{n_0}^{n} |w[n] - y[n]| \end{aligned} \tag{3.34}$$

Por tanto, el sistema a evaluar tendrá mejores prestaciones cuanto más pequeño sea el IAE. Otra forma de medir las prestaciones de un sistema de control, sobre todo cuando se trata de un sistema de control basado en eventos (Vasyutynskyy and Kabitzsch [144], Otanez et al. [107]), es considerar la Integral del Error Absoluto comparando con un sistema discreto equivalente $IAEP$ (*Integral Absolute Error compared to Periodic loop*). Este nuevo criterio compara la salida del sistema basado en eventos $y_{eve}[n]$ con su equivalente discreto $y[n]$ tal como se expresa en la Ecuación 3.35.

$$IAEP = \sum_{n_0}^{n} |y_{eve}[n] - y[n]| \tag{3.35}$$

Aunque el $IAEP$ es un criterio para medir las prestaciones de un sistema de control basado en eventos, desde un punto de vista práctico, muchos investigadores (Miskowicz [90], Pawlowski et al. [109], Lian et al. [73], Nguyen and Suh [103], Yook et al. [154]) se apoyan en el Ratio de Eventos N_N. Este nuevo criterio mide la relación entre el número de eventos y su equivalente en un sistema discreto según la expresión:

$$N_N = \frac{N_{eve}}{N_{per}} \tag{3.36}$$

donde N_{eve} es el número de eventos que produce el sistema bajo estudio y N_{per} las muestras en el sistema discreto equivalente en el mismo periodo de análisis. Tal como sucedía en el criterio de IAE o el $IAEP$, cuanto más pequeño sea el N_N más eficiente será el sistema basado en eventos respecto a su equivalente discreto.

Previo a analizar la eficiencia en el uso de los diferentes recursos del sistema, se definirá el concepto de Ratio de Actividad Empírico N_R de un recurso R. Este parámetro puede definirse como la relación entre el consumo de un recurso (ancho de banda, carga computacional, energía, etc.) en el sistema basado en eventos con respecto al consumo de ese mismo recurso en el sistema de control

discreto equivalente. Por tanto, se define N_R de la forma:

$$N_R = \frac{Consumo\_Recurso\_R(eve)}{Consumo\_Recurso\_R(dis)} \tag{3.37}$$

De la misma manera se puede definir la eficiencia $\eta_R(\,\%)$ en el consumo de un recurso R del sistema basado en eventos respecto al discreto según la Ecuación 3.38.

$$\eta_R(\,\%) = (1 - N_R)100 \tag{3.38}$$

Hay que destacar en este punto que, el ratio de actividad empírico de un recurso N_R tiene una cota inferior igual a N_N. Esta condición la marca el hecho de que el uso de un recurso concreto en el sistema basado en eventos no puede ser inferior al que marca el ratio de eventos N_N. Por tanto, de manera general $N_R \geq N_N$ y como consecuencia, la eficiencia de un recurso $\eta_R(\,\%)$ será siempre igual o inferior a la eficiencia máxima teórica $\eta_N(\,\%) = (1 - N_N)100$.

Desde un punto de vista analítico, lo ideal sería disponer de una estimación del valor de N_N y N_R para la arquitectura basada en eventos desarrollada en este trabajo. Dado que este factor tiene una gran dependencia del escenario donde esté navegando el robot no se puede obtener una formulación cerrada para estimar el valor de estos parámetros. Por tanto, en las siguientes secciones se propondrá la forma de medir la eficiencia de esta estructura de forma empírica tanto para los recursos de comunicaciones, computacionales como energéticos.

3.4.1. Eficiencia en Comunicaciones

Para analizar la eficiencia en el uso de recursos de comunicaciones que se obtiene en el sistema de eventos propuesto, se considerará el esquema mostrado en la Figura 3.18. Los recursos de comunicación se pueden medir teniendo en cuenta el ancho de banda utilizado (en bits por segundo, bps). El ancho de banda utilizado puede definirse según la expresión:

$$BW(\text{bps}) = \text{bits transmitidos/tiempo de análisis} \tag{3.39}$$

Tanto en el sistema discreto como en el basado en eventos, cada vez que se transmite información por el canal inalámbrico, se envía un paquete de información p_{inf} en ambos sentidos (Figura 3.18 a)). Este paquete de información tendrá una

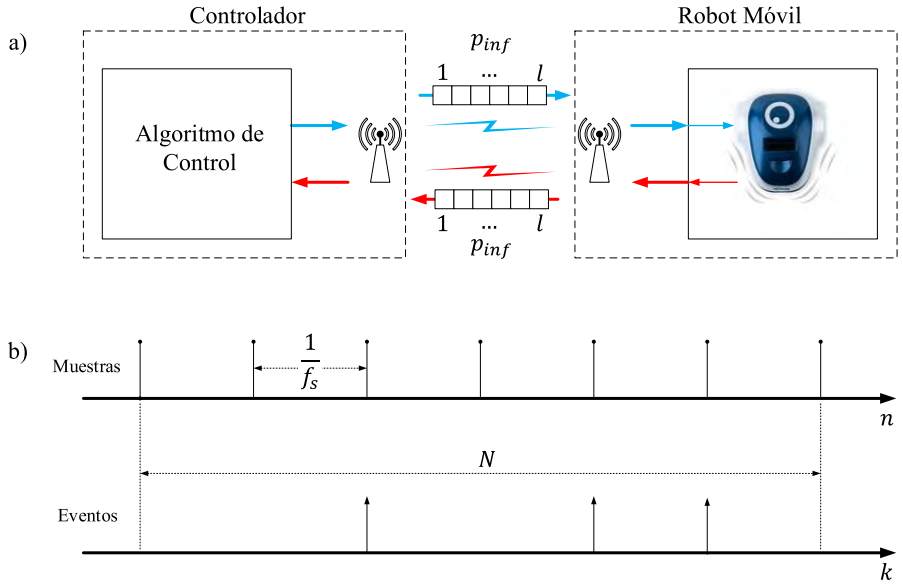

Figura 3.18: Modelo de recursos de comunicación.

longitud de bits determinada en función de como esté definido el protocolo de comunicaciones. Para este análisis, se considerará un paquete de información de l bits. Por otro lado, para comparar ambos sistemas, se considerará un tiempo de análisis N que se corresponde a un número de periodos discretos concreto N_{per}, con lo que se cumple que $N = N_{per}\frac{1}{f_s}$, (véase Figura 3.18 b)). Basado en estos postulados, se puede definir el ancho de banda ocupado por el sistema discreto según la ecuación:

$$BW_{dis}(\text{bps}) = \frac{lN_{per}}{N} \tag{3.40}$$

De igual forma se puede definir el ancho de banda ocupado por el sistema de eventos según la expresión:

$$BW_{eve}(\text{bps}) = \frac{lN_{eve}}{N} \tag{3.41}$$

Comparando las Ecuaciones 3.40 y 3.41 se puede obtener el ratio de actividad empírico para el ancho de banda N_{BW} dado por,

$$N_{BW} = \frac{BW_{eve}(\text{bps})}{BW_{dis}(\text{bps})} = \frac{N_{eve}}{N_{per}} = N_N \tag{3.42}$$

y la eficiencia en el uso del recurso ancho de banda $\eta_{BW}(\,\%)$ queda definido por

$$\eta_{BW}(\,\%) = (1 - N_N)100 \tag{3.43}$$

Como consecuencia, la eficiencia en el consumo de ancho de banda en el sistema basado en eventos propuesto es la máxima teórica ya que se cumple en este caso que $N_{BW} = N_N$ y por tanto $\eta_{BW}(\%) = \eta_N(\%)$.

3.4.2. Eficiencia en Recursos Computacionales

Para tener una idea de cómo impacta la estructura de control basada en eventos en el ahorro de recursos de computación, se hará el análisis considerando el modelo de la Figura 3.19.

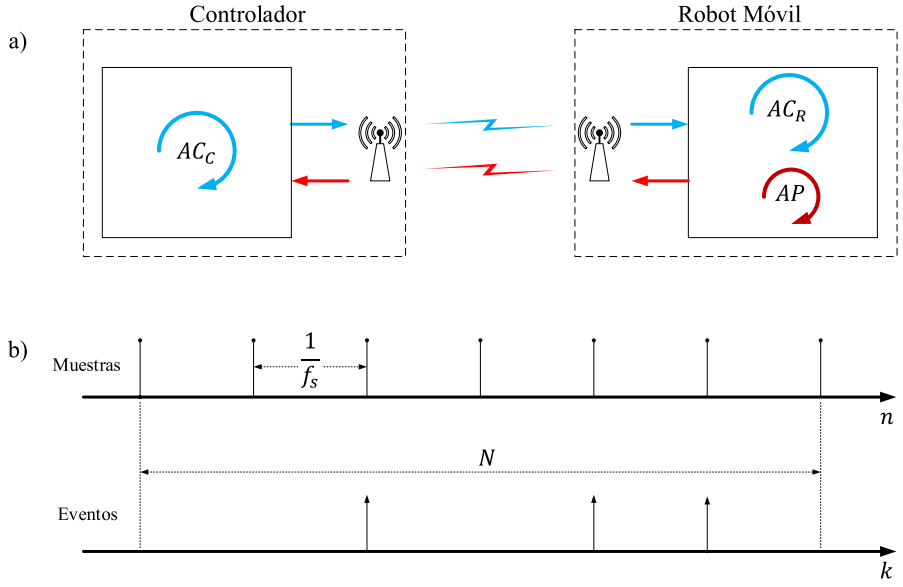

Figura 3.19: Modelo de recursos de computación.

Tal como se observa en la figura, se pueden considerar tres tipos de algoritmos diferentes:

1. *Algoritmo de control en el controlador* AC_C (véase Figura 3.19 a)). Este proceso se invoca desde el robot con el algoritmo AP (algoritmo de *polling*). Cuando estamos en el dominio discreto, cada periodo de muestreo $1/f_s$ el robot envía la información de los sensores y el controlador determina las nuevas velocidades para el robot con el algoritmo AC_C. En el caso de eventos, la filosofía es similar, solo que no se envía de forma periódica sino cuando ocurre un evento en el sistema. Por tanto el algoritmo AC_C solo es invocado cada vez que se produce un evento en el sistema.

2. *Algoritmo de control en el robot* AC_R (véase Figura 3.19 a)). Este algoritmo es el encargado de leer los valores de velocidad enviados por el controlador y

aplicarlos a los motores del robot. Como sucede en el caso del algoritmo de control en el controlador, este algoritmo AC_R se ejecuta de forma periódica en el dominio discreto y en el de eventos solo cuando suceden éstos.

3. *Algoritmo AP* (véase Figura 3.19 a)). Se define como algoritmo de *polling*. Este algoritmo funciona de forma periódica (con el periodo $1/f_s$) en el robot y de igual manera tanto en la solución discreta como en la de eventos. La función de este código es leer los sensores cada periodo de tiempo. En el caso discreto, tras cada lectura se transmite esta información al controlador. Para el caso de eventos, una vez leída esta información, se comprueba la condición de evento y en función del resultado, se transmite o no esta información.

En la Sección 3.3 cuando se definieron los algoritmos de navegación tanto en el dominio discreto como en el de eventos, se vio que ambos tienen aproximadamente una carga computacional similar (tiene que realizar un número de operaciones similares y de igual complejidad). Por otro lado, desde el punto de vista de ahorros de recursos computacionales, la solución basada en eventos ejecutará menos veces los algoritmos AC_C y AC_R estando esta diferencia directamente relacionada con el ratio de eventos N_N. Respecto al algoritmo AP no existe ninguna eficiencia ya que trabaja de igual forma y con la misma periodicidad tanto en el caso discreto como en el de eventos.

Considérese que el algoritmo AC_C necesita ejecutar no_C operaciones para realizar todas las acciones que tiene programadas, que el algoritmo AC_R tiene que ejecutar no_R operaciones y los de *polling* no_{Pd} en el caso discreto y no_{Pe} para el caso de los eventos, siendo $no_{Pd} \leq no_{Pe}$. En el caso discreto, la carga computacional CC_{dis}(op/s) en el periodo de análisis N (véase Figura 3.19 b)) considerando los algoritmos AC_C, AC_R y AP se puede escribir mediante la ecuación:

$$CC_{dis}(\text{op/s}) = \frac{(no_C + no_R + no_{Pd})N_{per}}{N} = \tag{3.44}$$

Teniendo en cuenta ahora la solución basada en eventos, la carga computacional CC_{eve}(op/s) considerando estos mismos algoritmos se puede escribir mediante la expresión:

$$CC_{eve}(\text{op/s}) = \frac{(no_C + no_R)N_{eve} + no_{Pe}N_{per}}{N} \tag{3.45}$$

A partir de las Ecuaciones 3.44 y 3.45, se puede obtener el ratio de actividad empírico para la carga computacional N_{CC} de la forma:

$$N_{CC} = \frac{(no_C + no_R)}{(no_C + no_R + no_{Pd})} N_N + \frac{no_{Pe}}{(no_C + no_R + no_{Pd})} \tag{3.46}$$

De igual manera, se puede definir la eficiencia en el uso de recursos computacionales $\eta_{CC}(\%)$ según la expresión:

$$\eta_{CC}(\%) = (1 - N_{CC})100 \tag{3.47}$$

A partir de las ecuaciones 3.46 y 3.47, se observa que si los términos no_{Pd} y no_{Pe} tienden a cero, el valor de N_{CC} tiende a N_N y la eficiencia en el consumo de recursos computacionales es la máxima teórica $\eta_N(\%)$. A medida que no_{Pd} y no_{Pe} aumentan y dado que $no_{Pd} \leq no_{Pe}$ el término N_{CC} tiende a 1 y por tanto la eficiencia $\eta_{CC}(\%)$ converge hacia cero. Según el resultado anterior, los algoritmos de *polling* serán los que marquen la eficiencia en el uso de los recursos de computación de la solución basada en eventos frente a la discreta.

3.4.3. Ahorro Energético

Respecto al ahorro energético, las estructuras de control basadas en eventos deberían tener un consumo inferior a las arquitecturas de control discretas tradicionales. El hecho de que cuando el sistema se encuentre en estado estacionario no se transmita información por el interfaz inalámbrico y que ciertos algoritmos no estén consumiendo recursos de computación, debería ocasionar un menor consumo energético. Para hacer un análisis de alto nivel del consumo energético, se utilizará el modelo mostrado en la Figura 3.20.

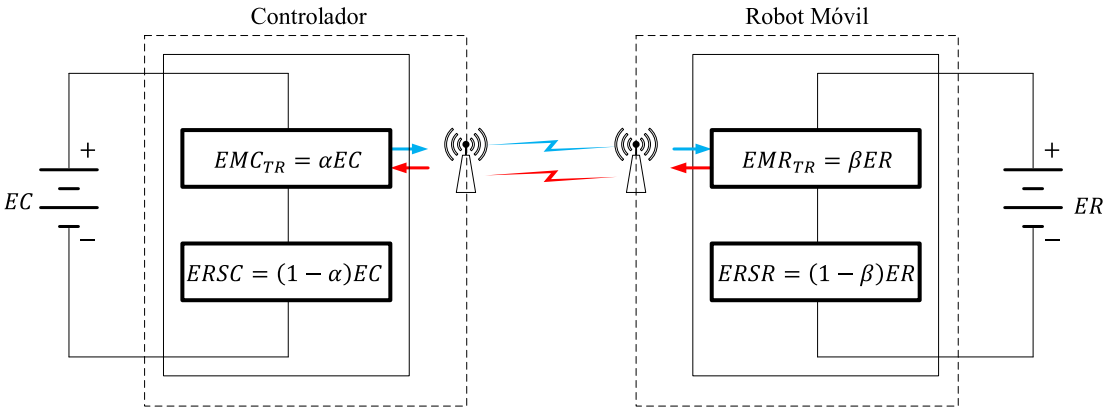

Figura 3.20: Modelo de consumo energético.

Según se muestra en dicha figura, la energía que consume el controlador EC

se compone de la energía que consume el módem del controlador para enviar y recibir información del enlace inalámbrico EMC_{TR} más la consumida por el resto de sistemas del controlador $ERSC$. A su vez, puede definirse el ratio α que marca qué parte de la energía la consume el módem para transmitir y recibir respecto a la energía total consumida por el sistema controlador. Con estas hipótesis, la energía consumida por el controlador tiene la forma:

$$EC = EMC_{TR} + ERSC = \alpha EC + (1 - \alpha)EC \tag{3.48}$$

Análogamente, puede escribirse una ecuación similar para el robot. En este caso, se define la energía consumida por el módem del robot para emitir y recibir información a través del interfaz radio como EMR_{TR} y la energía consumida por los restantes sistemas del robot como $ERSR$. Haciendo un reparto de la energía consumida por el módem respecto al resto de sistemas del robot a través del parámetro β, se puede escribir la ecuación del consumo de energía para el robot según la expresión:

$$ER = EMR_{TR} + ERSR = \beta ER + (1 - \beta)ER \tag{3.49}$$

Si se considera ahora que el controlador y el robot trabajan en tiempo discreto con una frecuencia de muestreo f_s, el consumo de energía del módem lo marcan los periodos de muestreo y el consumo del resto de sistemas es independiente de esta frecuencia. Por tanto, las expresiones del controlador y del robot para el caso discreto pueden escribirse según las Ecuaciones 3.50 y 3.51 respectivamente.

$$EC_{dis} = \alpha ECN_{per} + (1 - \alpha)EC \tag{3.50}$$

$$ER_{dis} = \beta ERN_{per} + (1 - \beta)ER \tag{3.51}$$

Haciendo un razonamiento similar cuando el sistema trabaja según una ley de control basada en eventos, las expresiones del consumo energético del controlador y del robot pueden escribirse según las Ecuaciones 3.52 y 3.53 respectivamente.

$$EC_{eve} = \alpha ECN_{eve} + (1 - \alpha)EC \tag{3.52}$$

$$ER_{eve} = \beta ERN_{eve} + (1 - \beta)ER \tag{3.53}$$

Si como en casos anteriores, se quiere determinar el ratio de actividad empírico para el consumo de energía N_{CE_C} para el controlador, ésta se calcula según:

$$N_{CE_C} = \frac{EC_{eve}}{EC_{dis}} = \frac{\alpha N_{eve} + (1-\alpha)}{\alpha N_{per} + (1-\alpha)} \tag{3.54}$$

En cambio, para el robot, este ratio de actividad N_{CE_R} se expresa como:

$$N_{CE_R} = \frac{ER_{eve}}{ER_{dis}} = \frac{\beta N_{eve} + (1-\beta)}{\beta N_{per} + (1-\beta)} \tag{3.55}$$

De igual forma, se pueden definir las eficiencias en el consumo de energía tanto para el controlador $\eta_{CE_C}(\%)$ como para el robot $\eta_{CE_R}(\%)$ mediante las Ecuaciones 3.56 y 3.57 respectivamente.

$$\eta_{CE_C}(\%) = (1 - N_{CE_C})100 \tag{3.56}$$

$$\eta_{CE_R}(\%) = (1 - N_{CE_R})100 \tag{3.57}$$

Finalmente, analizando las expresiones 3.54 - 3.57 en función de los parámetros α y β los cuales tiene el siguiente rango de valores posibles $0 \leq \alpha \leq 1$ y $0 \leq \beta \leq 1$, se puede concluir lo siguiente. Cuando α tiende a cero, la eficiencia $\eta_{CE_C}(\%)$ también tiende a cero. En cambio, a medida que α tiende a 1 la eficiencia $\eta_{CE_C}(\%)$ tiende a la máxima teórica del sistema basado en eventos $\eta_N(\%)$. Igual razonamiento se puede hacer con la eficiencia en el consumo energético del robot $\eta_{CE_R}(\%)$, pero en este caso con el parámetro β, llegando a los mismos resultados. Como se observa, a medida que el consumo energético del módem tiene más peso en el consumo global del sistema, los sistemas de control basados en eventos propuestos generan un mayor ahorro energético.

3.5. Conclusiones

En este capítulo se ha propuesto una nueva arquitectura de control basada en eventos para la implementación de algoritmos de navegación en robots móviles. El proceso de sintonía de la arquitectura planteada es similar a las estructuras clásicas basadas en control discreto y por tanto es de aplicación directa en entornos de navegación. Con este nuevo esquema se pueden definir de manera sencilla los algoritmos típicos de navegación además de poder garantizar su estabilidad.

También, se propone una implementación en este nuevo esquema de control que garantiza que el sistema es inmune a los efectos tipo Zeno. En este trabajo se han desarrollo tres algoritmos típicos de navegación (Directo al Destino, Evitar Obstáculos y Seguimiento de Paredes) para esta nueva arquitectura, pero el sistema propuesto es lo suficientemente flexible como para incorporar cualquier otro algoritmo de navegación de forma sencilla e intuitiva. Finalmente, se ha obtenido de manera teórica y a través de simulación que la solución planteada presenta grandes eficiencias en el uso de los recursos si se comparan con los esquemas clásicos basados en control discreto.

Capítulo 4

Mecanismos de Ajuste Dinámico de Umbrales

En este capítulo se analizan las diferentes metodologías y estructuras utilizadas para generar eventos en los sistemas de control analizados. Se investiga sobre los efectos que tiene el ruido en la generación de eventos y se analizan las ventajas de tener sistemas con umbrales dinámicos frente a sistemas clásicos con umbrales fijos. Se propone una arquitectura de control que estima de forma dinámica los umbrales en el sistema en función del ruido. Esta arquitectura también puede utilizarse en el caso de que se disponga del modelo de perturbación que afecta a las medidas del sistema. Finalmente, se analiza la respuesta de esta solución en los algoritmos de navegación planteados en este trabajo así como en sistemas más complejos como puede ser un UAV.

4.1. Introducción

Tal como se ha descrito en los capítulos anteriores, la estructura general de un sistema de control basado en eventos tiene una arquitectura como la mostrada en la Figura 4.1 (Åström [12], Cervin and Åström [23]).

El sistema consta de los elementos siguientes:

1. *El proceso.*

2. *El detector de eventos.* Este elemento genera una señal de salida $e_r(t_k)$ que se envía al controlador cuando ocurre un evento en el instante t_k. Un criterio para generar eventos podría ser cuando la señal de error $e_r(t) = y(t) - w(t)$

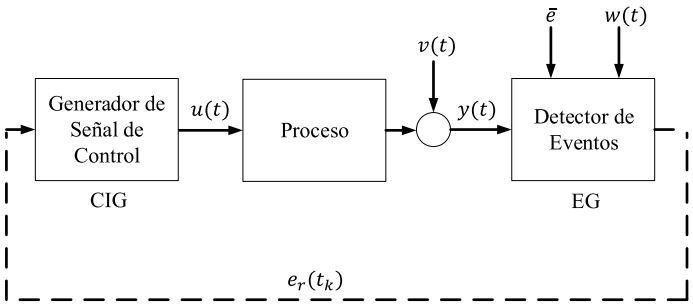

Figura 4.1: Esquema general de un sistema de control basado en eventos.

cruza ciertos niveles (*event threshold* $\bar{e}$). El valor del umbral se selecciona como compromiso entre el número de eventos por unidad de tiempo y el comportamiento del control.

3. *El generador de la señal de control.* Este elemento es el encargado de obtener la señal de entrada al proceso. Este módulo puede considerarse como un retenedor generalizado (Ohno et al. [105]).

Por tanto, partiendo del esquema general de un sistema de control basado en eventos, en este capítulo se hará un análisis profundo del bloque generador de eventos ya que tiene una influencia crucial desde el punto de vista de la estabilidad, prestaciones del sistema y consumo de recursos. Uno de los puntos a tratar es cómo fijar los umbrales de forma óptima para tener una respuesta similar a un sistema discreto, pero sin perder las ventajas que aporta un sistema basado en eventos en lo relativo al mejor uso de los recursos. Se analizarán y se propondrán nuevas estructuras para fijar los umbrales de forma dinámicas además de investigar los efectos que tiene el ruido de medida en estos sistemas.

4.2. Generador de Eventos

El planteamiento de muestreo basado en eventos no es en realidad una idea nueva y sus orígenes se remontan a los años 50 (Ellis [40]). Este autor planteó que el método de muestreo más apropiado consistía en transmitir solamente datos cuando existe un cambio significativo en la señal que justifique la captura de una nueva muestra y se actúe en el sistema con una nueva acción de control.

Para realizar un análisis de estas leyes de muestreo, se considerarán los dos casos siguientes que han recibido una considerable atención en un gran volumen

de trabajos sobre control basado en eventos. Estas dos leyes de control, son por otro lado, los esquemas más naturales en este tipo de estrategias:

1. $\int_{t_k}^{t_{k+1}} \mid e_r(t) - e_r(t_k) \mid dt = \bar{e}$. Esta ley de muestreo fue propuesta por Dorf et al. [33]. Si se aproxima el error $e_r(t)$ por un desarrollo en serie de Taylor truncado en el primer término $e_r(t) \approx e_r(t_k) + \dot{e}_r(t_k)(t - t_k)$ y se sustituye en la ley de muestreo se obtiene la siguiente expresión para el periodo de muestreo T_k:

$$T_k = \frac{\sqrt{2\bar{e}}}{\sqrt{\mid \dot{e}(t_k) \mid}} \tag{4.1}$$

2. $\mid e_r(t) - e_r(t_k) \mid = \bar{e}$. Siguiendo un procedimiento similar de aproximación se obtiene el periodo de muestreo según la expresión:

$$T_k = \frac{\bar{e}}{\mid \dot{e}(t_k) \mid} \tag{4.2}$$

   esta ley se conoce como criterio de diferencias de amplitudes constante (Rodríguez et al. [116], Dormido and Mellado [35]).

En Dormido et al. [34] y de la Sen [30] se ha hecho un estudio donde se unifican todas estas leyes de muestreo y se propone una base analítica a todas ellas. De todos estos esquemas de generación de eventos, hoy en día predominan principalmente las tres estructuras que se describe a continuación.

4.2.1. Generador de Eventos por Cruce de Nivel

En este tipo de detectores la señal de error se muestrea por cruce de nivel. En este caso, los niveles se encuentran equiespaciados y además hay histéresis (los cruces sucesivos por un mismo nivel no provocan muestras) tal como se muestra en la Figura 4.2.

Este tipo de estrategias de muestreo recibe diferentes denominaciones en la literatura: criterio de diferencia de amplitudes constante (Rodríguez et al. [116], Dormido and Mellado [35]), muestreo basado en magnitud (Miśkowicz [89]), modulación delta asíncrona (Inose et al. [59]), método de banda muerta (Otanez et al. [107]), método send-on-delta (Miskowicz [92]), muestreo por cruce de nivel (Sayiner et al. [121]) y muestreo de Lebesgue (Åström and Bernhardsson [13]).

A pesar de sus denominaciones, el principio básico es el mismo: la señal se muestrea cuando el valor absoluto de la diferencia entre el valor actual $e_r(t)$ y

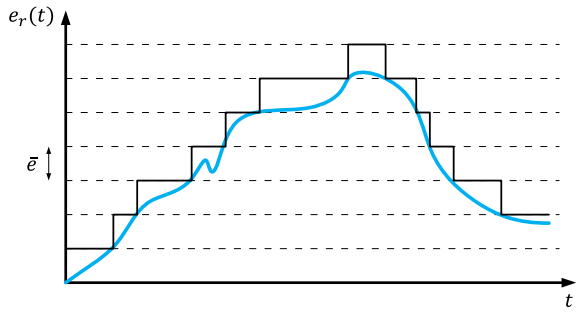

Figura 4.2: Generador de eventos por cruce de nivel.

el último valor muestreado $e_r(t_k)$ es mayor que un límite especificado $\bar{e}$, (véase Figura 4.3 b)). Cuando el cambio en la señal es relativamente pequeño, el número de muestreos es significativamente menor al de un sistema basado en muestreo periódico (véase Figura 4.3 a)).

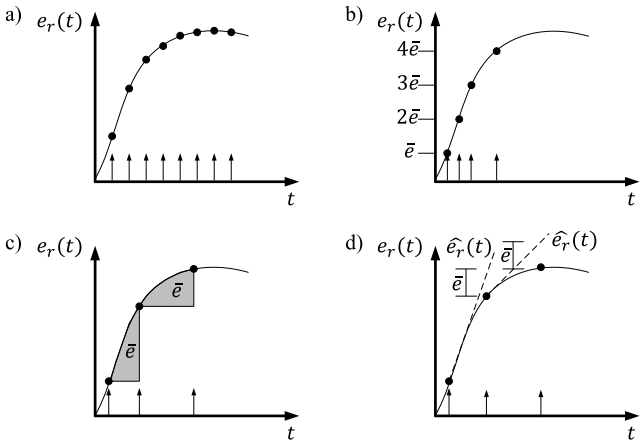

Figura 4.3: Esquemas de muestreo.

4.2.2. Generador de Eventos por Cruce de Nivel Integral

La idea del muestreo basado en eventos por cruce de nivel se puede analizar desde un punto de vista más amplio al muestrear en diferentes dominios de medidas de la señal. Desde el punto de vista más empírico, las medidas de señales más importantes son aquellas que tienen una interpretación física clara (p.e. la integral, la energía, etc.). Utilizando estas medidas para la definición de la generación de los eventos, es posible definir el muestreo integral en el dominio de la energía. Por ejemplo, en los años 60 Dorf et al. [33] propuso una estrategia basada en eventos donde la señal de entrada se muestrea de acuerdo con un criterio integral. La idea para utilizar el muestreo integral es que con asiduidad la integral

del valor absoluto del error (IAE) se toma como índice del comportamiento de un sistema de control (Åström and Hägglund [14]). En el muestreo con el criterio integral (Figura 4.3 c)) la señal $e_r(t)$ se muestrea si la integral del valor absoluto de la diferencia entre el valor actual de la señal y el último valor muestreado $e_r(t_k)$ alcanza un valor umbral $\bar{e}$ dado por: (Miskowicz [93])

$$\int_{t_k}^{t_{k+1}} \mid e_r(t) - e_r(t_k) \mid dt = \bar{e} \tag{4.3}$$

Se pueden resaltar varias ventajas del muestreo integral en aplicaciones de control. En primer lugar en comparación con el muestreo por cruce de nivel da una medida más útil de la calidad en el seguimiento de una señal. Esto es debido a que el criterio integral es una función monótona creciente con el tiempo y como resultado de ello no compensa entre si los errores positivos y negativos. En segundo lugar el muestreo integral da una garantía de que siempre el disparo genere un evento en el futuro, cosa que no se puede garantizar en el muestreo por cruce de nivel donde puede transcurrir un tiempo excesivamente elevado ya que los cambios en la señal no son significativos como para activar la adquisición de una nueva muestra. En ciertos casos, el muestreo integral puede resultar de gran utilidad con medidas indirectas, en particular si las medidas se llevan a cabo en el dominio de la función primitiva, como sucede por ejemplo si la distancia recorrida se estima a través del vector velocidad.

4.2.3. Generador de Eventos por Cruce de Nivel y Predicción Lineal

En el trabajo de Suh [137] se propone un nuevo algoritmo que utiliza un mecanismo de predicción lineal junto con un esquema de muestreo por cruce de nivel que reduce aún más el número de eventos. Su modo de trabajo es el siguiente: la señal se transmite si la diferencia entre su valor actual y el valor predicho es mayor que el *event threshold* $\bar{e}$ especificado (Figura 4.3 d)). El valor predicho se calcula utilizando los valores pasados. Por tanto, el algoritmo por cruce de nivel se podría considerar como un caso especial del algoritmo propuesto en el sentido de que el valor predicho es el último valor transmitido. Se podría esperar que en el método por cruce de nivel, el número de muestreos necesarios sea pequeño cuando en la señal no se producen cambios, esto sucede cuando la primera derivada es casi constante. En el algoritmo propuesto, el número de transmisiones

se reduce cuando las derivadas de orden superior son casi constantes. Puede demostrarse que el nuevo algoritmo reduce el número de transmisiones comparado con el método por cruce de nivel convencional. En la Ecuación 4.4 se muestra el algoritmo para el nodo emisor y en la Ecuación 4.5 el correspondiente al nodo receptor.

$$
\begin{aligned}
&\hat{x}_0 = \cdots = \hat{x}_{-M+1} = 0 \\
&\textbf{for } k = 1 \textbf{ to } N \\
&\quad \textbf{repeat} \\
&\qquad \hat{x}_k = f(\hat{x}_{k-1}, \cdots, \hat{x}_{k-M-1}) \\
&\quad \textbf{until } |x_k - \hat{x}_k| > \bar{e} \\
&\quad \textbf{transmitir } x_k, \cdots, x_{k-M} \\
&\textbf{end}
\end{aligned}
\tag{4.4}
$$

$$
\begin{aligned}
&\hat{x}_0 = \cdots = \hat{x}_{-M+1} = 0 \\
&\textbf{for } k = 1 \textbf{ to } N \\
&\quad \textbf{while dato no recibido} \\
&\qquad \hat{x}_k = f(\hat{x}_{k-1}, \cdots, \hat{x}_{k-M-1}) \\
&\quad \textbf{end} \\
&\quad \hat{x}_k = x_k, \cdots, \hat{x}_{k-M} = x_{k-M} \\
&\textbf{end}
\end{aligned}
\tag{4.5}
$$

En el nodo sensor se supone que se muestrea con periodo T_s. Una señal de tiempo discreto x_k se define por $x_k = x(kT_s)$. Aunque el nodo emisor se muestrea con período T_s, la velocidad de transmisión no es T_s ya que no se transmiten todas las muestras. En el nodo emisor, $\hat{x}_k = f(\hat{x}_{k-1}, \cdots, \hat{x}_{k-M-1})$ se trata de un predictor lineal, donde M es la longitud de la memoria. Si $M = 1$, $\hat{x}_k$, se calcula en función de $\hat{x}_{k-1}$ y $\hat{x}_{k-2}$. Es importante destacar que se utiliza $f(\hat{x}_{k-1}, \cdots, \hat{x}_{k-M-1})$ y no $f(x_{k-1}, \cdots, x_{k-M-1})$ como cabría esperar. Si se utilizase $f(x_{k-1}, \cdots, x_{k-M-1})$, se obtendría $\hat{x}_k$ de forma más exacta, sin embargo esto obligaría a transmitir x_k al receptor cada T_s segundos lo que supondría que no se puede reducir el número de transmisiones. Si la diferencia entre el valor actual x_k y el valor predicho $\hat{x}_k$ es mayor que $\bar{e}$ se transmite $x_k, \cdots, x_{k-M}$ en lugar de transmitir x_k tal como se hace en el método de cruce por nivel.

4.3. Umbrales Fijos vs. Umbrales Dinámicos

En los métodos mostrados anteriormente y en la gran mayoría de las publicaciones sobre metodologías de control basado en eventos, se consideran umbrales de disparo fijos, que una vez definidos se mantienen invariables. Uno de los inconvenientes de usar umbrales fijos es que la velocidad de cruce de los mismos depende de la magnitud de cambio del valor de referencia o de la perturbación a compensar. De esta manera, a mayor magnitud de la entrada (cambios de referencia o perturbaciones) el cruce de los umbrales se hará a más velocidad, incrementado de forma considerable el número de eventos y por tanto el envío de datos y generación de nuevas acciones de control. Este problema ha sido abordado en Durand and Marchand [38] donde se propone establecer un tiempo mínimo entre muestras para evitar eventos consecutivos innecesarios. Esta solución sin embargo solo resuelve parcialmente el problema, añadiendo un nuevo parámetro cuya selección no responde a unas reglas claras.

En el trabajo Romero et al. [117] se propone un algoritmo de control PI basado en eventos mediante cruce de nivel con umbrales adaptables. El algoritmo está centrado en el rechazo de perturbaciones y su principal ventaja es que se reduce significativamente la dependencia entre la magnitud de la perturbación a corregir y el número de eventos generados como consecuencia del cruce de niveles. Esto da como resultado una reducción importante de la cantidad de eventos respecto de controladores con umbrales fijos, sin perjuicio del comportamiento del sistema controlado. Ideas similares se presentan en el trabajo de Molin and Hirche [94].

Se han investigado otras ideas considerando en este caso las perturbaciones o el ruido que afecta al sistema en el trabajo de Socas et al. [127]. En este caso se hace una estimación dinámica del ruido que afecta al sistema y basado en el valor de esta perturbación, se va actualizando de forma dinámica el *event threshold* $\bar{e}$. Con esta metodología se reduce de forma considerable el número de eventos que genera el sistema manteniendo unas prestaciones similares al caso de utilizar un umbral fijo.

4.4. Efectos del Ruido en la Generación de Eventos

Para estudiar el Generador de Eventos con umbrales dinámicos propuesto en este trabajo, se presentará previamente un caso sencillo para ilustrar como afecta el ruido a un sistema de control basado en eventos con umbrales fijos.

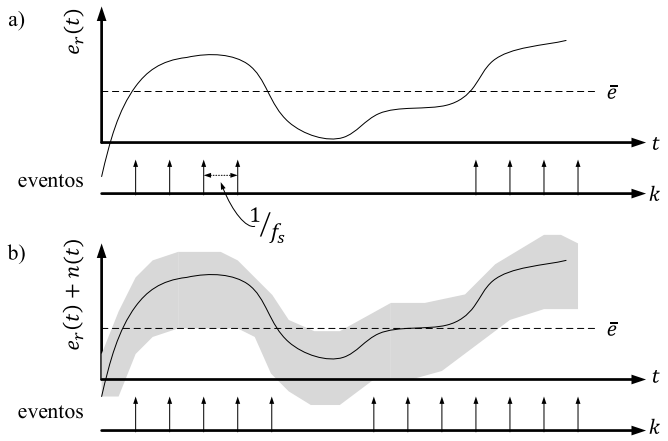

Figura 4.4: Efecto del ruido en la generación de eventos.

Analizando la Figura 4.4, se considera la señal de error $e_r(t)$ como la magnitud que determina cuando se producirá un evento en el sistema y la condición de evento una regla de la forma $e_r(t) > \bar{e}$, siendo además el *event threshold* $\bar{e}$ del sistema un valor fijo. Tal como se muestra en la Figura 4.4 a), cuando la señal de error supera el umbral $\bar{e}$, el sistema genera un evento. En este caso se está considerando que este sistema está implementado en un sistema discreto nativo como sucede en nuestro sistema de experimentación, con frecuencia de muestreo f_s. Por tal razón, siempre que esté la condición de evento activa, se estarán produciendo eventos con un periodo $1/f_s$.

Se considera ahora que la señal de error está contaminada por ruido aditivo $e_r(t) + n(t)$ (Figura 4.4 b)). Si el sistema está trabajando con el mismo *event threshold* $\bar{e}$ estático, a causa del ruido se generarán muchos más eventos que los que el sistema original produciría. Por tanto, se observa que el ruido provoca que el sistema genere muchos más eventos lo que penaliza en gran medida la eficiencia en el uso de recursos.

4.5. Ruido en los Algoritmos de Navegación basados en Eventos

Para analizar el efecto del ruido en los algoritmos de navegación con control basado eventos propuestos en el Capítulo 3, se hará un análisis del algoritmo Directo al Destino (DD) considerando que los sensores de distancia recorrida del robot están contaminados por ruido. Tal como se mostró en la Sección 3.3.2, el algoritmo DD está descrito por los parámetros del controlador K_ρ y K_α, la condición de parada e_ρ y el *event threshold* $\bar{e}_\alpha$ tal como se muestra en la Figura 4.5.

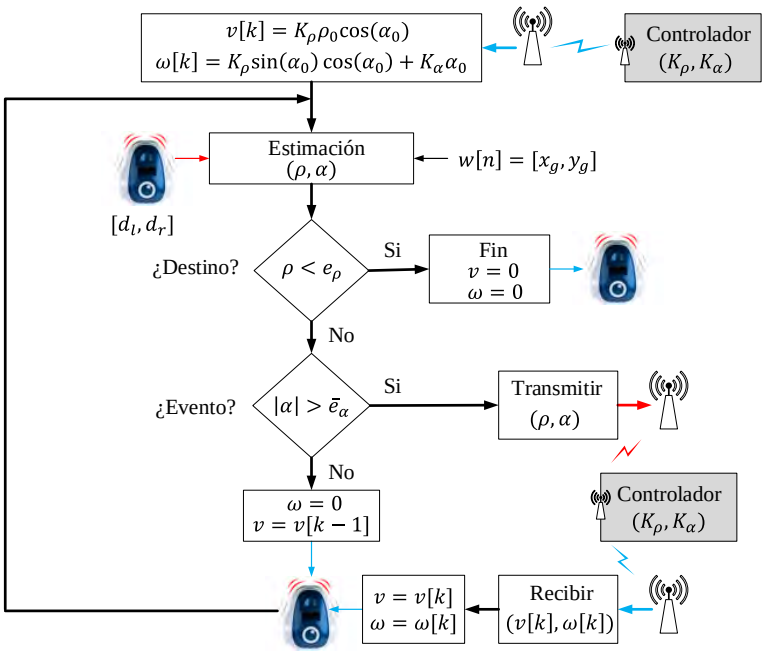

Figura 4.5: Algoritmo DD en el sistema de control basado en eventos propuesto.

Para mostrar los efectos del ruido sobre esta arquitectura de control, se analizará una simulación como la ilustrada en el Figura 4.6.

En este caso, el controlador basado en eventos tiene programado el algoritmo DD para guiar el robot desde la posición inicial $(0, 0)$ hasta la posición $x_g = 50$ cm e $y_g = -50$ cm. Por otro lado, se considerarán dos escenarios: el primero estará libre de ruido y en el segundo se incorporará ruido en los sensores de distancia recorrida. Los sensores de distancia recorrida obtienen en cada periodo de muestreo (en este análisis $f_s = 10$ Hz) una estimación de la distancia recorrida por cada una de las ruedas, d_l para la rueda izquierda y d_r para la derecha. Además,

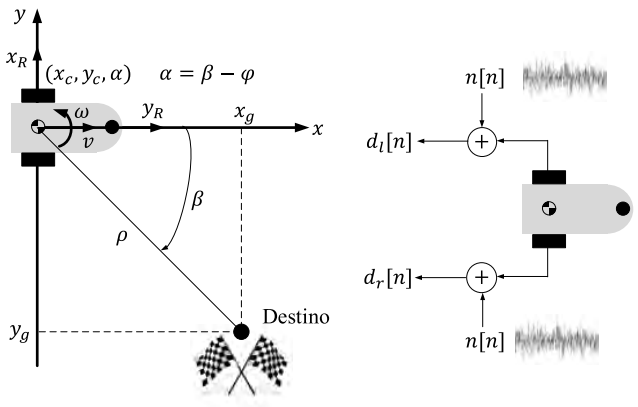

Figura 4.6: Escenarios de análisis del algoritmo DD con ruido.

se contaminan estas medidas en el escenario 2 con un ruido modelado según una distribución normal $\mathcal{N}(\mu, \sigma)$ parametrizada por su media μ y su desviación típica σ. En la Tabla 4.1 se muestran los parámetros de los escenarios comentados.

Tabla 4.1: Escenarios de análisis para los algoritmo DD con y sin ruido de medida.

Escenario	K_ρ(1/s)	K_α(1/s)	$\bar{e}_\alpha$ (rad)	e_ρ (cm)	μ (cm)	σ (cm)
1	0,1	0,5	0,05	0,5	0	0
2	0,1	0,5	0,05	0,5	0	0,08

En los escenarios analizados, el controlador de eventos propuesto en este trabajo para implementar el algoritmo DD tiene las respuestas mostradas en la Figura 4.7. Como se observa, en ambas situaciones el algoritmo resuelve de forma satisfactoria el problema de navegación. El robot parte de la situación inicial y alcanza aproximadamente el objetivo (x_g, y_g) sin inestabilidades. En el Escenario 1, los sensores estiman las distancias recorridas por cada una de las ruedas con el error que las técnicas basadas en odometría introducen en el sistema. En cambio, para el Escenario 2, aparte de este error inherente a la metodología de estimación, se introduce un ruido aditivo $n[n]$ de distribución normal $\mathcal{N}(\mu, \sigma)$ que provoca un mayor error en el sistema. Por tanto, en el escenario con ruido, la posición final (x_g, y_g) distará aún más de la ideal ya que se suma al efecto de la odometría el ruido y esto contribuye a una peor estimación de la posición del robot. En la Figura 4.8, se muestran las medidas de los sensores considerando estos errores para ambos escenarios.

Si se considera el número de eventos generados en cada escenario, se observa que el Escenario 2 genera un número de eventos muy superior al Escenario 1 tal

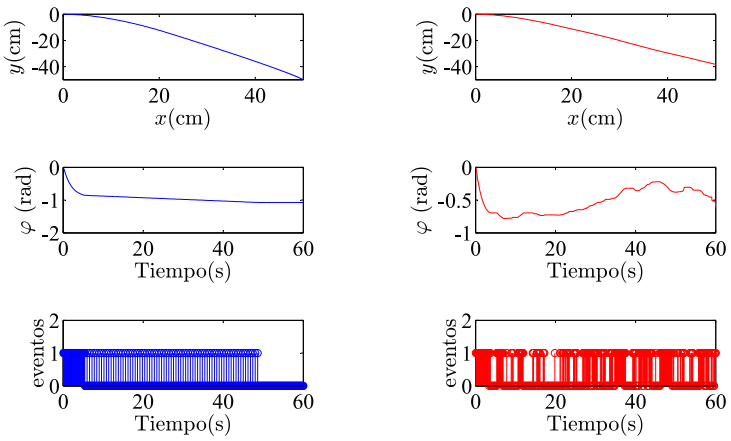

Figura 4.7: Respuesta del algoritmo DD. En azul escenario 1 y en rojo escenario 2.

como se muestra en la Figura 4.9. Al igual que sucedía en el caso genérico presentado en la Sección 4.4, el controlador en el Escenario 2 (sensores contaminados por ruido) genera un número de eventos superior al escenario sin ruido.

Por tanto, se puede concluir que, las perturbaciones y el ruido en los sistemas de control basados en eventos y en concreto en las arquitecturas propuestas en este trabajo, afectan de forma directa al número de eventos generado por el sistema y como consecuencia a sus prestaciones.

4.6. Nuevo Generador de Eventos con Umbrales Dinámicos

Tal como se mostró en la Sección 4.4, el ruido en el sistema afecta de forma directa a la generación de eventos y como consecuencia a las eficiencias en el uso de recursos (transmisión, computacionales, etc.) que estas arquitecturas de control presentan. Dicho de otra forma, a medida que el ruido en el sistema aumenta, las arquitecturas de control basadas eventos pasan a comportarse como estructuras de control discreto y por tanto las ventajas de estas nuevas soluciones se diluyen.

Para compensar los efectos del ruido y la pérdida de prestaciones anteriormente descrita, en este trabajo de investigación se propone una arquitectura para el Generador de Eventos con umbrales dinámicos. La solución propuesta trata de compensar en la medida de lo posible los efectos adversos que el ruido produce en la excesiva generación de eventos.

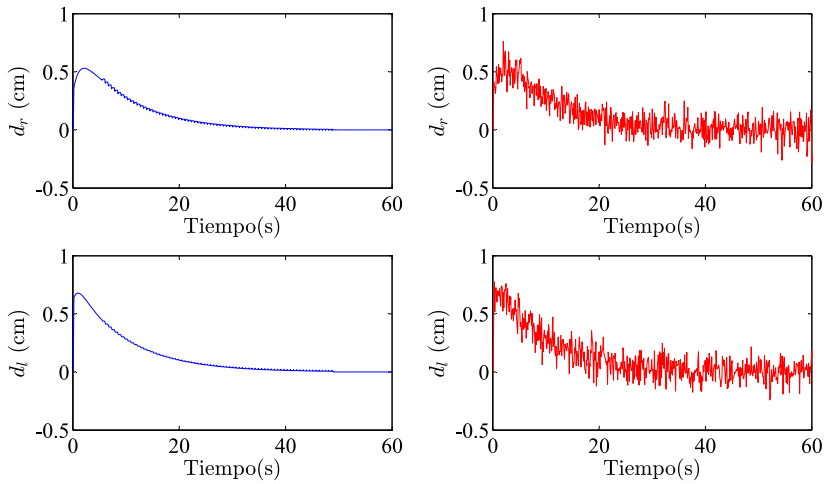

Figura 4.8: Medidas de los sensores de distancia. En azul escenario 1 y en rojo escenario 2.

La propuesta para el generador de eventos basado en umbrales dinámicos se presenta en la Figura 4.10. El nuevo sistema se implementa en el dominio discreto con una frecuencia de muestreo f_s, frecuencia en la que trabajan de forma nativa los robots utilizados en los experimentos de este trabajo. En este nuevo esquema de control, el bloque que se incorpora, el calculador de umbral, determina un *event threshold* $\bar{e}[n]$ dinámico que es función de la señal de salida del sistema y del *event threshold* estático $\bar{e}$ que se fija en el sistema considerando una situación ideal sin presencia de ruido.

El modo de trabajo de este bloque es el siguiente:

1. A partir de la señal de salida del sistema $y[n]$, se hace una estimación del ruido $n[n]$ que contamina las medidas. Es esta estimación de ruido la que se utilizará como la parte variable $\bar{e}_d[n]$ del umbral dinámico $\bar{e}[n]$.

2. Posteriormente para obtener el umbral dinámico $\bar{e}[n]$ que alimenta al Generador de Eventos, basta sumar la parte estática $\bar{e}$ a la parte dinámica $\bar{e}_d[n]$ obtenida anteriormente, expresándose como:

$$\bar{e}[n] = \bar{e} + \bar{e}_d[n] \tag{4.6}$$

La idea general del sistema es incrementar el *event threshold* estático $\bar{e}$ en un valor aproximado al nivel de ruido y evitar así los falsos positivos que el ruido produce. Este nuevo valor es el $\bar{e}_d[n]$. Esta idea se presenta en la Figura 4.11.

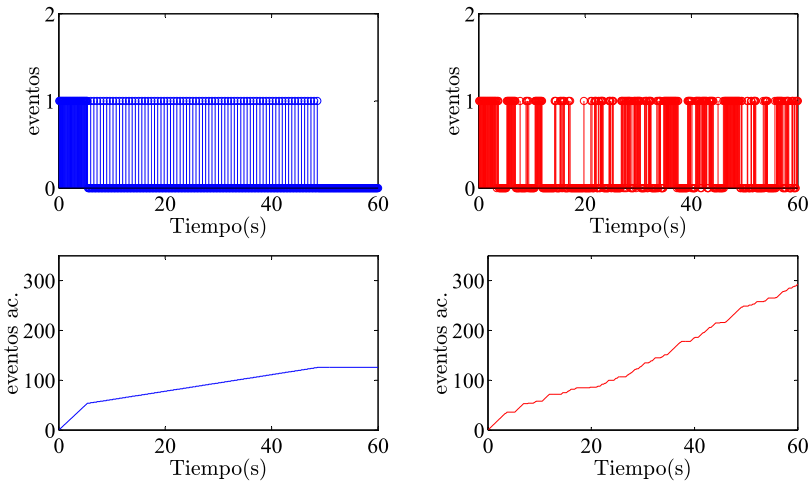

Figura 4.9: Volumen de eventos. En azul escenario 1 y en rojo escenario 2.

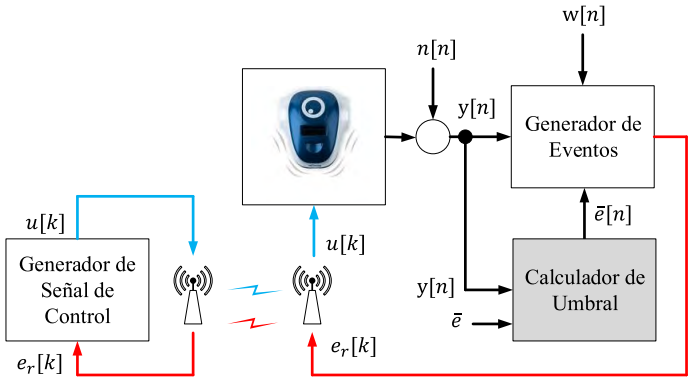

Figura 4.10: Generador de eventos basado en umbrales dinámicos.

Tal como se ilustra en la Figura 4.11 a), una vez que se tiene fijado un $\bar{e}$ estático y definiendo una condición de evento sencilla como la mostrada en la Ecuación 4.7, cuya filosofía es que siempre que la señal $e_r[n]$ supera el umbral se genera un evento. Se supone que el umbral estático ha sido elegido considerando el sistema en condiciones ideales (sin ruido).

$$\begin{aligned} &\textbf{if } abs(e_r[n]) > \bar{e} \\ &\quad \{evento = true\} \\ &\textbf{else} \\ &\quad \{evento = false\} \\ &\textbf{end} \end{aligned} \tag{4.7}$$

Si ahora este sistema está contaminado por ruido y se mantiene el mismo

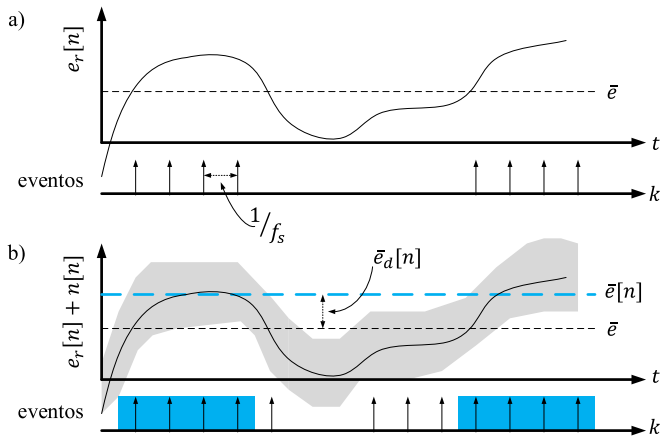

Figura 4.11: Estimación del umbral dinámico.

umbral $\bar{e}$, el número de eventos se incrementa de forma considerable. Por otro lado, si se aplica un nuevo umbral dinámico $\bar{e}[n]$ según la Ecuación 4.6, el sistema ahora solo generará los eventos sombreados en azul en la Figura 4.11 b). Si se analiza este nuevo resultado, se observa que el número de eventos que genera en esta nueva situación el sistema con el mecanismo de umbrales dinámicos, se puede aproximar al que genera el sistema cuando está libre de ruido. Por tanto, si se consigue obtener la componente dinámica del umbral $\bar{e}_d[n]$ con un valor similar al del ruido que contamina el sistema, se puede hacer que éste sea inmune a estas perturbaciones en cuanto a la generación de eventos.

La forma de obtener el umbral dinámico $\bar{e}[n]$ que limita el número de eventos en entornos ruidosos se obtiene mediante la Ecuación 4.6. En este caso existen dos alternativas para obtenerlo:

1. Si la dinámica del sistema no es demasiado rápida, se puede plantear un estimador sencillo que determine el ruido que perturba el sistema, y de esta forma obtener el término $\bar{e}_d[n]$.

2. Cuando la distribución del ruido que perturba el sistema sea conocida se puede obtener de forma sencilla el termino dinámico del *event threshold*. Sobre este aspecto se mostrarán varias metodologías en el Capítulo 5 para modelar estas perturbaciones en los robots móviles bajo estudio en este trabajo y poder fijar de forma más precisa los umbrales.

En la siguiente sección se presenta la forma de implementar el calculador de umbral considerando los dos aspectos anteriores.

4.6.1. Calculador de Umbral

Tal como se analizó anteriormente, (véase Ecuación 4.6), el umbral dinámico lo componen dos términos:

1. El término estático $\bar{e}$, que se determina según el diseño del sistema y considerando un entorno ideal libre de perturbaciones. A medida que el sistema trabaje en una situación más parecida a la ideal en cuanto a ausencia de ruido, este término será el que gobierne la generación de eventos.

2. El término dinámico $\bar{e}_d[n]$, que se plantea estimarlo de dos forma diferentes:

   a) Si el sistema tiene una dinámica no excesivamente rápida y además se considera que el ruido es aditivo y con una distribución normal $\mathcal{N}(0, \sigma)$, puede utilizarse un estimador de ruido como término dinámico $\bar{e}_d[n]$ del *event threshold*.

   b) Si se tiene modelada la perturbación que afecta al sistema $P[n]$, este será el término dinámico del *event threshold* $\bar{e}_d[n] = P[n]$. Las metodologías para estimar las perturbaciones $P[n]$ que afectan a los robots analizados en este trabajo, así como la forma de fijar el umbral, se presentarán en Capítulo 5.

En la Figura 4.12 se presenta la arquitectura propuesta como calculador de umbral. En este caso, se presenta el esquema general con las dos entradas $y[n]$ y $P[n]$. En este capítulo se desarrollará la primera parte, el estimador de umbral (bloque sombreado), dejando para el Capítulo 5 el análisis del módulo basado en el modelo de perturbación.

El estimador de umbral propuesto tiene los siguientes elementos:

1. Una memoria de longitud l. En este dispositivo cada vez que llega una muestra de la salida $y[n]$ se almacena este valor y se descarta el valor más antiguo. Por tanto, el sistema mantiene almacenado los valores de la salida en el registro $y_l[n]$, que contiene las últimas l muestras de la señal $y[n]$.

2. Posteriormente, se calcula la media de este registro $y_l[n]$, obteniéndose una media móvil de longitud l de la salida. La operación para obtener $\bar{y}_l$ está dada por:

$$\bar{y}_l = mean(y_l[n]) = \frac{1}{l} \sum_{i=1}^{l} y_l[i] \tag{4.8}$$

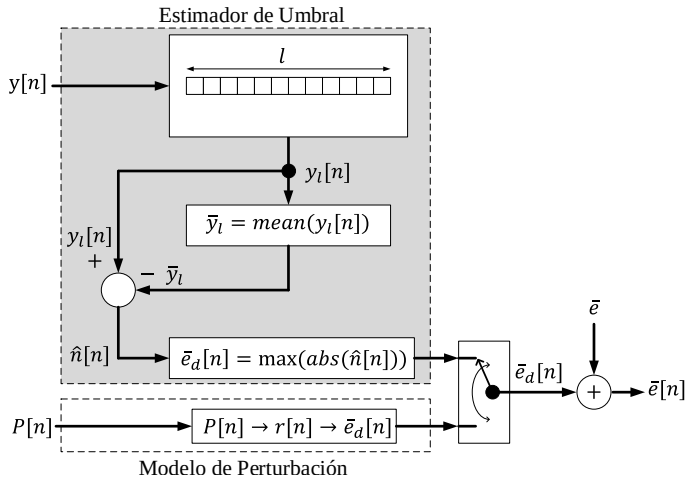

Figura 4.12: Calculador de umbral propuesto.

3. Dado que se ha supuesto un ruido con distribución normal $\mathcal{N}(0, \sigma)$ con valor medio 0, una estimación aproximada del ruido $\hat{n}[n]$ puede ser la diferencia entre la señal almacenada en el registro y la media móvil de la señal de salida $\bar{y}_l$, de forma que $\hat{n}[n] = y_l[n] - \bar{y}_l$.

4. Finalmente, el nivel máximo de ruido, y por tanto la componente dinámica del *event threshold* $\bar{e}_d[n]$ se puede determinar mediante

$$\bar{e}_d[n] = max\left(abs(\hat{n}[n])\right) \tag{4.9}$$

El esquema propuesto dispone de un parámetro l que es preciso definir. Este valor define la longitud de la media móvil que se utiliza para estimar el ruido aditivo que afecta al sistema. El criterio de sintonía es el siguiente: si la dinámica es muy rápida el parámetro l debe ser pequeño. En cambio, si la dinámica es lenta, el parámetro l puede ser mayor y por tanto será más precisa su estimación.

4.6.2. Generador de Eventos con Umbrales Dinámicos en Algoritmos de Navegación para Robot Móviles

Tal como se analizó en las secciones 3.3.2 y 4.5, el algoritmo de navegación DD es una estructura de control basada en eventos que tiene una condición de evento marcada por el *event threshold* $\bar{e}_\alpha$. Por tanto, la variable a analizar es el error de orientación del robot $\alpha[n]$. Si se adapta el calculador de umbral de la Figura 4.12 a este nuevo algoritmo de navegación se obtiene el esquema que se

muestra en la Figura 4.13 (véase también el trabajo Socas et al. [129] para más detalles).

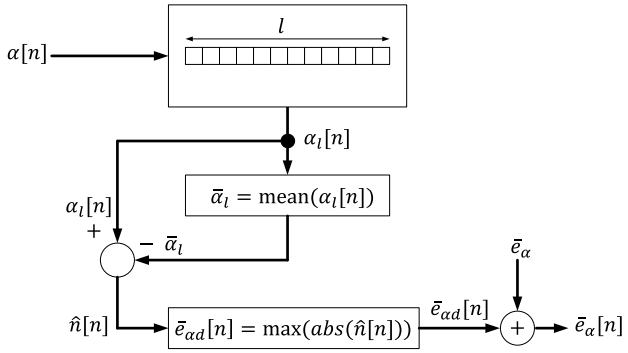

Figura 4.13: Calculador de umbral para el algoritmo DD.

Para analizar las prestaciones del sistema con umbrales dinámicos, el algoritmo mostrado en la Figura 4.5 se hace trabajar tanto con un $\bar{e}_\alpha$ estático como con uno dinámico obtenido según la estructura presentada en la Figura 4.13. Para ello, se analizan los dos escenarios mostrados en la Tabla 4.2.

Tabla 4.2: Escenarios de análisis del algoritmo DD con diferentes niveles de ruido .

Escenario	K_ρ(1/s)	K_α(1/s)	$\bar{e}_\alpha$ (rad)	e_ρ (cm)	μ (cm)	σ (cm)	l
1	0,1	0,5	0,05	0,5	0	0,08	10
2	0,1	0,5	0,05	0,5	0	0,12	10

En cada uno de los escenarios, se harán dos experimentos:

1. El primer experimento consiste en hacer trabajar el algoritmo DD con un umbral estático cuyo valor es el $\bar{e}_\alpha$ mostrado en la Tabla 4.2.

2. En el segundo experimento, se activará el calculador de umbral utilizando también como valor estático del *event threshold* el mismo valor de $\bar{e}_\alpha$ mostrado en esa misma tabla .

En ambos escenarios (Tabla 4.2) se han escogido los mismo parámetros K_ρ y K_α para el controlador. También, el *event threshold* $\bar{e}_\alpha$ escogido es el mismo en los dos escenarios. Para el caso que esté activo el calculador de umbral, éste será modificado según el nivel de ruido del sistema. Por último, la condición de parada e_ρ al igual que el parámetro l también permanece constantes en ambos escenarios.

En las Figuras 4.14 y 4.15 se muestran los resultados obtenidos para el Escenario 1. Para el Escenario 2, los resultados se presentan en las Figuras 4.16 y 4.17.

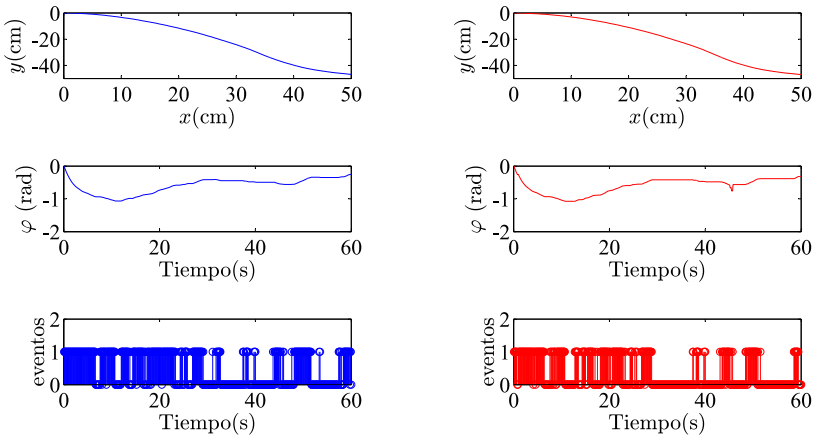

Figura 4.14: Escenario 1: Posicionamiento, orientación y número de eventos. Trazo azul con umbral estático y trazo rojo con calculador de umbral.

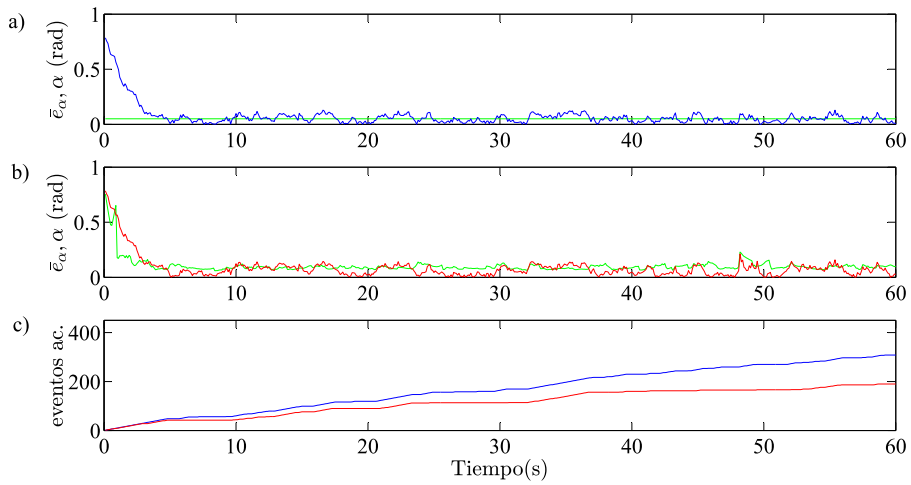

Figura 4.15: Escenario 1: Error de orientación (α), umbral ($\bar{e}_\alpha$) y eventos acumulados. Trazo azul con umbral estático y trazo rojo con calculador de umbral.

En el Escenario 1 se observa que algoritmo de navegación DD resuelve el problema sin inestabilidades (Figura 4.14). En este escenario, tanto con el uso del umbral estático (trazos azules) como con el calculador de umbral activo (trazos rojos) el robot alcanza aproximadamente su destino (posiciones x e y) salvo por los errores de odometría y ruido que afectan a la estimación de su posición. Por

otro lado, analizando los resultados de la Figura 4.15 para el Escenario 1, se observa que:

- Cuando el calculador de umbral está activo, el *event theshold* se va modificando en función del ruido que se estime en cada momento (Figura 4.15 b)). Se observa también, que en los casos en que la dinámica del sistema es más rápida (cambios bruscos del error de orientación α), este umbral no estima bien el nivel de ruido.

- Desde el punto de vista de los eventos generados por el sistema, se observa en la Figura 4.15 c) que cuando está activo el calculador de umbral, el número de eventos es muy inferior al caso de un sistema con umbral estático.

Para el caso de Escenario 2 de la Tabla 4.2, se ha incrementado el nivel de ruido respecto al Escenario 1. En esta situación, ambos sistemas (con umbral estático y con umbral dinámico) resuelven el problema de navegación también de forma satisfactoria (véase Figura 4.16).

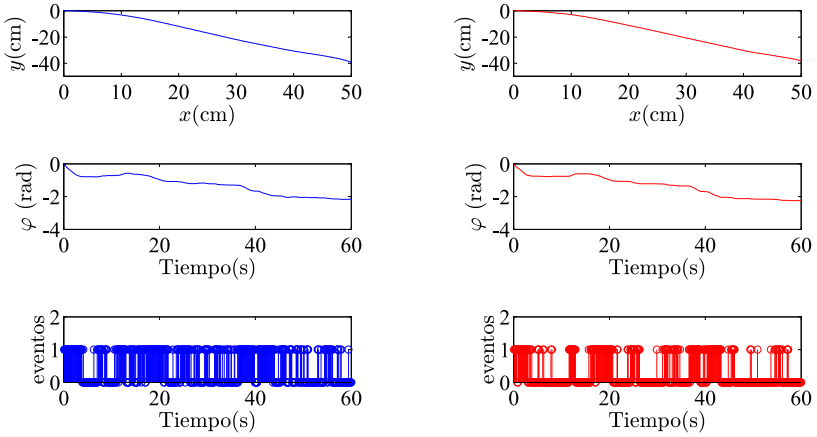

Figura 4.16: Escenario 2: Posicionamiento, orientación y número de eventos. Trazo azul con umbral estático y trazo rojo con calculador de umbral.

Por último, para analizar el efecto del incremento de ruido en la generación de eventos se deben comparar las Figuras 4.15 c) y 4.17 c). Contrastando ambos resultados se observa lo siguiente:

1. Al aumentar el ruido del Escenario 1 al 2, el número de eventos en el sistema con umbrales estáticos aumenta. Obsérvese el trazo azul de la Figura 4.15 c) frente al trazo azul de la Figura 4.17 c).

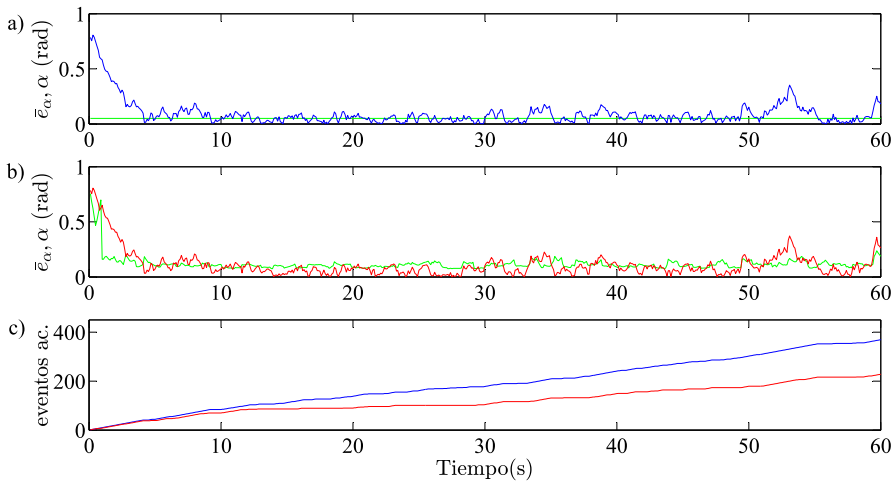

Figura 4.17: Escenario 2: Error de orientación (α), umbral ($\bar{e}_\alpha$) y eventos acumulados. Trazo azul con umbral estático y trazo rojo con calculador de umbral.

2. Respecto a la actividad del controlador, en el sistema de umbrales dinámicos el volumen de eventos permanece constante en ambos casos incluso con un incremento de ruido de un escenario al otro. Compárese el trazo rojo de la Figura 4.15 c) frente al trazo rojo de la Figura 4.17 c).

Por tanto, como conclusión general se puede indicar que:

1. El incluir el calculador de umbral dentro del sistema de control basado en eventos no modifica la respuesta del sistema. Esta nueva arquitectura tiene una respuesta similar al controlador basado en eventos con umbrales estáticos.

2. Respecto a la generación de eventos, este nuevo módulo reduce de forma considerable la generación de eventos en entornos contaminados por ruido y por tanto, mejora de forma considerable la eficiencia del controlador.

4.6.3. Generador Eventos con Umbrales Dinámicos en Plantas Complejas

El calculador de umbral desarrollado en este trabajo tiene como principal objetivo el evitar que se dispare el número de eventos en entornos con ruido. En los algoritmos de navegación de los robots móviles, se ha mostrado que su aplicación presenta grandes ventajas. Por otro lado, el autor de este trabajo,

ha realizado experiencias de aplicación de estos calculadores de umbrales sobre plantas de mayor complejidad, consiguiendo también resultados satisfactorios. A continuación se muestra una breve exposición de los resultados obtenidos al aplicar estas ideas sobre plantas complejas como los UVAs y que pertenecen al trabajo de investigación Socas et al. [127].

En este caso se ha diseñado un piloto automático para el control longitudinal del UAV *Aerosonde*. Esta aeronave es un sistema complejo con seis grados de libertad. Su modelo no lineal se describe mediante doce variables: Posición inercial (p_n, p_e, p_d), las velocidades respecto al sistema de referencia situado en el centro de gravedad de la aeronave (u, v, w), los ángulos de Euler (ϕ, θ, ψ) y las velocidades angulares respecto a su centro de gravedad (p, q, r). Las variables de control en este sistema son la configuración de las superficies de control (alerones δ_a, timón de profundidad δ_e y el timón de dirección δ_r) y el nivel de potencia del motor (δ_t).

Para el guiado de esta aeronave, las tres variables principales que se deben controlar a lo largo del tiempo son:

1. Velocidad longitudinal $u(t)$.
2. Altitud $h(t)$, que en el modelo descrito anteriormente coincide con $h(t) = -p_n(t)$.
3. Rumbo $\psi(t)$.

Los movimientos longitudinales de este UAV están descritos por las variables de velocidad longitudinal $u(t)$ y la altitud $h(t)$, para los movimientos laterales es el rumbo $\psi(t)$ quien describe estos movimientos (Cook [28], Beard and McLain [17]).

Las ecuaciones de movimiento del *Aerosonde* son un conjunto de doce ecuaciones diferenciales acopladas de primer orden. Como primer paso para diseñar el controlador, es usual linealizar y desacoplar estas ecuaciones en un modelo de estados más acorde para estas tareas de diseño. En este caso, se obtiene un modelo lineal MIMO (Ecuación 4.10) que describe los movimientos longitudinales en el punto de equilibrio (altitud constante, planos nivelados y velocidad de 250 m/s). En esta situación, el timón de profundidad está a $\delta_{e0} = -6,26°$ y los gases a $\delta_{t0} = 33,45\,\%$

$$\begin{pmatrix} \dot{u} \\ \dot{w} \\ \dot{q} \\ \dot{\theta} \\ \dot{h} \end{pmatrix} = \mathbf{A}_{lo} \begin{pmatrix} u \\ w \\ q \\ \theta \\ h \end{pmatrix} + \mathbf{B}_{lo} \begin{pmatrix} \delta_e \\ \delta_t \end{pmatrix} \tag{4.10}$$

donde $\mathbf{A}_{lo}$ y $\mathbf{B}_{lo}$ tienen los valores mostrados en las Ecuaciones 4.11 y 4.12 respectivamente.

$$\mathbf{A}_{lo} = \begin{pmatrix} -0{,}52 & 0{,}47 & -2{,}05 & -9{,}77 & 0{,}00 \\ -0{,}60 & -2{,}28 & 24{,}92 & -0{,}80 & 0{,}00 \\ 0{,}04 & -0{,}55 & -0{,}50 & 0{,}00 & 0{,}00 \\ 0{,}00 & 0{,}00 & 1{,}00 & 0{,}00 & 0{,}00 \\ -0{,}08 & 0{,}99 & 0{,}00 & -25{,}00 & 0{,}00 \end{pmatrix} \tag{4.11}$$

$$\mathbf{B}_{lo} = \begin{pmatrix} -0{,}48 & 40{,}64 \\ 5{,}79 & 0{,}00 \\ -18{,}24 & 0{,}00 \\ 0{,}00 & 0{,}00 \\ 0{,}00 & 0{,}00 \end{pmatrix} \tag{4.12}$$

En el diseño del piloto automático para controlar los movimientos longitudinales del UAV se han tenido en cuenta como señales de control $u(t)$ la deflexión del timón de profundidad y el nivel de gases del motor $u(t) = (\delta_e(t) \quad \delta_t(t))^T$. Como salidas del sistema $y(t)$ se han considerado la velocidad longitudinal y la altitud $y(t) = (u(t) \quad h(t))^T$. Las señales de consigna son la velocidad longitudinal $u_r(t)$ y la altitud $h_r(t)$. Por otro lado, las perturbaciones en el sistema son un ruido aditivo $v(t)$ en los sensores y una perturbación $d(t)$ que afecta a la planta (véase Figura 4.18).

En la implementación de este piloto automático se han analizado dos realizaciones: La primera con un umbral estático (Figura 4.18 a)) y la segunda con umbrales dinámicos donde se aplican las ideas desarrolladas en este trabajo Figura 4.18 b). En este documento solo se describirán el generador de eventos y el calculador de umbral ya que son los que presentan interés de cara al análisis de umbrales dinámicos. Para obtener más detalles del sistema como puede ser el diseño del CIG, etc. revisar el trabajo Socas et al. [127]. El generador de eventos propuesto en este sistema se muestra en la Figura 4.19. Como se observa en

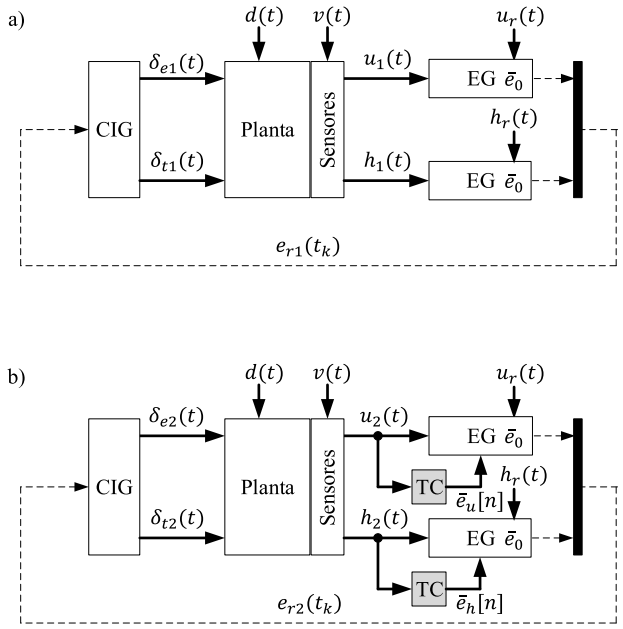

Figura 4.18: Piloto automático. a) Umbrales estáticos, b) umbrales dinámicos.

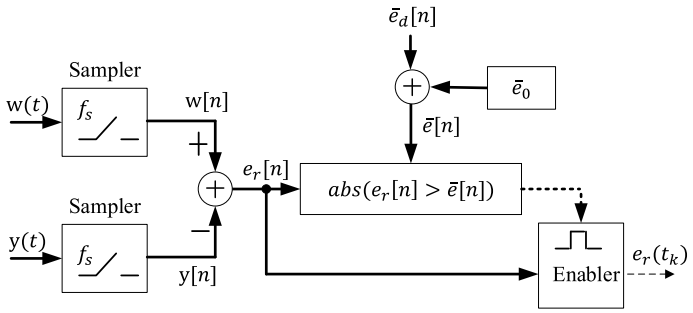

Figura 4.19: Generador de eventos del piloto automático.

esta arquitectura, este módulo trabaja en tiempo discreto con una $f_s = 50$ Hz (frecuencia típica de los sistemas UAVs). La condición de evento es tal que se generará un evento cuando la señal de error $e_r[n]$ supera el umbral dinámico $\bar{e}[n]$. Este umbral dinámico es la suma de una componente estática $\bar{e}_0$ más la componente dinámica $\bar{e}_d[n]$ que se obtiene en el calculador de umbral. El calculador de umbral propuesto en este caso se muestra en la Figura 4.20. Si se compara esta arquitectura con la mostrada en la Figura 4.13 puede observarse que siguen una filosofía similar y por tanto se está aplicando la misma solución que al caso de los robots móviles.

Para analizar la respuesta de este sistema ante las perturbaciones (ruido en los sensores) se simularán las dos realizaciones mostradas en la Figura 4.18 a) y a la Figura 4.18 b). En la simulación la señal de consigna $u_r(t)$ varía desde 25

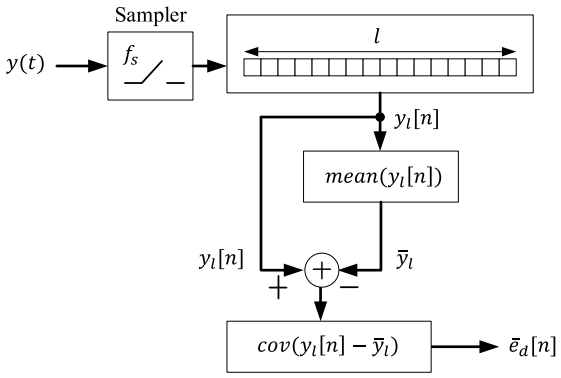

Figura 4.20: Calculador de umbral.

m/s hasta 30 m/s en $t = 10$ s y luego retorna a 25 m/s en el instante $t = 25$ s. La referencia de altura $h_r(t)$ se hace variar desde los 300 m hasta los 350 m en el instante $t = 10$ s. Respecto a la perturbación que afecta al sistema, los sensores están contaminados por un ruido con distribución normal $\mathcal{N}(0, \sigma)$ donde $\sigma = 1$ m/s para el sensor de velocidad y $\sigma = 1$ m para el de altitud.

Para la sintonía del generador de eventos se ha puesto un $\bar{e}_0 = 0,3$, es decir, $0,3$ m/s para la velocidad y $0,3$ m para la altitud. Respecto al calculador de umbral, que se incluye solo en la segunda realización (Figura 4.18 b)), se fija el valor de $l = 10$.

Una vez simulados ambos sistemas, la estructura de control basada en eventos con umbrales estáticos y con umbrales dinámicos, sus respuestas se presentan en las Figuras 4.21, 4.22 y 4.23.

En la Figura 4.21 se observa que el sistema con umbrales dinámicos en cada periodo de muestreo determina un umbral en función del ruido para $\bar{e}_u[n]$ y $\bar{e}_h[n]$. Además, se puede comprobar que el sistema de umbrales estáticos genera muchos más eventos que el sistema propuesto basado en umbrales dinámicos. Este análisis ha mostrado que el sistema de umbrales dinámicos tiene un comportamiento mejor desde el punto de vista de generación de eventos y por tanto en eficiencia en el uso de recursos. Ahora habría que valorar cual es la respuesta del sistema y analizar si cumple con las prestaciones exigidas. Para ello, hay que analizar la salida del sistema mostrada en la Figura 4.22.

Como se observa, la respuesta del sistema con umbrales estáticos es similar al que trabaja con umbrales dinámicos y ambos siguen de forma correcta las referencias. Por tanto, ambos sistemas cumplen los requisitos marcados al sistema de control. En cuanto al número de eventos, si se analiza este parámetro de forma

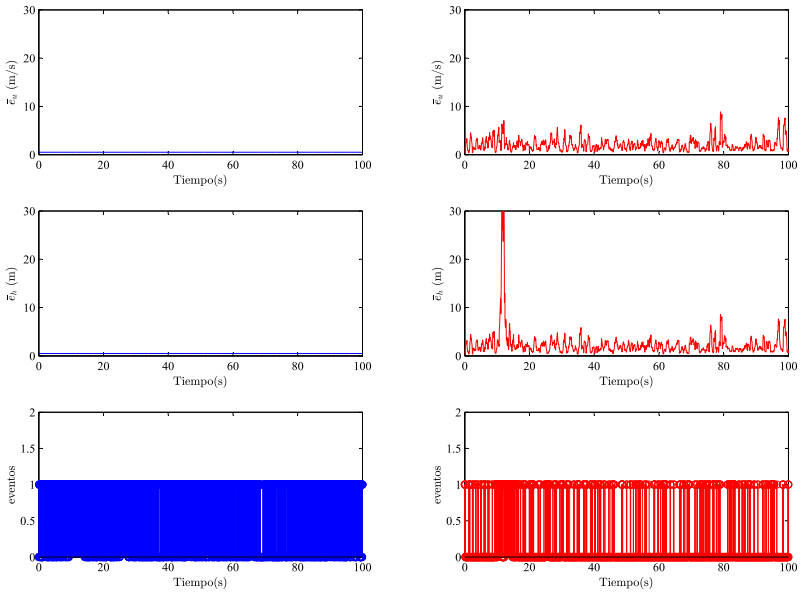

Figura 4.21: Umbral de velocidad ($\bar{e}_u$), umbral de altitud ($\bar{e}_h$) y número de eventos. Trazo azul con umbrales estáticos y trazo rojo con calculador de umbral.

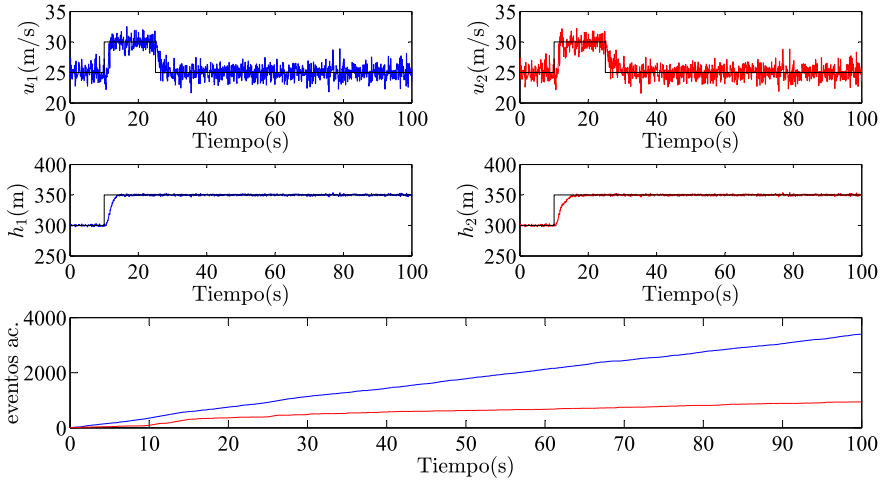

Figura 4.22: Respuesta en velocidad, altitud y número de eventos acumulados. Referencia en trazo negro, en trazo azul con umbrales estáticos y en trazo rojo con calculador de umbral.

acumulada se concluye que el sistema con umbrales dinámicos presenta grandes eficiencias respecto al que trabaja con umbrales estáticos.

Por último, si se analizan las señales de control en ambos sistemas (señales de timón de profundidad y mando de gases) tal como muestra la Figura 4.23, se observa que las señales de control del sistema con umbrales dinámicos tiene un

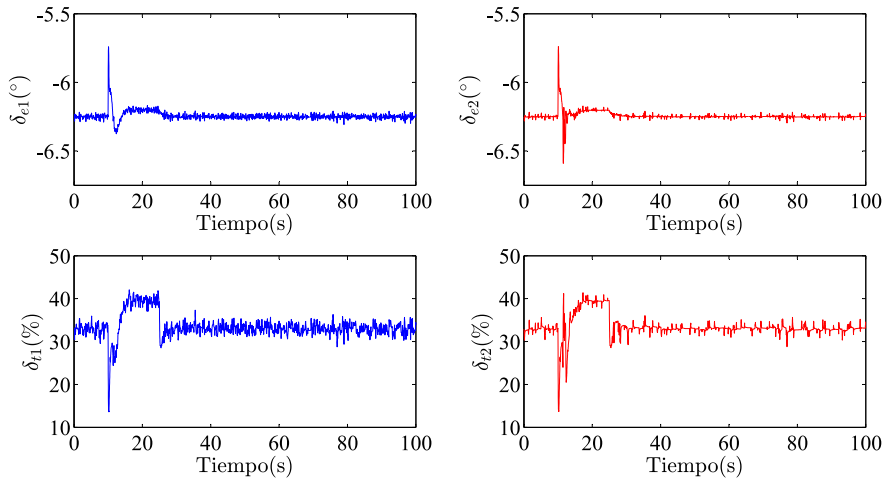

Figura 4.23: Señales de control. Posición del elevador y mando de gases. En trazo azul con umbrales estáticos y en trazo rojo con calculador de umbral.

nivel de ruido inferior. Esto se debe a que el CIG envía una orden de control a los actuadores solo cuando se produce un evento. En el sistema propuesto dado que la condición de evento se adapta de forma dinámica al ruido del sistema, el número de eventos es inferior y por tanto el ruido en los actuadores disminuye. Gracias a este módulo, en entornos de alto nivel de ruido los actuadores reciben menos perturbaciones lo que permite aminorar los efectos de desgaste y el *chattering* de éstos.

4.7. Conclusiones

En este capítulo se ha demostrado que el ruido de medida impacta negativamente en un sistema de control basado en eventos. Esta perturbación provoca un exceso de eventos de forma innecesaria ocasionando que el sistema de control sea menos eficiente en el consumo de recursos. En este capítulo se ha investigado el comportamiento de las arquitecturas de control propuestas en presencia de ruido. Tras este análisis se concluye que, si se puede determinar la envolvente del ruido que afecta a las medidas, pueden fijarse los umbrales de eventos de forma más precisa. De esta forma, se consigue que el sistema sea más eficiente en cuanto al consumo de recursos. Se han propuesto dos estructuras para conseguir fijar los umbrales de forma óptima: La primera se basa en la estimación del ruido a partir de la salida del sistema (desarrollada en este capítulo) y la segunda, a través de un

modelo de la perturbación que afecta a los sensores (se presentará en el Capítulo 5). La estructura aquí desarrollada y aplicada en algoritmos de navegación para robots móviles presenta grandes eficiencias respecto a un sistema que trabaje con un umbral estático. También se ha demostrado que, esta solución puede aplicarse a estructuras más complejas como son los UAVs consiguiendo resultados igual de satisfactorios.

Capítulo 5

Análisis y Modelado de Perturbaciones

En el presente capítulo se desarrolla la idea de fijar los umbrales en el sistema de control basado en eventos a partir del modelo de perturbación que afecta a las medidas de los sensores. Se analiza cómo fijar estos umbrales a partir del modelo de perturbación y conseguir de esta forma tener una mayor eficiencia en el consumo de recursos de comunicación, computación y energía. Termina el capítulo con la definición de una metodología que permite obtener un modelo de ruido de los sensores a partir de utilizar patrones conocidos, utilizándose luego este modelo para fijar los umbrales en el sistema.

5.1. Introducción

En la Sección 4.4 se analizaron los efectos que tiene el ruido en el sistema a la hora de generar eventos. Posteriormente, en la Sección 4.6, se propuso una arquitectura de control que calculaba los umbrales de forma dinámica teniendo en cuenta el ruido del sistema. De esta forma el sistema tiene una respuesta similar al sistema con umbrales estáticos, pero sin que el número de eventos aumente de forma descontrolada. Finalmente en la Sección 4.6.1 se propuso una estructura para determinar estos umbrales dinámicos considerando dos posibles entradas, la señal de salida del sistema o un modelo del perturbación del ruido de medida que afecta al sistema.

El algoritmo de navegación DD con control basado en eventos propuesto en este trabajo (véase Sección 3.3.2), está dirigido por la condición de parada e_ρ y

por el *event threshold* $\bar{e}_\alpha$. Por otro lado, la variable que gobierna la generación de eventos es el error de posicionamiento $\alpha[n]$. Cuando esta variable supera el valor umbral $\bar{e}_\alpha$, se genera un evento en el sistema. Si se añade ruido de medida en los sensores, el sistema tendrá un comportamiento como el que se muestra en la Figura 5.1.

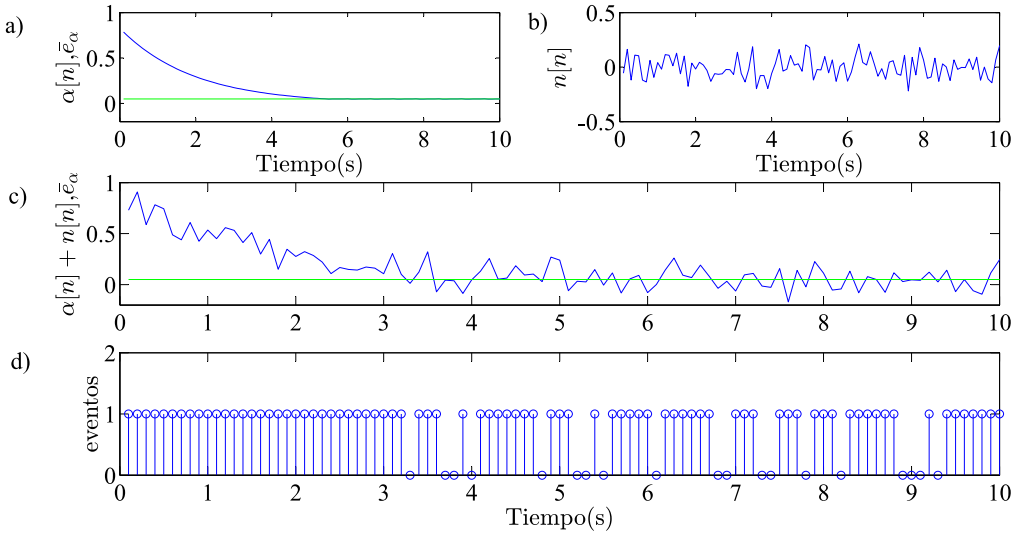

Figura 5.1: Respuesta del algoritmo DD con ruido de medida.

Tal como se observa en la Figura 5.1 a), el parámetro $\alpha[n]$ tiende a cero a medida que el robot se aproxima al destino. Una vez que este parámetro sea inferior al valor $\bar{e}_\alpha$ el sistema dejará de generar eventos. Si ahora la señal de medida se contamina con ruido (Figura 5.1 b)), cuando el sistema esté cercano al estado estacionario, seguirá generando eventos a causa del ruido (véase Figuras 5.1 c) y d)).

Si el ruido que afecta a las medidas es un ruido aditivo, lo que se necesita para evitar que esta perturbación dispare el número de eventos en el sistema es estimar el valor de su envolvente. De esta forma, se puede utilizar un valor para el *event threshold* que sea la suma de este valor de la envolvente del ruido más el umbral estático del sistema y así controlar que no se dispare la generación de eventos en el sistema.

Se concluye que, si se dispone de un modelo de ruido que permita estimar de forma precisa la envolvente de esta perturbación que afectan a las medidas de los sensores, el sistema generará eventos solo cuando sea necesario. Por tanto para llevar a cabo esta tarea será necesario estimar cuales son los modelos de ruido

que afectan a las medidas de los sensores.

5.2. Modelado de las Perturbaciones

En la Sección 4.6.1 se propuso una arquitectura general para estimar los umbrales tanto de forma dinámica como a partir de un modelo de perturbación. En este sistema se puede obtener este umbral a partir de la salida del sistema $y[n]$ o bien a través del modelo del ruido que afecta a los sensores $P[n]$. La estructura propuesta se presenta en la Figura 5.2.

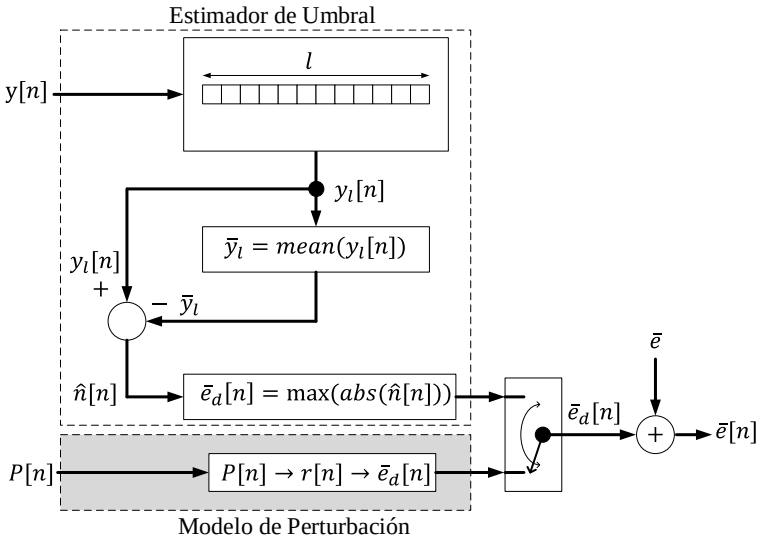

Figura 5.2: Calculador de umbral basado en un modelo de perturbación.

En el Capítulo 4 se desarrolló el bloque del estimador de umbral (Figura 5.2) y se analizó su aplicación al algoritmo DD. En este capítulo, se va a estudiar la segunda alternativa, es decir, analizar el comportamiento del sistema si tenemos un modelo de perturbación $P[n]$ conocido.

Si se dispone de un modelo de la perturbación $P[n]$, obtener un valor aproximado de la envolvente es tan sencillo como calcular el percentil $Per()$ del valor absoluto de una realización $r[n]$ del modelo de perturbación $P[n]$. Dicho de otra forma, la componente dinámica del *event threshold* $\bar{e}_d[n]$ se puede escribir de la forma:

$$\bar{e}_d[n] = Per(abs(r[n]), p) \tag{5.1}$$

En este caso, el parámetro p indica el percentil. Por lo general se elige un valor

que puede oscilar entre el 70 % y el 80 % para tener una muestra lo suficientemente representativa de $r[n]$.

Con este nuevo planteamiento, se va a comparar esta nueva metodología de cálculo con la presentada en el Capítulo 4. Para ello, se utilizan tres distribuciones diferentes de ruido que contaminan los sensores que miden las distancias recorridas por las ruedas d_l y d_r (d_l distancia recorrida por la rueda izquierda y d_r por la rueda derecha). Las mencionadas distribuciones se presentan en la Tabla 5.1.

Tabla 5.1: Distribuciones utilizadas como ruido de medida.

Distribución	parámetro 1 (cm)	parámetro 2 (cm)
Normal	$\mu = 0$	$\sigma = 0,2$
Uniforme	$a = -0,3$	$b = 0,3$
Rayleigh	$b = 0,2$	-

Posteriormente, se hace trabajar el algoritmo de navegación DD sobre el sistema de control basado en eventos con dos escenarios diferentes:

- *Escenario 1*: El sistema obtiene los umbrales de forma dinámica con el módulo desarrollado en el Capítulo 4.
- *Escenario 2*: Se obtiene el *event threshold* partiendo del modelo de perturbación $P[n]$ que afecta al sistema (véase Tabla 5.1). Dicho umbral se obtiene a través de la Ecuación 5.1.

Los parámetros más relevantes de los escenarios comentados se presentan en la Tabla 5.2.

Tabla 5.2: Escenarios de análisis para el algoritmo DD con calculador de umbral y estimación del modelo de perturbación.

Escenario	Color	K_ρ(1/s)	K_α(1/s)	$\bar{e}_\alpha$ (rad)	e_ρ (cm)	l	p (%)
1	azul	0,1	0,5	0,05	0,5	10	-
2	rojo	0,1	0,5	0,05	0,5	-	70

En las Figuras 5.3 y 5.4 se presentan los resultados tras simular los mencionados escenarios..

Tal como se muestra en la Figura 5.3, independientemente del tipo de perturbación que afecte a las medidas del sistema, la opción de estimar el umbral a

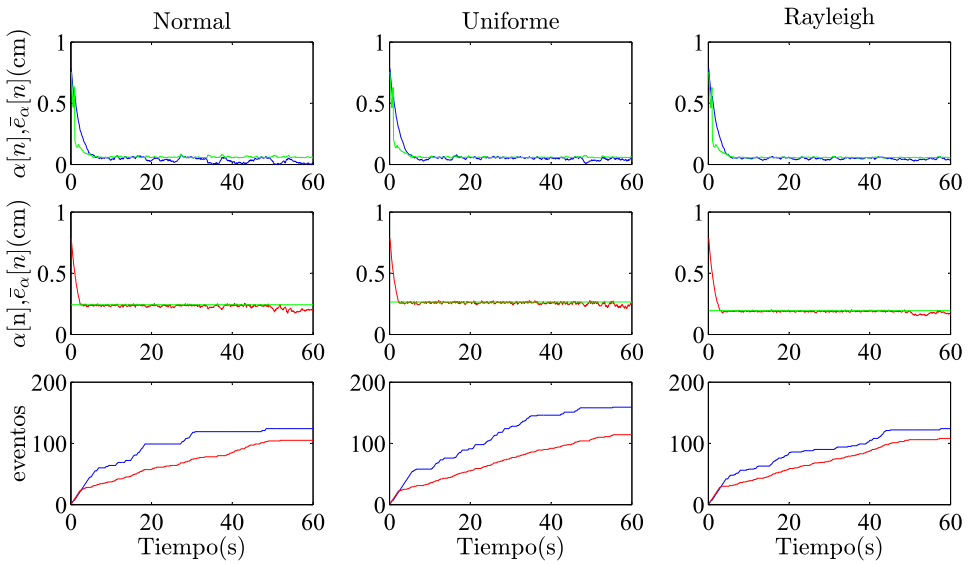

Figura 5.3: Error de orientación (α), umbrales ($\bar{e}_\alpha$) y eventos acumulados por distribución. Umbral en trazo verde, trazo azul sistema con umbrales dinámicos y trazo rojo sistema con modelo de ruido.

través de un modelo de perturbación conocido (Escenario 2) provoca que el sistema genere menos actividad y que por tanto sea más eficiente. Como se observa en esta figura, aunque la estimación según la metodología expuesta en el Capítulo 4 es una buena aproximación de la envolvente de ruido, ésta es menos eficiente que la nueva propuesta basada en el modelo de ruido $P[n]$. La razón de que la metodología anteriormente desarrollada pierda precisión se debe a que el sistema comete errores al estimar el ruido cuando está en estado transitorio. Por tanto, el nuevo sistema propuesto mejora la actividad de forma notable.

Otro aspecto que se debe considerar es la precisión del sistema. Si se comparan los Escenarios 1 y 2 con el mismo sistema de control basado en eventos sin perturbaciones, se puede definir el error de posicionamiento en el eje x como $E_x[n]$ y en el eje y como $E_y[n]$ según las ecuaciones:

$$E_x[n] = abs(x_r[n] - x_{sr}[n]) \tag{5.2}$$

$$E_y[n] = abs(y_r[n] - y_{sr}[n]) \tag{5.3}$$

donde $x_r[n]$ e $y_r[n]$ son las medidas de posición obtenidas por el sistema en presencia de ruido. En cambio $x_{sr}[n]$ e $y_{sr}[n]$ son las medidas del mismo sistema en ausencia de ruido. En la Figura 5.4 se presentan estos valores para cada uno de

los escenarios analizados.

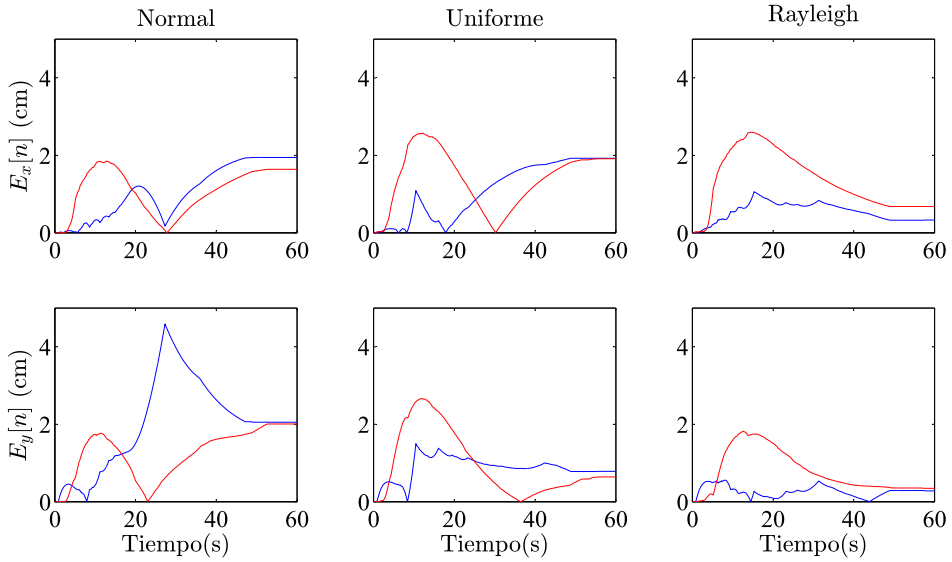

Figura 5.4: Errores de posicionamiento por distribución. Trazo azul con umbrales dinámicos y trazo rojo con modelo de ruido.

Tal como se muestra en la Figura 5.4, para ambos escenarios los errores de posicionamiento al final de la navegación ($t = 60$ s) son prácticamente los mismos, no superando el 4 % en todos los casos. Los valores intermedios de este error no deben considerarse ya que el objetivo del algoritmo es alcanzar el destino y por tanto el valor relevante es el error que se comete al final del trayecto.

Tal como se ha analizado en esta sección, el método más óptimo para fijar los umbrales en los sistemas de eventos desarrollados en este trabajo es el basado en un modelo de perturbación. En las simulaciones analizadas se ha observado que independiente de la perturbación que afecte al sistema (Normal, Uniforme, Rayleigh, etc.) la metodología desarrollada en este capítulo presenta mejores resultados que las soluciones planteadas en el Capítulo 4.

Llegados a este punto, se pueden plantear dos estrategias a la hora de fijar los umbrales del sistema. Si se puede estimar el modelo de ruido que afecta a los sensores se estimará la distribución de ésta y posteriormente, mediante una realización de este modelo se obtiene el valor del umbral. Si la opción anterior no es posible, se pueden usar la técnicas mostradas en el Capítulo 4 que también mejoran de forma considerable la respuesta del sistema respecto a uno que trabaje con umbrales estáticos.

5.3. Estimación de los Parámetros del Modelo de Perturbación

En la sección anterior ha quedado demostrado que la mejor manera de fijar los umbrales en las arquitecturas de control por eventos es utilizar un modelo de perturbación $P[n]$. Por tanto, en este tipo de sistemas, si es posible modelar la perturbación que afecta a las medidas de los sensores, se podrá fijar el *event threshold* que gobierna el sistema de control de manera más precisa.

Las perturbaciones que afectan a las medidas de los sensores $P[n]$ son variables aleatorias (VA) que están descritas mediante distribuciones de probabilidad. Estas funciones generalmente pueden modelarse con una serie de parámetros característicos y por tanto conocidos éstos, la VA bajo estudio queda totalmente definida. Por tanto, el problema a resolver es estimar los parámetros que modelan esta VA a partir de un conjunto de medidas (realización) de ésta.

Dentro de los métodos más típicos para obtener estos parámetros se encuentran los siguientes (Soong [135], Gorgas et al. [50], Leon-Garcia [70]):

- *Método de los Momentos.*
- *Método de Máxima Verosimilitud.*

En las siguientes secciones se describirá como aplicar estos métodos para obtener un modelo de perturbación.

5.3.1. Método de los Momentos

Este método consiste en igualar los momentos muestrales respecto del origen a los correspondientes momentos poblacionales α_k que están relacionados con los parámetros de la distribución. Si $P[n]$ es una VA continua (secuencia aleatoria continua) con función de densidad de probabilidad $f(r)$, el momento de orden k ($k \geq 1$) respecto al origen α_k se define mediante:

$$\alpha_k = \int_{-\infty}^{+\infty} r^k f(r) dr \tag{5.4}$$

Si el número de parámetros que hay que estimar es k, dada una muestra aleatoria simple $r[n]$ de tamaño n $(r_1, ..., r_n)$, se plantea el sistema de ecuaciones

5.5 que de forma general será no lineal.

$$\begin{aligned} \alpha_1 &= \textstyle\sum_{i=1}^{n} \frac{r_i}{n} \\ \alpha_2 &= \textstyle\sum_{i=1}^{n} \frac{(r_i)^2}{n} \\ &\vdots \\ \alpha_k &= \textstyle\sum_{i=1}^{n} \frac{(r_i)^k}{n} \end{aligned} \tag{5.5}$$

De este sistema de ecuaciones se obtienen los parámetros en función de los valores de la muestra, obteniendo así los estimadores por el método de los momentos. Generalmente, los parámetros de los que depende la distribución de una VA suelen ser la media poblacional, o la varianza o algún valor relacionado con éstos. Puede verse fácilmente que estas medidas están relacionadas con los momentos respecto del origen, por ejemplo $\alpha_1 = \mu$ y $\alpha_2 = \sigma^2 + \mu^2$.

Si con este método se quiere obtener p.e. un estimador de la media poblacional, puesto que hay que estimar un único parámetro, se plantea solo la siguiente ecuación:

$$\alpha_1 = \sum_{i=1}^{n} \frac{r_i}{n} \tag{5.6}$$

como $\alpha_1 = \mu$, sustituyendo en la Ecuación 5.6, se obtiene la expresión,

$$\mu = \sum_{i=1}^{n} \frac{r_i}{n} \tag{5.7}$$

y por tanto el estimador será $\hat{\mu} = \sum_{i=1}^{n} \frac{r_i}{n}$.

Si por el contrario, se quiere obtener la media y la varianza poblacional, en este caso hay que estimar dos parámetros y como tal habrá que plantear el siguiente sistema de dos ecuaciones:

$$\begin{aligned} \alpha_1 &= \textstyle\sum_{i=1}^{n} \frac{r_i}{n} \\ \alpha_2 &= \textstyle\sum_{i=1}^{n} \frac{(r_i)^2}{n} \end{aligned} \tag{5.8}$$

Como los momentos respecto al origen están relacionados con la media y la varianza, se puede escribir la ecuación:

$$\begin{aligned} \mu &= \textstyle\sum_{i=1}^{n} \frac{r_i}{n} \\ \sigma^2 + \alpha^2 &= \textstyle\sum_{i=1}^{n} \frac{(r_i)^2}{n} \end{aligned} \tag{5.9}$$

Despejando μ y σ se obtiene la expresión:

$$\begin{aligned} \mu &= \sum_{i=1}^{n} \frac{r_i}{n} \\ \sigma^2 &= \sum_{i=1}^{n} \frac{(r_i - \bar{r})^2}{n} \end{aligned} \tag{5.10}$$

Por tanto, los correspondientes estimadores por el método de los momentos son $\hat{\mu} = \sum_{i=1}^{n} \frac{r_i}{n}$ y $\hat{\sigma}^2 = \sum_{i=1}^{n} \frac{(r_i - \bar{r})^2}{n}$

5.3.2. Método de Máxima Verosimilitud

La idea fundamental de este método es tomar como estimación del parámetro estudiado el valor que haga máxima la probabilidad de obtener la muestra observada. Este método se formula como se describe a continuación. Sea $P[n]$ una VA cuya distribución depende de un conjunto de parámetros $\theta_1, \theta_2, \cdots, \theta_k$, desconocidos y cuyo valor se quiere estimar. Por otro lado, sea $(r_1, \cdots, r_n)$ una muestra aleatoria simple de $P[n]$. Si se denota $\boldsymbol{\theta} = (\theta_1, \theta_2, \cdots, \theta_k)$, se denomina función de verosimilitud para la muestra $(r_1, \cdots, r_n)$ a la función definida sobre el conjunto de posibles valores del parámetro $\boldsymbol{\theta}$ dada por la ecuación:

$$l(\boldsymbol{\theta}) = f_{\boldsymbol{\theta}}(r_1, r_2, \cdots, r_n) = f_{\boldsymbol{\theta}}(r_1) \cdot f_{\boldsymbol{\theta}}(r_2) \cdots f_{\boldsymbol{\theta}}(r_n) \tag{5.11}$$

El estimador de máxima verosimilitud (EMV) $\hat{\boldsymbol{\theta}}$ para la muestra $(r_1, \cdots, r_n)$ es el valor del vector $\boldsymbol{\theta}$ para el cuál la función de verosimilitud alcanza el máximo absoluto según la expresión:

$$\hat{\boldsymbol{\theta}} = \max_{\theta}[l(\boldsymbol{\theta})] \tag{5.12}$$

Por tanto, para aplicar el método de máxima verosimilitud hay que seguir los siguientes pasos:

1. Formar la función de verosimilitud para una muestra arbitraria de tamaño n.

2. Resolver el correspondiente problema de máximos absolutos en el dominio de los parámetros.

3. Definir como EMV las expresiones obtenidas al determinar el máximo absoluto.

Por otro lado, el método de máxima verosimilitud plantea ciertas dificultades en la práctica como son:

- No siempre existe el máximo absoluto para la función de verosimilitud.
- Aún cuando éste exista, para determinarlo es necesario resolver un problema de extremos absolutos restringidos a un dominio de $\mathbb{R}^n$, problema que no siempre es fácil de resolver.

En muchas ocasiones, en lugar de maximizar la función de verosimilitud es más fácil maximizar la función $L(\boldsymbol{\theta}) = \ln(l(\boldsymbol{\theta}))$, que se denomina función de soporte. Si la función $l(\boldsymbol{\theta})$ es estrictamente positiva en el dominio de $\boldsymbol{\theta}$, entonces los máximos de una y otra función se corresponden y por tanto, maximizar una es equivalente a obtener los máximos de la otra. Según esto se puede enunciar el Teorema de Invarianza que indica que si $\hat{\boldsymbol{\theta}}$ es el EMV de $\boldsymbol{\theta}$, entonces $g(\hat{\boldsymbol{\theta}})$ es el EMV de $g(\boldsymbol{\theta})$.

Comentar respecto a los métodos de estimación que, el método de los momentos fue muy popular durante varias décadas pero perdió interés cuando Fischer introdujo el método de Máxima Verosimilitud allá por 1922. Por tanto, el método de máxima verosimilitud se ha convertido en uno de los métodos más relevantes de estimación. Su mayor atractivo radica en el hecho de que si se tienen muestras grandes se obtienen estimadores suficientes, asintóticamente no sesgados y de varianza mínima (Soong [135], Gorgas et al. [50]). Por estas razones, las metodologías que se desarrollan en este capítulo se apoyarán en el método de Máxima Verosimilitud.

5.4. Metodología para Estimar el Modelo de Perturbación

Una vez que se dispone de un modelo del ruido que afecta al sistema se pueden definir los umbrales que gobiernan al sistema basado en eventos de forma más precisa tal como se explicó en la Sección 5.2. Por tanto, en el presente capítulo, se plantea una metodología que ayuda a estimar el modelo de la perturbaciones $P[n]$ que afectan a las medidas de los sensores, para ello se hará un análisis basado en el esquema mostrado en la Figura 5.5.

Tal como se muestra en dicha figura, el procedimiento que se sigue en esta metodología es el siguiente:

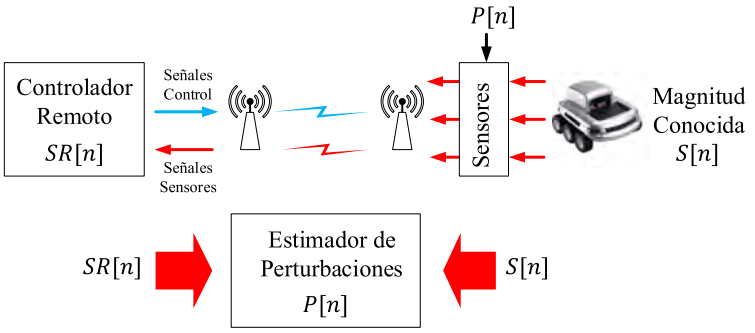

Figura 5.5: Esquema de análisis y modelado de perturbaciones.

1. Los sensores del robot una vez calibrados miden un patrón conocido $S[n]$. Dependiendo del sensor que se quiera modelar se elegirá el patrón concreto a medir.

2. Las medidas capturadas por el sensor, están contaminadas por el ruido $P[n]$ que afecta a este sensor. Posteriormente estos valores se envian al controlador a través del enlace inalámbrico. Las señales que llegan al controlador se representan como $SR[n]$.

3. Las señales que llegan al controlador tendrán por tanto los valores del patrón conocido $S[n]$ contaminado por el ruido de los sensores $P[n]$. Por tanto, la señal $SR[n]$ se puede escribir según la expresión:

$$SR[n] = f(S[n], P[n]) \tag{5.13}$$

El problema a resolver ahora es estimar $P[n]$ a partir de la Ecuación 5.13. Por tanto, antes de definir el método de cálculo se fijarán unas hipótesis de como son las perturbaciones que afectan a los sensores. Para definir la naturaleza de la perturbación $P[n]$ se hará una breve recapitulación de las posibles opciones a valorar según se presenta en la Figura 5.6. En el caso que nos ocupa se supone que el ruido $P[n]$ es una variable aleatoria que afecta a la señal de los sensores de forma aditiva. Desde el punto de vista estadístico se supondrá que se trata de una secuencia aleatoria continua. Con estas premisas, la forma de estimar el modelo de distribución de probabilidad $f(r)$ o en su caso los parámetros del modelo $\hat{\boldsymbol{\theta}}_r$ se presenta en el esquema mostrado en la Figura 5.7

Según se observa en la Figura 5.7, las fases para estimar el modelo de distribución que mejor se ajusta a las perturbaciones bajo estudio y cuales son los

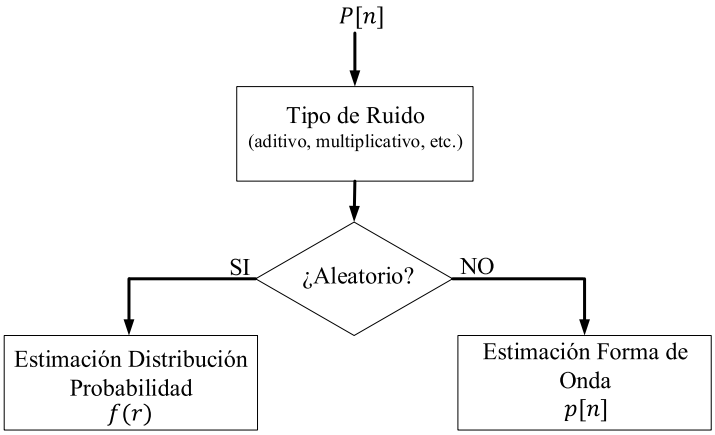

Figura 5.6: Naturaleza de las perturbaciones.

parámetros $\hat{\boldsymbol{\theta}}_r$ de éste son las siguientes:

- Paso 1. Obtención de una realización de la perturbación $r[n]$. Dado que se asume que el ruido que afecta al sistema de sensores es aditivo, una realización de éste puede obtenerse a través la señal que llega al controlador $SR[n]$ y la señal patrón $S[n]$ que se utiliza para estimar la perturbación. Puesto que ambas señales son conocidas, $r[n]$ se obtiene mediante la ecuación:

$$r[n] = SR[n] - S[n] \tag{5.14}$$

- Paso 2. A partir de una amplia BBDD de distribuciones, se estiman mediante el método de máxima verosimilitud los parámetros que ajustan cada una de las distribuciones i de la BBDD a la realización de la perturbación $r[n]$ bajo estudio. Por tanto, para cada distribución i de la BBDD, se construye la función de verosimilitud $l(\boldsymbol{\theta}_i)$ y luego se maximiza según la Ecuación 5.15 para obtener los parámetros $\hat{\boldsymbol{\theta}}_i$.

$$\hat{\boldsymbol{\theta}}_i = \max_{\theta}[l(\boldsymbol{\theta}_i)] \tag{5.15}$$

- Paso 3. Una vez obtenido un modelo de cada una de las distribuciones i ajustadas a la realización $r[n]$ según el método de máxima verosimilitud, se obtienen las funciones de distribución acumulada $F_i(x)$ de cada una de ellas. Al mismo tiempo, también se obtiene $F_r(x)$ que se corresponde con la distribución $r[n]$ a estimar. La función de distribución acumulada se define

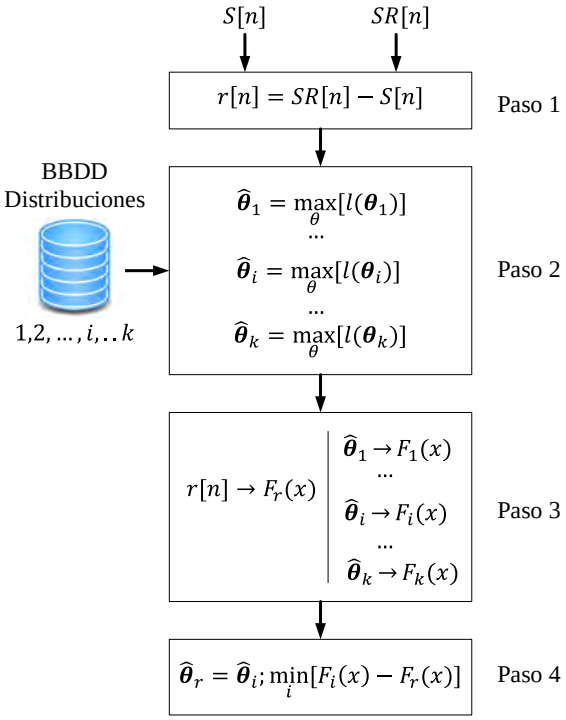

Figura 5.7: Metodología de estimación del modelo de ruido.

según la expresión:

$$F_i(x) = \sum_{r_j \leq x} f_i(r_j) \tag{5.16}$$

- Paso 4. Finalmente, se comparan cada una de las funciones $F_i(x)$ con la de la realización $F_r(x)$. Como resultado el modelo de ruido será aquel $\boldsymbol{\theta}_i$ que más aproxime su función $F_i(x)$ a $F_r(x)$ tal como se denota en la siguiente ecuación:

$$\begin{gathered} \hat{\boldsymbol{\theta}}_r = \hat{\boldsymbol{\theta}}_i \\ \underset{i}{min}[F_i(x) - F_r(x)] \end{gathered} \tag{5.17}$$

Llegados a este punto, queda definida una metodología para obtener el modelo de ruido que afecta a las medidas de los sensores. En el caso particular de este trabajo, se aplicará para fijar los umbrales en el sistema de control (véase Socas et al. [129]), pero este método puede ser también de aplicación en otros ámbitos dentro de la ingeniería de control o en el campo de la robótica en general.

5.5. Conclusiones

En el presente capítulo se ha desarrollado una metodología que ayuda a establecer de forma eficiente los umbrales de eventos en los sistemas de control bajo estudio. El método propuesto permite modelar de forma sencilla e intuitiva las perturbaciones que afectan a los sensores del robot y obtener un modelo teórico de éste. Con la ayuda de este modelado, se demuestra mediante un entorno de simulación, que la respuesta del sistema mejora de forma considerable al conseguir un ajuste más preciso del umbral de eventos. Por tanto, con esta nueva metodología se consigue que los sistemas de control desarrollados en esta tesis sean más robustos frente al ruido.

Capítulo 6

Posicionamiento de Robots Móviles

En este capítulo se plantean nuevas ideas que permiten mejorar el posicionamiento de los robots móviles de bajo coste, centrando la atención en los modelos utilizados en la plataforma de experimentación de este trabajo. Dado que este tipo de robots obtienen su posicionamiento básicamente con técnicas odométricas, se propone un algoritmo de estimación de posicionamiento que combina la odometría con estimaciones de inclinación basadas en medidas inerciales. Se plantea a su vez, un nuevo sistema de sincronización absoluta de la orientación a partir de las medidas del acelerómetro para mejorar la estimaciones de posicionamiento frente a las técnicas clásicas basadas en odometría.

6.1. Técnicas de Posicionamiento

En los algoritmos de navegación que se están tratando en esta tesis y en general, para cualquier aplicación con robots móviles es crucial una estimación precisa del posicionamiento y de la orientación (Borenstein and Feng [20]). Pueden usarse diferentes técnicas para resolver este problema. Las más comunes pueden dividirse en siete categorías (Wehe et al. [148], Lee and Park [69]):

1. *Odometría - Odometry.*
2. *Navegación inercial - Inertial navigation.*
3. *Brújulas magnéticas - Magnetic compasses.*

4. *Balizas activas - Active beacon.*
5. *Sistemas de Posicionamiento Global por satélite (GPS) - Global Positioning Systems.*
6. *Mediante puntos de referencia - Landmark navigation.*
7. *Emparejamiento de modelos - Model matching.*

Al mismo tiempo, estas técnicas se pueden categorizar en dos grupos según la naturaleza de sus medidas: relativas, también denominadas de estima (*dead reckoning*) la cuales incluyen las categorías 1 y 2 y absolutas, que incluyen el resto de categorías. La mayoría de las aplicaciones utilizan una combinación de categorías, una de cada grupo para compensar las carencias de un método único. En el caso en que se trate con robots de bajo coste, que normalmente disponen de pocos sensores, no podrán combinarse estas categorías y por tanto, habrá que buscar métodos alternativos que permitan tener un posicionamiento lo más preciso posible.

6.2. Posicionamiento en Robots Móviles de Bajo Coste

Los robots de bajo coste, entre los que se incluyen las plataformas utilizadas para la experimentación en esta tesis, disponen de pocos sensores. Estos sistemas típicamente contienen codificadores para estimar el movimiento de las ruedas, acelerómetros y detectores de obstáculos (Siegwart et al. [126], Everett [43], Faisal et al. [45]). Por esta razón, en este tipo de plataformas las únicas técnicas de posicionamiento que pueden aplicarse son Odometría y la Navegación Inercial. Aunque ambas técnicas producen errores acumulativos, tienen una precisión razonable en periodos de trabajo no muy largos. La odometría se basa en medir la distancia que recorre cada una de las ruedas a lo largo del tiempo. Con esta información se puede obtener tanto la posición como la orientación del robot. Por otro lado, la navegación inercial se apoya en los acelerómetros y los giróscopos para medir la aceleración y las velocidades angulares de rotación del robot. Estas medidas tienen que ser integradas a lo largo del tiempo para obtener la posición y

la orientación mediante técnicas de navegación inercial. Por otro lado, los giróscopos son sensores que tienen un precio elevado, por tanto, en este tipo de robots no suele ser habitual que se disponga de ellos.

Debido a esto, en los robots móviles de bajo coste las dos técnicas que pueden usarse para estimar la posición son: odometría y navegación inercial solo con acelerómetros. Generalmente los acelerómetros de bajo coste tienen una mala relación señal-ruido cuando el robot tiene bajas aceleraciones. Estas bajas aceleraciones son una situación común en muchas aplicaciones, por ejemplo en sistemas de robots con velocidad constante. Por tanto, la estimación del posicionamiento y la orientación mediante acelerómetros en plataformas de bajo coste es una mala solución (Liu and Pang [76]). Aunque, por otro lado, este tipo de acelerómetros sí suelen tener una muy buena respuesta en la estimación de ángulos de inclinación (Trimpe and DÁndrea [141], Luczak et al. [78]).

6.3. Posicionamiento y Orientación en 3D

La estimación de la posición y de la orientación de los robots móviles es una condición básica para su autonomía. El posicionamiento en 3D para un robot móvil puede definirse como un vector con seis componentes $\mathbf{p_0} = (\hat{x}\ \hat{y}\ \hat{z}\ \hat{\varphi}\ \hat{\theta}\ \hat{\psi})^T$ (Roberson and Schwertassek [115]). Por tanto, la posición $(\hat{x}\ \hat{y}\ \hat{z})$ y la orientación $(\hat{\varphi}\ \hat{\theta}\ \hat{\psi})$ se definen respecto al sistema de referencia inercial (O) tal como se muestra en la Figura 6.1.

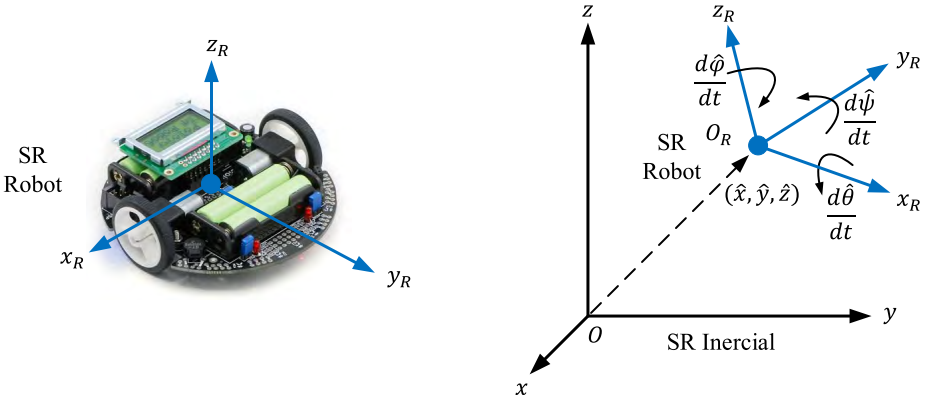

Figura 6.1: Parámetros que definen la posición y la orientación de un robot en 3D.

En este caso, la orientación $\hat{\varphi}$, la elevación $\hat{\theta}$ y el alabeo $\hat{\psi}$ son los ángulos de Tait-Bryan. En el esquema presentado en la Figura 6.1 también se ha definido el sistema de referencia móvil O_R situado en el centro de gravedad del robot.

Cuando el robot está navegando sobre una superficie plana S, sobre ésta puede definirse un nuevo sistema de referencia O_S (véase Figura 6.2 a)) que permita hacer una estimación de la posición basada en odometría.

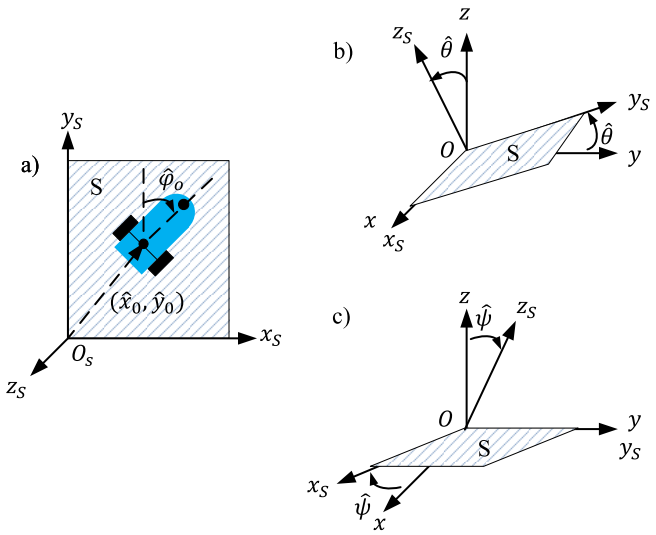

Figura 6.2: Posicionamiento del robot respecto a una superficie S.

Por tanto, a través de odometría pueden estimarse la posición ($\hat{x}_0$,$\hat{y}_0$) y la orientación $\hat{\varphi}_0$ respecto al sistema de referencia definido sobre la superficie O_S (véase Figura 6.2 a)). Por otro lado, mediante los acelerómetros del robot se pueden estimar las inclinaciones de esta superficie S y por tanto estimar los ángulos de elevación $\hat{\theta}$ (véase Figura 6.2 b)) y de alabeo $\hat{\psi}$ (véase Figura 6.2 c)) respecto al sistema de referencia inercial O.

Con los parámetros $\hat{x}_0$, $\hat{y}_0$, $\hat{\varphi}_0$, $\hat{\theta}$ y $\hat{\psi}$ se pueden estimar las componentes del vector de posicionamiento en 3D $\mathbf{p_0}$ del robot. La metodología propuesta en este trabajo para estimar el vector $\mathbf{p_0}$ se presenta con detalle en las próximas secciones.

6.4. Posición y Orientación vía Odometría

En aplicaciones basadas en robots móviles diferenciales, es típico obtener la posición y la orientación del robot cuando éste se traslada sobre una superficie plana mediante odometría (Abbas et al. [1], Jha and Kumar [60]). Los codificadores de las ruedas permiten obtener la distancia recorrida por cada una de ellas, en este caso d_l para la rueda izquierda y d_r para la derecha. Con estas medidas y conocida la distancia entre ambas ruedas L, se puede obtener una estimación

de la posición y la orientación del robot Figura 6.3 a través de las expresiones definidas en tiempo discreto por las Ecuaciones 6.1 - 6.3.

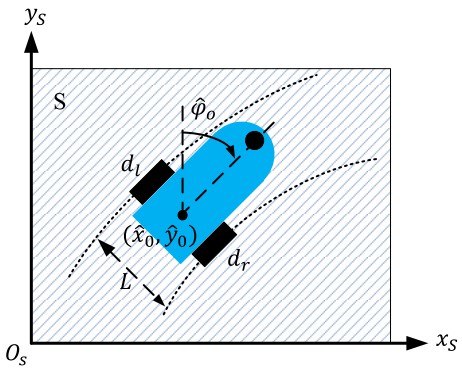

Figura 6.3: Estimación de la posición y la orientación del robot vía odometría.

$$\hat{\varphi}_o[n] = \hat{\varphi}_o[n-1] + \frac{d_l[n] - d_r[n]}{L} \tag{6.1}$$

$$\hat{x}_o[n] = \hat{x}_o[n-1] + d_c[n] sin(\hat{\varphi}_o[n]) \tag{6.2}$$

$$\hat{y}_o[n] = \hat{y}_o[n-1] + d_c[n] cos(\hat{\varphi}_o[n]) \tag{6.3}$$

donde

$d_l[n]$: Distancia recorrida por la rueda izquierda en cada periodo de medida.
$d_r[n]$: Distancia recorrida por la rueda derecha en cada periodo de medida.
$d_c[n]$: Distancia media definida como $d_c[n] = \frac{d_l[n]+d_r[n]}{2}$.
L: Distancia entre las ruedas.

De esta forma, se pueden estimar los parámetros $\hat{x}_0[n]$, $\hat{y}_0[n]$, $\hat{\varphi}_0[n]$ que posteriormente serán utilizados para obtener el posicionamiento 3D del robot.

6.5. Estimación de la Inclinación

El acelerómetro es sensible a la aceleración total **a** que experimenta el robot. Esta magnitud se compone de la aceleración inercial, la gravitatoria g, la centrípeta y la tangencial. El sensor mide las tres componentes de la aceleración total $\mathbf{a} = (a_x \ a_y \ a_z)^T$ con respecto al sistema de referencia situado en el centro de gravedad del robot O_R (generalmente el acelerómetro está situado en un punto muy próximo al centro de gravedad). En los robots diferenciales considerados en este trabajo, la aceleración gravitatoria es la componente más importante de la

aceleración total, por tanto, el resto de aceleraciones no se tendrán en cuenta a la hora de calcular la inclinación.

Por otro lado, el robot puede realizar tres rotaciones diferentes: la elevación $\hat{\theta}$ a través del eje Ox, el balanceo $\hat{\psi}$ sobre el eje Oy y la orientación $\hat{\varphi}$ sobre el Oz, considerando siempre estas medidas respecto al sistema de referencia inercial O. Con estas hipótesis, el sensor del robot mide la aceleración de la gravedad g (el resto de aceleraciones se consideran irrelevantes) con respecto al sistema de referencia del robot O_R (véase la Figura 6.4).

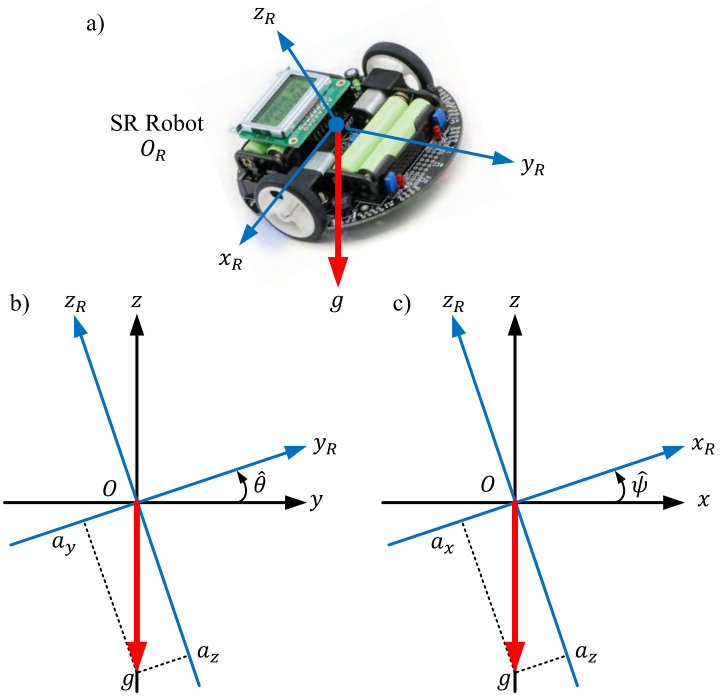

Figura 6.4: Rotaciones respecto al sistema inercial. a) Sistema de referencia del robot. b) Elevación. c) Balanceo.

Respecto al sistema de referencia inercial O la aceleración medida siempre tiene el mismo valor $(0 \;\; 0 \;\; -g)^T$. Por tanto, si se relacionan las medidas de la aceleración en ambos sistemas de referencia O y O_R se obtiene la ecuación:

$$\begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix} = \mathbf{R}(\hat{\theta}, \hat{\psi}, \hat{\varphi}) \cdot \begin{pmatrix} 0 \\ 0 \\ -g \end{pmatrix} \tag{6.4}$$

La matriz de rotación $\mathbf{R}(\hat{\theta}, \hat{\psi}, \hat{\varphi})$ depende del orden en que se producen las rotaciones. Para estimar los ángulos $\hat{\theta}$, $\hat{\psi}$ y $\hat{\varphi}$ es necesario conocer el orden en que éstos han variado. Por ejemplo, si se considera un robot que rota primero un

ángulo $\hat{\theta}$ y luego un ángulo $\hat{\psi}$, las medidas en el sensor quedan descritas por la ecuación:

$$\begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix} = \begin{pmatrix} -gcos(\hat{\theta})sin(\hat{\psi}) \\ -gsin(\hat{\theta}) \\ -gcos(\hat{\theta})cos(\hat{\psi}) \end{pmatrix} \tag{6.5}$$

En cambio, si rota primero un ángulo $\hat{\psi}$ y luego un ángulo $\hat{\theta}$, ahora el acelerómetro mediría las aceleraciones descritas por:

$$\begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix} = \begin{pmatrix} -gsin(\hat{\psi}) \\ -gcos(\hat{\psi})sin(\hat{\theta}) \\ -gcos(\hat{\psi})cos(\hat{\theta}) \end{pmatrix} \tag{6.6}$$

Si se comparan las Ecuaciones 6.5 y 6.6, las componentes a_x y a_y son diferentes en cada ecuación, y por tanto, se puede concluir que sin conocer el orden de las rotaciones, el problema está indeterminado. Si el robot dispusiera de giróscopos, el orden de las rotaciones podrían conocerse y por tanto los ángulos de rotación se podrían estimar. Como se ha mencionado anteriormente, los robots de bajo coste suele disponer de acelerómetros, pero no así de giróscopos. Por esta razón, de forma general en este tipo de robots el ángulo de elevación $\hat{\theta}$ y el ángulo de alabeo $\hat{\psi}$ no pueden estimarse de forma directa. Con estas consideraciones, se propondrá un algoritmo que estime el posicionamiento en 3D de los robots móviles teniendo en cuenta estas limitaciones.

6.6. Algoritmo de Estimación del Posicionamiento en 3D

En la Sección 6.5 han sido analizadas las restricciones que poseen los robots móviles de bajo coste para estimar el ángulo de elevación y el ángulo de alabeo. Para evitar estas limitaciones, en el presente trabajo y de cara a hacer una estimación del posicionamiento 3D de este tipo de robots, se consideran las siguientes hipótesis:

1. Los robots solo pueden desplazarse sobre una superficie plana.
2. La superficie puede tener un ángulo de elevación $\hat{\theta}$ o un ángulo de alabeo $\hat{\psi}$, pero no puede tener ambas rotaciones al mismo tiempo.

Con estas consideraciones, para estimar la posición del robot en 3D se ha propuesto un algoritmo cuya arquitectura se muestra en la Figura 6.5.

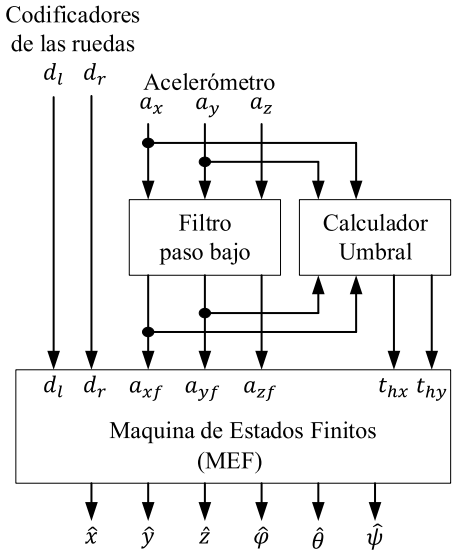

Figura 6.5: Arquitectura del algoritmo de estimación de posicionamiento.

El algoritmo definido se compone de los siguientes bloques:

1. Las señales de entrada $d_l[n]$ y $d_r[n]$ que se obtienen directamente de los codificadores de las ruedas y las señales $a_x[n]$, $a_y[n]$ y $a_z[n]$ que se obtienen directamente del acelerómetro.

2. Un filtro paso bajo para eliminar el ruido eléctrico y mecánico que afecta a la salida de este sensor. La salida de este bloque son las señales del acelerómetro filtradas $a_{xf}[n]$, $a_{yf}[n]$ y $a_{zf}[n]$.

3. Un calculador de umbral que determina los umbrales $t_{hx}[n]$ y $t_{hy}[n]$. Estas señales se utilizarán en la Máquina de Estados Finitos (MEF) para definir los cambios entre los diferentes estados.

4. Una MEF que estima los parámetros de posición y orientación del robot.

5. Finalmente, las señales de salida $\hat{x}[n]$, $\hat{y}[n]$, $\hat{z}[n]$, $\hat{\varphi}[n]$, $\hat{\theta}[n]$ y $\hat{\psi}[n]$.

En las siguientes secciones de analiza con detalle cada uno de los bloques funcionales que componen este algoritmo (véanse también los trabajos Socas et al. [130] y Socas et al. [132]).

6.6.1. Filtro Paso Bajo

Filtrar paso bajo las señales del acelerómetro es un buen método para eliminar el ruido mecánico y eléctrico que afecta a este sensor (Seifert and Camacho [122]). En este trabajo se ha propuesto el filtro IIR descrito por la ecuación

$$a_{if}[n] = (1 - \alpha)a_i[n] + \alpha a_{if}[n - 1] \tag{6.7}$$

para $i = \{x, y, z\}$.

Las entradas del filtro son las señales del acelerómetro $a_x[n]$, $a_y[n]$ y $a_z[n]$, las salidas son las señales filtradas $a_{xf}[n]$, $a_{yf}[n]$ y $a_{zf}[n]$. El parámetro α fija la frecuencia de corte y el retardo de grupo del filtro. En el proceso de diseño, α debe ser seleccionado de tal forma que se obtenga un compromiso entre el nivel de ruido cancelado y el retraso en las medidas de la aceleración.

6.6.2. Calculador de Umbral

Como se ha mencionado anteriormente, el algoritmo de posicionamiento se basa en una máquina de estados finitos (MEF). Los estados, como se definirá en la Sección 6.6.3, dependen de los valores del ángulo de elevación $\hat{\theta}[n]$ y del ángulo de alabeo $\hat{\psi}[n]$. Cuando el robot está en una superficie con $\hat{\theta}[n] \approx 0$ o $\hat{\psi}[n] \approx 0$, el ruido en el acelerómetro puede provocar cambios de estados y hacer que el sistema se vuelva inestable. Por esta razón, el algoritmo necesita un sistema de umbrales dinámico que controle los cambios entre los estados teniendo en cuenta el ruido. En la Figura 6.6 se muestra el calculador de umbral propuesto en este trabajo.

Como se explicará posteriormente, cuando el ángulo de elevación y el ángulo de alabeo están cercanos a cero ($\hat{\theta}[n] \approx 0$, $\hat{\psi}[n] \approx 0$), los valores de las aceleraciones $a_{xf}[n]$ y $a_{yf}[n]$ también deberían ser cero excepto por el ruido en el sensor. Por esta razón, el ruido en estas medidas serán consideradas en el calculador de umbral.

El ruido $n_x[n]$ y $n_y[n]$ en las señales del acelerómetro $a_x[n]$ y $a_y[n]$ pueden estimarse por la diferencia entre las señales del acelerómetro y sus señales filtradas de la forma:

$$n_x[n] = a_x[n] - a_{xf}[n] \tag{6.8}$$

$$n_y[n] = a_y[n] - a_{yf}[n] \tag{6.9}$$

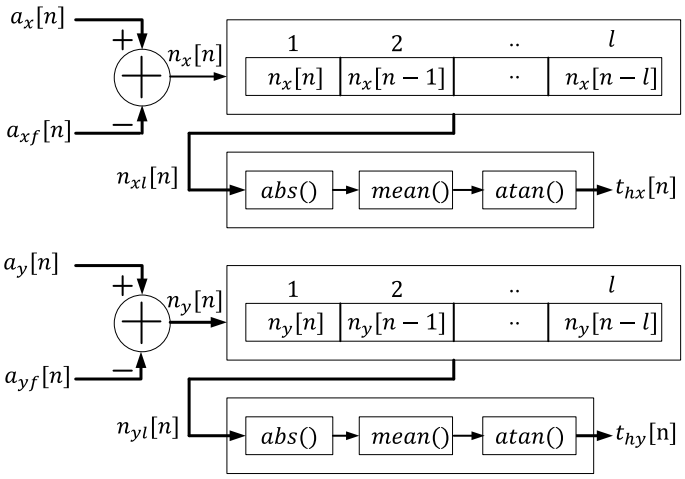

Figura 6.6: Esquema del calculador de umbral propuesto.

Posteriormente se ha utilizado una media móvil para obtener un valor estable del ruido. Finalmente se aplica la función $atan()$ para obtener las magnitudes angulares a partir del ruido y determinar los umbrales dinámicos según las expresiones:

$$t_{hx}[n] = atan\left(\frac{1}{l}\sum_{k=1}^{l} abs(n_x[k])\right) \tag{6.10}$$

$$t_{hy}[n] = atan\left(\frac{1}{l}\sum_{k=1}^{l} abs(n_y[k])\right) \tag{6.11}$$

En este sentido, el sistema propuesto obtiene dos umbrales dinámicos $t_{hx}[n]$ y $t_{hy}[n]$, uno para cada señal de aceleración. Por otro lado, este sistema dispone de un grado de libertad, el parámetro l, que debe ser sintonizado dependiendo de como varía la dinámica de la variable aceleración.

6.6.3. Máquina de Estados Finitos

En la arquitectura propuesta se ha incluido una máquina de estados finitos (MEF) para estimar la posición 3D del robot. Este sistema se presenta en la Figura 6.5. Las entradas son las señales de los codificadores de las ruedas, las aceleraciones filtradas y los umbrales dinámicos. Las salidas son las posiciones 3D del robot $\hat{x}$, $\hat{y}$, $\hat{z}$, $\hat{\varphi}$, $\hat{\theta}$ y $\hat{\psi}$. Como se ha mencionado anteriormente, el robot solo puede navegar sobre una superficie plana y además esta superficie solo puede tener un ángulo de elevación o un ángulo de alabeo respecto al sistema inercial.

Por tanto, la principal condición de contorno es que no haya elevación y alabeo al mismo tiempo. Teniendo en cuenta estas consideraciones, se ha definido el comportamiento de la MEF.

Tal como se presenta en la Figura 6.7, la MEF se compone de cinco estados y cinco condiciones de transición entre dichos estados.

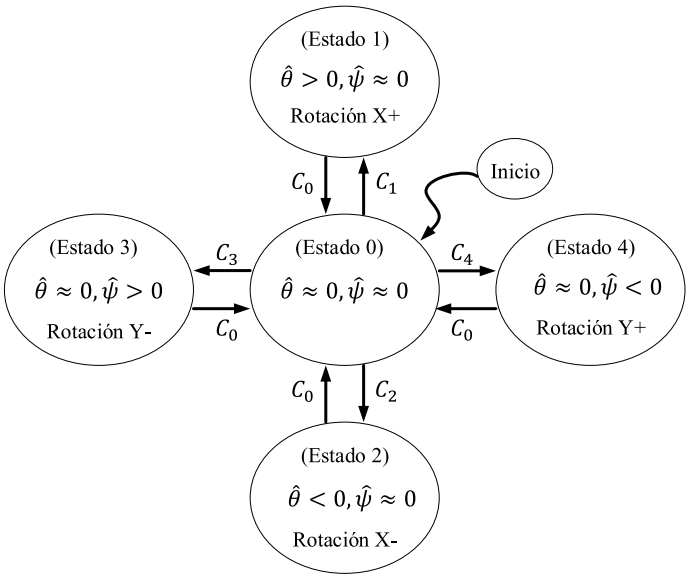

Figura 6.7: Máquina de estados finitos.

El significado de cada uno de los estados es el siguiente:

- Estado 0. El robot está en la superficie sin ángulo de elevación ($\hat{\theta} \approx 0$) y sin ángulo de alabeo ($\hat{\psi} \approx 0$) respecto al sistema inercial O.
- Estado 1. El robot está sobre una superficie con un ángulo de elevación positivo ($\hat{\theta} > 0$) y sin ángulo de alabeo ($\hat{\psi} \approx 0$).
- Estado 2. El robot está sobre una superficie con un ángulo de elevación negativo ($\hat{\theta} < 0$) y sin ángulo de alabeo ($\hat{\psi} \approx 0$).
- Estado 3. El robot está sobre una superficie con un ángulo de alabeo positivo ($\hat{\psi} > 0$) y sin ángulo de elevación ($\hat{\theta} \approx 0$).
- Estado 4. El robot está sobre una superficie con un ángulo de alabeo negativo ($\hat{\psi} < 0$) y sin ángulo de elevación ($\hat{\theta} \approx 0$).

Por otro lado, se asume que el sistema trabaja en tiempo discreto y que en cada periodo de muestreo están disponibles las medidas de los sensores $d_l[n]$, $d_r[n]$, $a_{xf}[n]$, $a_{yf}[n]$ y $a_{zf}[n]$.

Las condiciones de transición entre estados (C_0, C_1, C_2, C_3 y C_4) se han definido en función de las componentes de la aceleración filtradas ($a_{xf}[n]$, $a_{yf}[n]$ y $a_{zf}[n]$), los valores de umbral ($t_{hx}[n]$ y $t_{hy}[n]$) y dos bandas de guarda positivas H y h. Los parámetros H y h crean dos bandas de guarda alrededor de los umbrales dinámicos con una histéresis para evitar inestabilidades en el sistema. Estas dos bandas originan unos nuevos umbrales ($t_{hy}[n]+H$ o $t_{hx}[n]+H$) cuando los ángulo de elevación y alabeo crecen y otro umbral ($t_{hy}[n]+h$ o $t_{hx}[n]+h$) cuando esos mismo ángulos decrecen (véase Figura 6.8). Considerando estos parámetros, las condiciones de transición entre estados se definen según las Ecuaciones 6.12 - 6.18.

$$\theta_{tilt} = atan2\left(a_{yf}[n]/a_{zf}[n]\right) \tag{6.12}$$

$$\psi_{tilt} = atan2\left(a_{xf}[n]/a_{zf}[n]\right) \tag{6.13}$$

$$\begin{array}{c} C_0 : (abs(\theta_{tilt}) \leq t_{hy}[n] + h) \\ \&(abs(\psi_{tilt}) \leq t_{hx}[n] + h) \end{array} \tag{6.14}$$

$$C_1 : \theta_{tilt} > t_{hy}[n] + H \tag{6.15}$$

$$C_2 : \theta_{tilt} < -t_{hy}[n] - H \tag{6.16}$$

$$C_3 : \psi_{tilt} > t_{hx}[n] + H \tag{6.17}$$

$$C_4 : \psi_{tilt} < -t_{hx}[n] - H \tag{6.18}$$

donde la función $atan2()$ se define según la Ecuación 6.19 para evitar singularidades cuando $x \approx 0$

$$atan2(y/x) = \begin{cases} atan(y/x) & si \quad x > 0 \\ \pi + atan(y/x) & si \quad y \geq 0, x < 0 \\ -\pi + atan(y/x) & si \quad y < 0, x < 0 \\ \frac{\pi}{2} & si \quad y > 0, x = 0 \\ -\frac{\pi}{2} & si \quad y < 0, x = 0 \\ 0 & si \quad y = 0, x = 0 \end{cases} \tag{6.19}$$

Los parámetros H y h deben sintonizarse teniendo en cuenta el nivel de ruido.

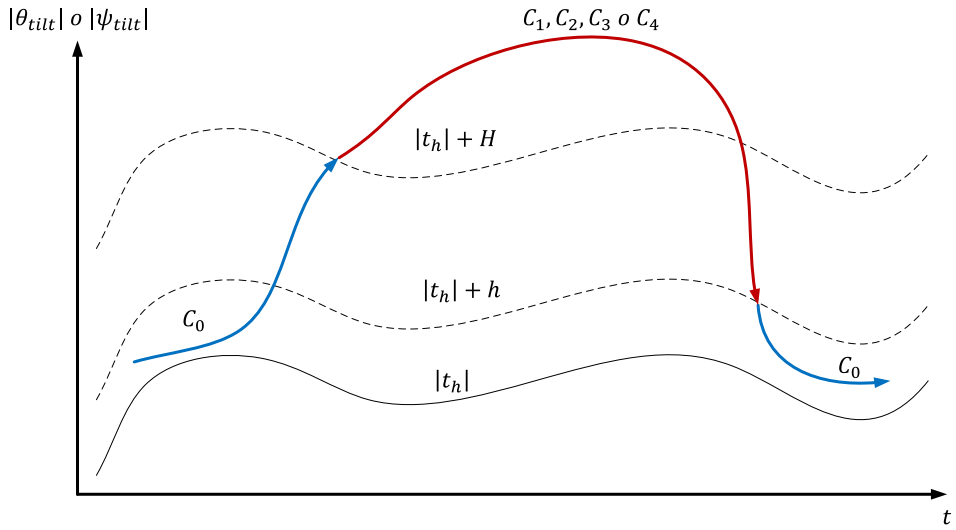

Figura 6.8: Significado de las bandas de guarda H y h.

6.6.4. Estimación de la Posición y la Orientación

En cada estado de la MEF se hace una estimación de la posición 3D del robot. En el Estado 0 se aplica la odometría clásica. Para el resto de estados se aplica una combinación entre técnicas de odometría y estimación de inclinación a partir de las señales del acelerómetro. El objetivo final de la estimación de posicionamiento 3D es obtener el vector $\mathbf{p_0} = (\hat{x}[n]\ \ \hat{y}[n]\ \ \hat{z}[n]\ \ \hat{\varphi}[n]\ \ \hat{\theta}[n]\ \ \hat{\psi}[n])^T$. Dependiendo del estado en que se encuentre la MEF se aplicarán unas ecuaciones u otras según se mostrará en las siguientes secciones.

Estimación en el Estado 0

En el Estado 0, debido a que los ángulos de inclinación son cero, se aplican las técnicas de odometría clásica para estimar el posicionamiento según las Ecuaciones 6.20 - 6.25.

$$\hat{\varphi}[n] = \hat{\varphi}[n-1] + \hat{\varphi}_o[n] \tag{6.20}$$

$$\hat{x}[n] = \hat{x}[n-1] + d_c[n]sin(\hat{\varphi}[n]) \tag{6.21}$$

$$\hat{y}[n] = \hat{y}[n-1] + d_c[n]cos(\hat{\varphi}[n]) \tag{6.22}$$

$$\hat{z}[n] = \hat{z}[n-1] \tag{6.23}$$

$$\hat{\theta}[n] = 0 \tag{6.24}$$

$$\hat{\psi}[n] = 0 \tag{6.25}$$

donde

$\hat{\varphi}_o[n]$: Estimación de la orientación mediante odometría $\hat{\varphi}_o[n] = \frac{d_l[n]-d_r[n]}{L}$

$d_c[n]$: Distancia media $d_c[n] = \frac{d_l[n]+d_r[n]}{2}$.

L: Distancia entre las ruedas.

Estimación en los Estados 1 y 2

El ángulo de elevación $\hat{\theta}[n]$ puede calcularse en el Estado 1 mediante la Ecuación 6.26 y en el Estado 2 a través de la Ecuación 6.27.

$$\hat{\theta}[n] = atan2\left(\frac{\sqrt{a_{xf}[n]^2 + a_{yf}[n]^2}}{-a_{zf}[n]}\right) \tag{6.26}$$

$$\hat{\theta}[n] = atan2\left(\frac{\sqrt{a_{xf}[n]^2 + a_{yf}[n]^2}}{a_{zf}[n]}\right) \tag{6.27}$$

En ambos estados las componentes $\hat{\varphi}[n]$, $\hat{x}[n]$, $\hat{y}[n]$ y $\hat{\psi}[n]$ tienen las mismas expresiones según se muestra en las Ecuaciones 6.28 - 6.31.

$$\hat{\varphi}[n] = \hat{\varphi}[n-1] + \hat{\varphi}_o[n] \tag{6.28}$$

$$\hat{x}[n] = \hat{x}[n-1] + d_c[n]sin(\hat{\varphi}[n]) \tag{6.29}$$

$$\hat{y}[n] = \hat{y}[n-1] + d_c[n]cos(\hat{\varphi}[n])cos(\theta_a) \tag{6.30}$$

$$\hat{\psi}[n] = 0 \tag{6.31}$$

donde $\theta_a = abs(\hat{\theta}[n])$

La componente $\hat{z}[n]$ se calcula para el Estado 1 mediante la Ecuación 6.32 y para el Estado 2 mediante la Ecuación 6.33.

$$\hat{z}[n] = \hat{z}[n-1] + d_c[n]sin(\hat{\varphi}[n])sin(\theta_a) \tag{6.32}$$

$$\hat{z}[n] = \hat{z}[n-1] - d_c[n]sin(\hat{\varphi}[n])sin(\theta_a) \tag{6.33}$$

Estimación en los Estados 3 y 4

Por otro lado, el ángulo de alabeo se puede estimar en el Estado 3 mediante la Ecuación 6.34 y con la Ecuación 6.35 para el Estado 4.

$$\hat{\psi}[n] = atan2\left(\frac{\sqrt{a_{xf}[n]^2 + a_{yf}[n]^2}}{-a_{zf}[n]}\right) \tag{6.34}$$

$$\hat{\psi}[n] = atan2\left(\frac{\sqrt{a_{xf}[n]^2 + a_{yf}[n]^2}}{a_{zf}[n]}\right) \tag{6.35}$$

En ambos estados, las componentes $\hat{\psi}[n]$, $\hat{x}[n]$, $\hat{y}[n]$ y $\hat{\theta}[n]$ se obtienen de las expresiones 6.36 - 6.39.

$$\hat{\varphi}[n] = \hat{\varphi}[n-1] + \hat{\varphi}_o[n] \tag{6.36}$$

$$\hat{x}[n] = \hat{x}[n-1] + d_c[n]sin(\hat{\varphi}[n])cos(\psi_a) \tag{6.37}$$

$$\hat{y}[n] = \hat{y}[n-1] + d_c[n]cos(\hat{\varphi}[n]) \tag{6.38}$$

$$\hat{\theta}[n] = 0 \tag{6.39}$$

donde $\psi_a = abs(\hat{\psi}[n])$

Finalmente, la componente $\hat{z}[n]$ se obtiene en el Estado 3 mediante la Ecuación 6.40 y para el Estado 4 mediante la Ecuación 6.41.

$$\hat{z}[n] = \hat{z}[n-1] + d_c[n]sin(\hat{\varphi}[n])sin(\psi_a) \tag{6.40}$$

$$\hat{z}[n] = \hat{z}[n-1] - d_c[n]sin(\hat{\varphi}[n])sin(\psi_a) \tag{6.41}$$

6.7. Sistema propuesto para la Sincronización de la Orientación

El ángulo de orientación $\hat{\varphi}[n]$ es la variable más importante de los parámetros de navegación debido a su influencia en los errores acumulados en la estimación de posicionamiento. En el sistema propuesto en este trabajo, cuando el robot está en el Estado 0 la única forma de determinar este parámetro es mediante odometría ($\hat{\varphi}[n] = \hat{\varphi}[n-1] + \hat{\varphi}_0[n]$, donde $\hat{\varphi}_0[n] = \frac{d_l[n] - d_r[n]}{L}$). Por otro lado, cuando el robot está navegando sobre una superficie con un ángulo de elevación $\hat{\theta}[n] \neq 0$ o con un ángulo de alabeo $\hat{\psi}[n] \neq 0$ (Estados 1, 2, 3 y 4), los valores de $a_{xf}[n]$ y $a_{yf}[n]$ se pueden utilizar también para estimar el ángulo de orientación $\hat{\varphi}[n]$ del robot.

Cuando la máquina de estados finitos está en los Estados 1, 2, 3 o 4, se

pueden obtener dos estimaciones del ángulo de orientación $\hat{\varphi}[n]$. Uno a través de odometría, $\hat{\varphi}_0[n]$ y otro a través de las medidas del acelerómetro, $\hat{\varphi}_a[n]$ según la Ecuaciones 6.42 - 6.45.

En el Estado 1,

$$\hat{\varphi}_a[n] = \begin{cases} atan2\left(\frac{|a_{xf}|}{|a_{yf}|}\right) & \text{if } (a_{xf}>0)\&(a_{yf}\leq 0) \\ \frac{\pi}{2} + atan2\left(\frac{|a_{yf}|}{|a_{xf}|}\right) & \text{if } (a_{xf}>0)\&(a_{yf}>0) \\ \pi + atan2\left(\frac{|a_{xf}|}{|a_{yf}|}\right) & \text{if } (a_{xf}\leq 0)\&(a_{yf}>0) \\ \frac{3\pi}{2} + atan2\left(\frac{|a_{yf}|}{|a_{xf}|}\right) & \text{if } (a_{xf}\leq 0)\&(a_{yf}\leq 0) \end{cases} \tag{6.42}$$

en el Estado 2,

$$\hat{\varphi}_a[n] = \begin{cases} \pi + atan2\left(\frac{|a_{xf}|}{|a_{yf}|}\right) & \text{if } (a_{xf}>0)\&(a_{yf}\leq 0) \\ \frac{3\pi}{2} + atan2\left(\frac{|a_{yf}|}{|a_{xf}|}\right) & \text{if } (a_{xf}>0)\&(a_{yf}>0) \\ atan2\left(\frac{|a_{xf}|}{|a_{yf}|}\right) & \text{if } (a_{xf}\leq 0)\&(a_{yf}>0) \\ \frac{\pi}{2} + atan2\left(\frac{|a_{yf}|}{|a_{xf}|}\right) & \text{if } (a_{xf}\leq 0)\&(a_{yf}\leq 0) \end{cases} \tag{6.43}$$

en el Estado 3,

$$\hat{\varphi}_a[n] = \begin{cases} \frac{\pi}{2} + atan2\left(\frac{|a_{xf}|}{|a_{yf}|}\right) & \text{if } (a_{xf}>0)\&(a_{yf}\leq 0) \\ \pi + atan2\left(\frac{|a_{yf}|}{|a_{xf}|}\right) & \text{if } (a_{xf}>0)\&(a_{yf}>0) \\ \frac{3\pi}{2} + atan2\left(\frac{|a_{xf}|}{|a_{yf}|}\right) & \text{if } (a_{xf}\leq 0)\&(a_{yf}>0) \\ atan2\left(\frac{|a_{yf}|}{|a_{xf}|}\right) & \text{if } (a_{xf}\leq 0)\&(a_{yf}\leq 0) \end{cases} \tag{6.44}$$

y finalmente, en el Estado 4

$$\hat{\varphi}_a[n] = \begin{cases} \frac{3\pi}{2} + atan2\left(\frac{|a_{xf}|}{|a_{yf}|}\right) & \text{if } (a_{xf}>0)\&(a_{yf}\leq 0) \\ atan2\left(\frac{|a_{yf}|}{|a_{xf}|}\right) & \text{if } (a_{xf}>0)\&(a_{yf}>0) \\ \frac{\pi}{2} + atan2\left(\frac{|a_{xf}|}{|a_{yf}|}\right) & \text{if } (a_{xf}\leq 0)\&(a_{yf}>0) \\ \pi + atan2\left(\frac{|a_{yf}|}{|a_{xf}|}\right) & \text{if } (a_{xf}\leq 0)\&(a_{yf}\leq 0) \end{cases} \tag{6.45}$$

donde $a_{xf} = a_{xf}[n]$ and $a_{yf} = a_{yf}[n]$.

Es importante destacar que esta nueva estimación del ángulo de orientación

$\hat{\varphi}_a[n]$ obtenida de los acelerómetros es una medida absoluta y que no depende de los valores previos. Por esta razón, el utilizar esta nueva medida para estimar la orientación ayuda a eliminar los errores acumulados que producen las metodologías basadas en odometría.

Por otro lado, debido a que las señales del acelerómetro han sido filtradas para eliminar el ruido del sensor, estas medidas sufren un retraso que depende del parámetro α del filtro (ver Sección 6.6.1). En este caso, si la estimación de la orientación del robot se obtiene mediante $\hat{\varphi}[n] = \hat{\varphi}_a[n]$ en una situación en la que el robot esté describiendo una trayectoria circular, esta será una mala estimación de la orientación debido al retraso que produce el filtro. Sin embargo, un tiempo después de que el robot esté siguiendo una trayectoria en línea recta, las medidas de aceleración del robot son estables (debido a que las medidas de $a_{xf}[n]$, $a_{yf}[n]$ y $a_{zf}[n]$ no varían) y por tanto el retraso que introduce el filtro no afecta a la estimación. Teniendo en cuenta estas consideraciones, en la Figura 6.9 se presenta el sistema de sincronización para la orientación propuesto.

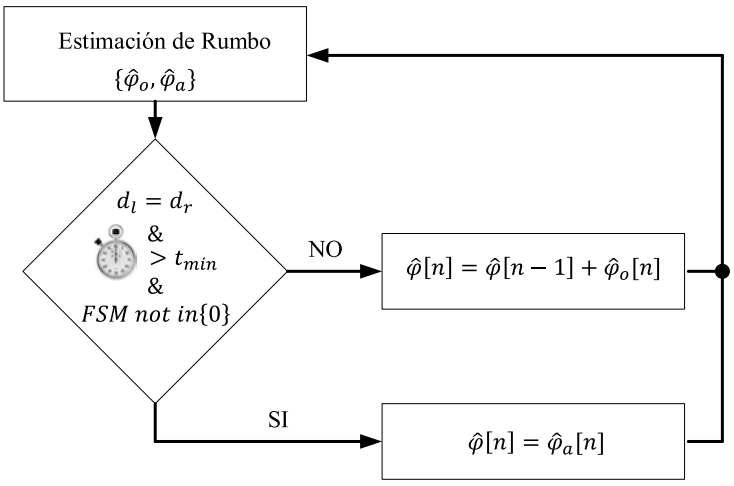

Figura 6.9: Sistema de sincronización de la orientación.

La filosofía de funcionamiento de este sistema es el siguiente:

1. En cada periodo de muestreo se obtienen dos estimaciones del ángulo de orientación, una mediante odometría $\hat{\varphi}_o[n]$ y otra a través de las medidas del acelerómetro $\hat{\varphi}_a[n]$ mediante las Ecuaciones 6.42 - 6.45.

2. Seguidamente, si el robot está siguiendo una trayectoria en línea recta $d_l[n] = d_r[n]$ después de un cierto tiempo $t > t_{min}$ y si la MEF está en un Estado diferente al 0, se obtiene la estimación del ángulo de orientación

mediante las medidas del acelerómetro $\hat{\varphi}[n] = \hat{\varphi}_a[n]$. En caso contrario, se siguen considerando las medidas basadas en odometría $\hat{\varphi}[n] = \hat{\varphi}[n-1]+\hat{\varphi}_o[n]$.

Por último, el parámetro t_{min} debe escogerse en función del valor de α utilizado en el filtro (véanse los trabajos Socas et al. [130] y Socas et al. [132] para más detalles).

6.8. Conclusiones

En el presente capítulo se ha desarrollado un algoritmo de posicionamiento que mejora de forma considerable las estimaciones de posición y orientación respecto a los algoritmos clásicos basados en odometría. El sistema combina las medidas inerciales con las odométricas para conseguir un posicionamiento en 3D. Además, permite sincronizar la orientación del robot mediante medidas absolutas de aceleración, y reducir de esta forma, los errores acumulativos que poseen las técnicas clásicas de posicionamiento. El algoritmo planteado se ha formulado para los robots móviles diferenciales, aunque la arquitectura de este sistema permite adaptarlo de forma sencilla y directa a cualquier otro tipo de robot móvil.

Capítulo 7

Resultados Experimentales

Una vez analizadas las contribuciones de este trabajo de investigación en los capítulos anteriores, en el presente capítulo se pretende validar estas ideas de forma empírica. Para ello, se analizarán diferentes experimentos con el apoyo del laboratorio de experimentación desarrollado para esta tesis (véanse los Apéndice A, B y C). La idea general que se aplica en estos experimentos es comparar las soluciones planteadas en este trabajo con implementaciones clásicas que resuelvan el mismo problema. Por tanto, los experimentos analizados contemplan los diferentes algoritmos de navegación planteados y el análisis de su estabilidad. Se analiza también, la eficiencia en el consumo de recursos y las mejoras obtenidas. Respecto al análisis de perturbaciones, se obtienen modelos empíricos del ruido que afecta a este tipo de plataformas y se utilizan éstos posteriormente para fijar los umbrales en el sistema. Finalmente, se analizan de forma experimental la respuesta de los sistemas de posicionamiento planteados.

7.1. Algoritmos Directo al Destino - DD

En el Sección 3.3.2 se ha presentado la implementación del algoritmo Directo al Destino (DD) sobre la arquitectura de control basada en eventos propuesta en este trabajo. De cara a analizar su comportamiento en un entorno real, en la plataforma de experimentación se han planteado dos escenarios. El primer escenario consiste en resolver el problema de navegación mediante un sistema de control discreto equivalente al sistema propuesto que sirva de marco de comparación. En el segundo, se aplica al mismo problema de navegación la arquitectura de control basada eventos propuesta en este trabajo de investigación. Los parámetros de

ambos algoritmos se presentan en la Tabla 7.1.

Tabla 7.1: Escenarios de análisis experimental del algoritmo DD.

Escenario	Arquitectura	f_s(Hz)	K_ρ(1/s)	K_α(1/s)	$\bar{e}_\alpha$ (rad)	e_ρ (cm)
1	Discreto	10	0,2	2,5	-	-
2	Eventos	-	0,2	2,5	0,2	1

Con esta configuración, a los algoritmos se les marca como señal de consigna el destino $x_g = 50$ cm e $y_g = 40$ cm, que debe ser alcanzado partiendo de la posición origen $(0, 0)$. Con el sistema de seguimiento basado en la herramienta *Tracker*, Figura 7.1, se analizan las trayectorias y el posicionamiento de los robots. Finalmente, con la información que gestiona el controlador se obtienen las velocidades de los robots y la actividad de ambas estructuras de control. Esta última magnitud se obtiene mediante la medida de los periodos de muestreo y el número de eventos generados por cada sistema respectivamente.

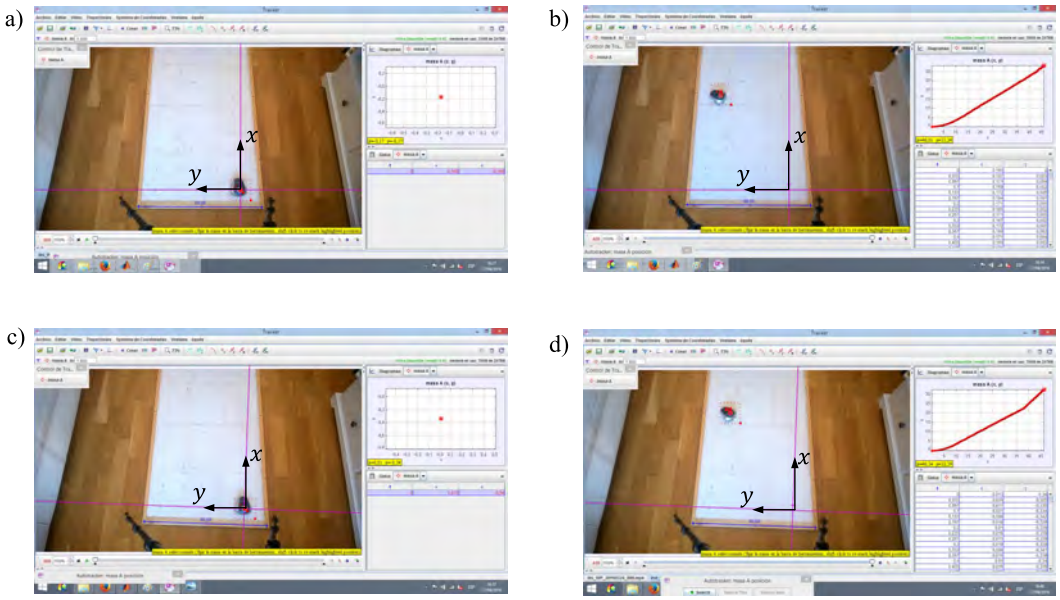

Figura 7.1: Análisis de posicionamiento con *Tracker*. a), b) escenario 1. c), d) escenario 2.

Tal como se muestra en la Figura 7.2, tanto en la solución discreta (Figura 7.2 a)) como en la solución propuesta basada en eventos (Figura 7.2 b)) el problema de navegación se resuelve de forma satisfactoria y sin inestabilidades. El posicionamiento, la orientación y la dinámica de ambos experimentos se presentan en la Figura 7.3. Como se observa en la Figura 7.3 a), las trayectorias seguidas por ambos robots (mediante control discreto y mediante control basado en eventos)

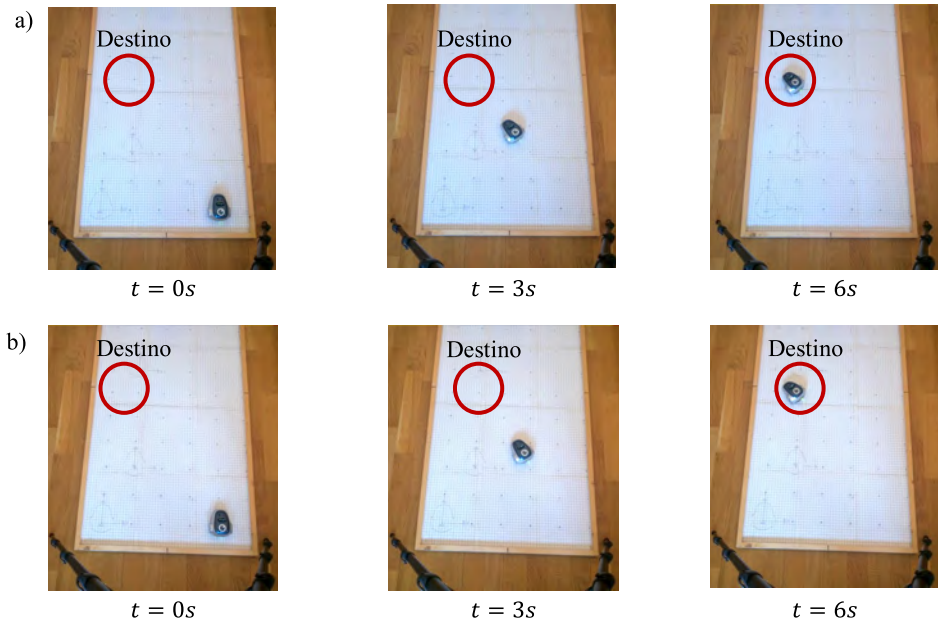

Figura 7.2: Respuesta del algoritmo DD. a) Escenario 1. b) Escenario 2.

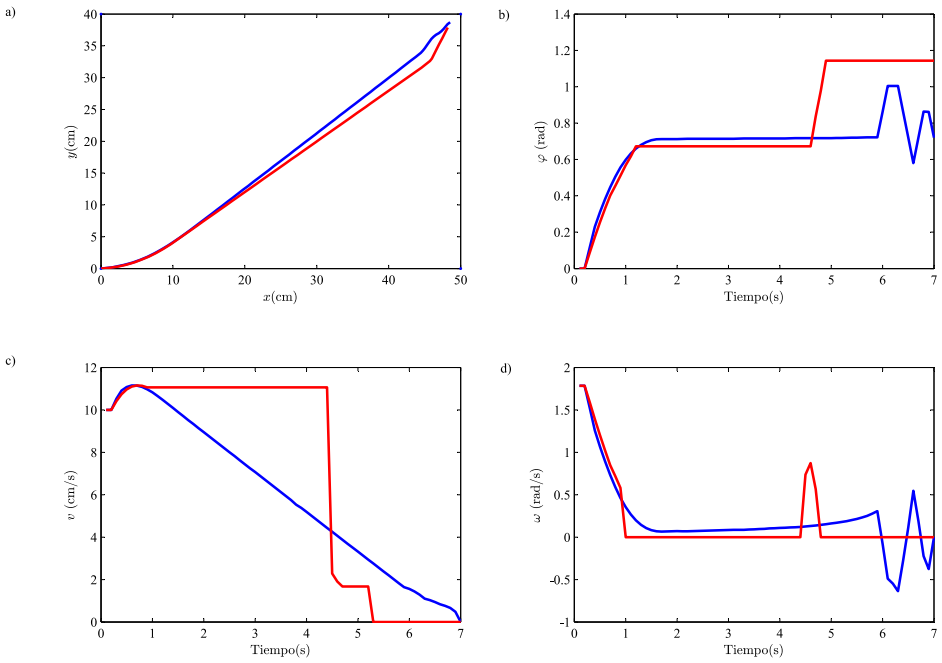

Figura 7.3: Resultados de la posición, orientación, velocidad lineal y velocidad angular. En azul el escenario 1 y en rojo el escenario 2.

son prácticamente similares. Respecto a la orientación (Figura 7.3 b)), el sistema discreto presenta mayor actividad en los instantes finales del recorrido, en cam-

bio el de eventos describe unos cambios de orientación menos pronunciados. Al final del recorrido, ambas soluciones presentan una diferencia de orientación de aproximadamente $0,6$ rad. En este caso, la orientación final no es un requisito del algoritmo, por tanto, se puede concluir que ambos resultados son satisfactorios.

Como siguiente paso, se considerarán ahora las velocidades lineales y angulares de los robots en cada uno de los escenarios analizados. Respecto a la velocidad lineal v de ambos experimentos (Figura 7.3 c)), siguen un comportamiento monótono y decreciente a medida que se aproxima al destino. En este caso, la velocidad lineal del control basado en eventos se anula mucho antes que la velocidad de la solución basada en tiempo discreto. Los tiempos de convergencia T_c van desde $5,2$ s para el caso basado en eventos a los $7,0$ s para el discreto. Se tiene por tanto una mejora en el tiempo de convergencia de $1,8$ s en la solución basada en eventos respecto a la discreta. Respecto a la velocidad angular ω (Figura 7.3 d)), ambas soluciones siguen una comportamiento similar sin variaciones bruscas que comprometan el cambio de rumbo del robot.

En cuanto a la actividad de ambos controladores, tal como se muestra en la Figura 7.4 c) y d), se observa una gran eficiencia de la solución basada en eventos respecto a su arquitectura equivalente basada en tiempo discreto. El número de muestras en la solución discreta es muy superior al número de eventos de la solución propuesta.

Finalmente, si se analiza esta actividad de forma acumulada durante el tiempo que dura el experimento (Figura 7.5), se puede concluir que la actividad del controlador discreto es $5,8$ veces (70 muestras frente a 12 eventos) superior a la solución propuesta en este trabajo.

7.2. Algoritmos para Evitar Obstáculos - EO

Como se mostró en la Sección 3.3.4 de esta tesis se ha propuesto un algoritmo para evitar obstáculos (EO) sobre la arquitectura de control basada en eventos. Al igual que el resto de experimentos, se ha implementado un sistema equivalente en tiempo discreto que sirva de marco de comparación con el sistema desarrollado. En este caso, se ha planteado un problema de navegación con dos robots, de esta forma se favorece que el entorno sea más dinámico y se pueda comprobar así la efectividad en cuanto a la generación de eventos del sistema propuesto. Para el mencionado experimento se han planteado dos escenarios cuyos parámetros se

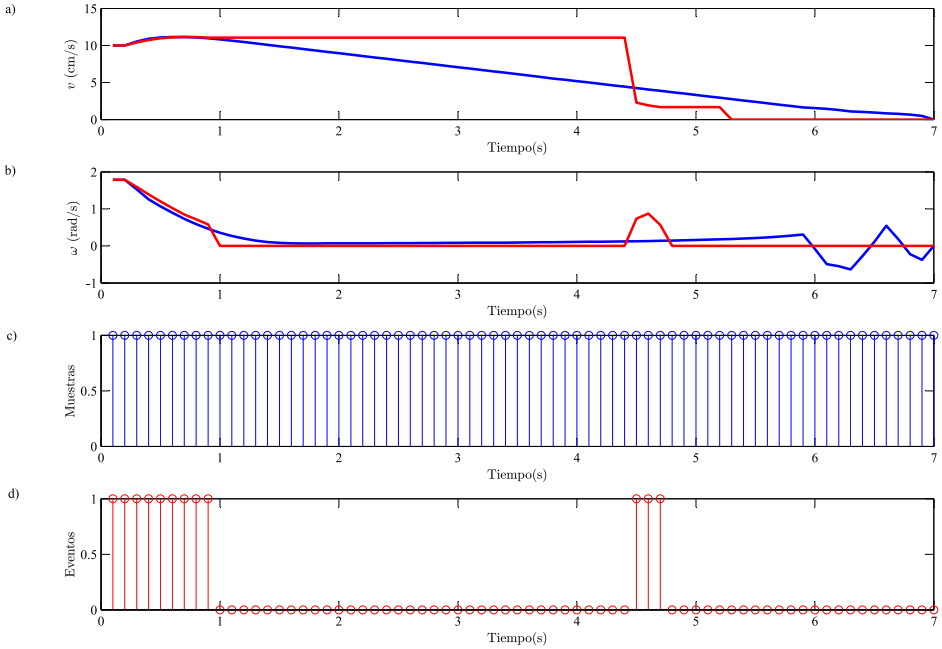

Figura 7.4: Variables dinámicas y actividad del sistema. En azul el escenario 1 y en rojo el escenario 2.

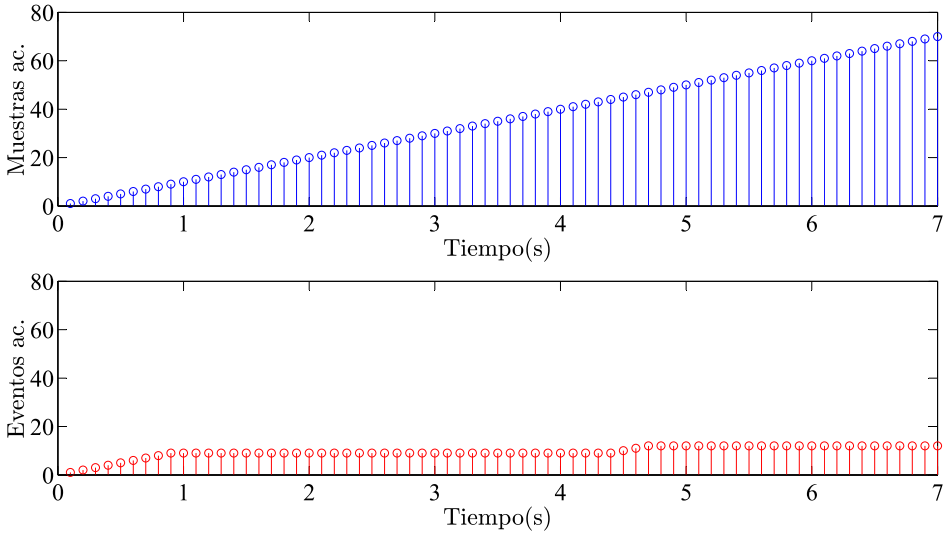

Figura 7.5: Volumen de actividad del controlador acumulado. En azul el escenario 1 y en rojo el escenario 2.

muestran en la Tabla 7.2.

Tal como se planteó en la Ecuación 3.21, la condición de evento tiene dos

Tabla 7.2: Escenarios de análisis experimental del algoritmo EO.

Escenario	Arquitectura	Número de Robots	f_s(Hz)	w(cm)	$\bar{e}_{EO}$(cm)
1	Discreto	2	10	-	-
2	Eventos	2	-	0,0	0,0

parámetros, la señal de referencia w y el *event threshold* $\bar{e}_{EO}$. En este análisis ambos parámetros se han puesto a cero ya que será la situación más desfavorable de cara a la generación de eventos en el sistema. Por otro lado, al algoritmo que se desarrolló para el controlador basado en eventos (Figura 3.16), se le ha incluido una nueva condición C_d. Esta nueva condición le permite trabajar también en un sistema en modo discreto. De esta forma, el mismo algoritmo del controlador sirve en ambos escenarios con activar/desactivar simplemente una de las ramas de éste (véase Figura 7.6).

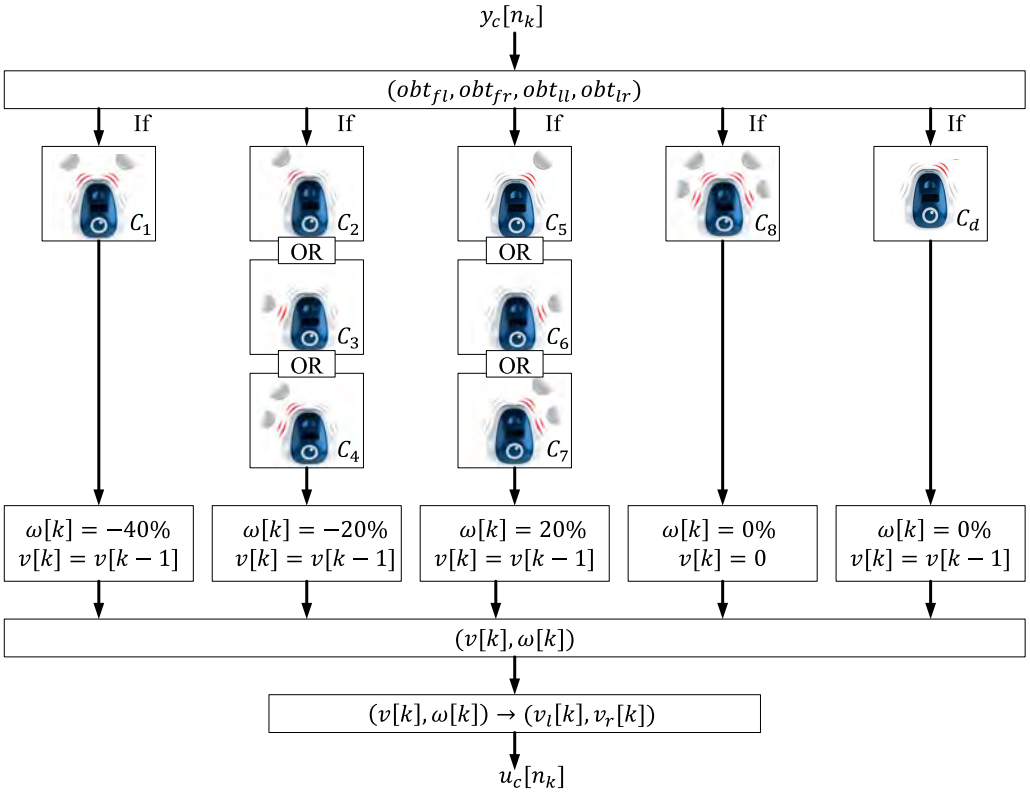

Figura 7.6: Algoritmo de control EO para ambos escenarios.

En ambos escenarios, tal como se observa para el caso discreto (Figura 7.7) como para el sistema basado en eventos (Figura 7.8), el problema se ha resuelto de forma satisfactoria.

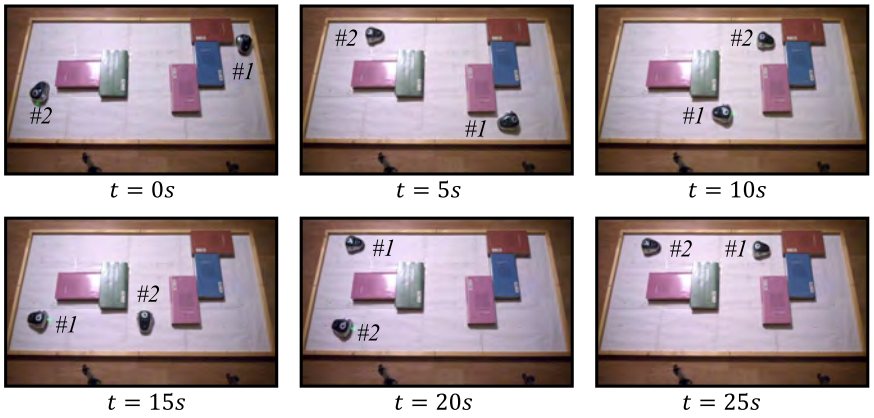

Figura 7.7: Respuesta del algoritmo EO en el escenario 1 (sistema discreto).

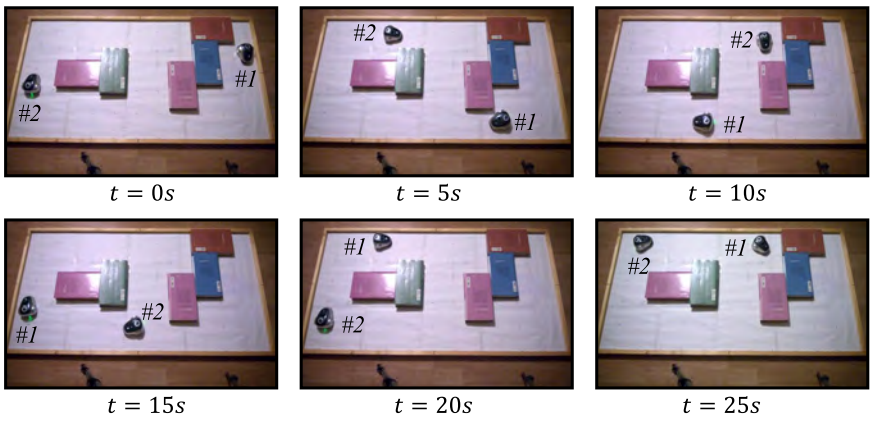

Figura 7.8: Respuesta del algoritmo EO en el escenario 2 (sistema basado en eventos).

Con ayuda de la herramienta *Tracker* se han capturado las trayectorias de ambos robots tanto en el caso discreto (Figura 7.9 a) y b)), como en el caso de que se aplique la arquitectura basada en eventos (Figura 7.9 c) y d)). Tal como se observa en la Figura 7.9, en ambos escenarios las trayectorias seguidas por ambos robots son casi idénticas.

Hasta ahora, ambos algoritmos presentan un comportamiento similar, pero si se analiza la actividad de ambas soluciones se observa que las cosas son muy diferentes. Analizando la Figura 7.10 se puede comprobar que la actividad del sistema discreto genera 20 invocaciones por segundo (10 por cada robot debido a que la frecuencia de muestreo es $f_s = 10$ Hz). En cambio, para el sistema propuesto, el número de invocaciones (eventos), no supera los 15 por segundo en los periodos de máxima actividad.

Finalmente, para tener un visión global de la eficiencia en la solución propuesta, se analiza el número de invocaciones acumuladas en ambos sistemas. Tal

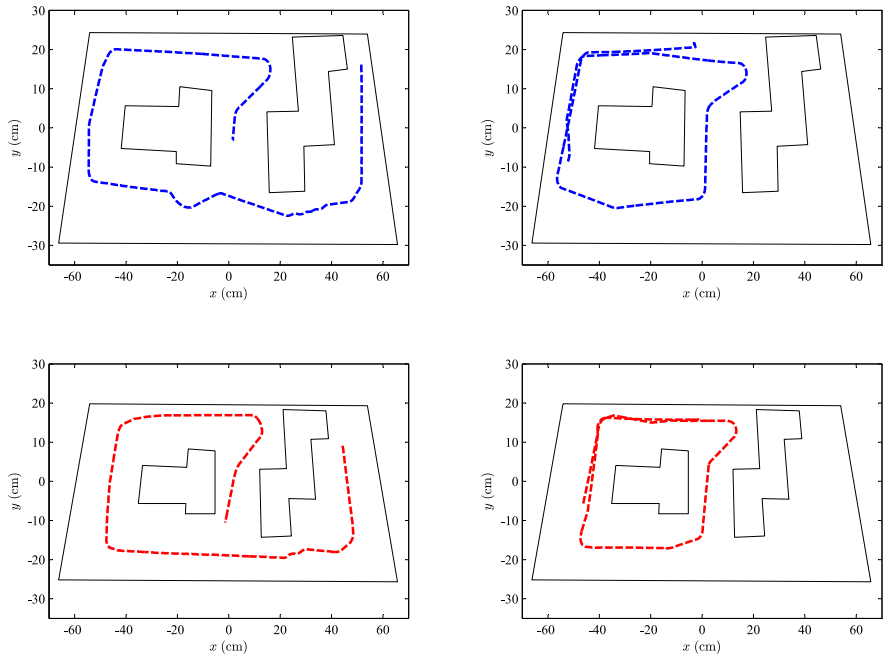

Figura 7.9: Análisis de trayectorias para el algoritmo EO. En azul el escenario 1 y en rojo el escenario 2.

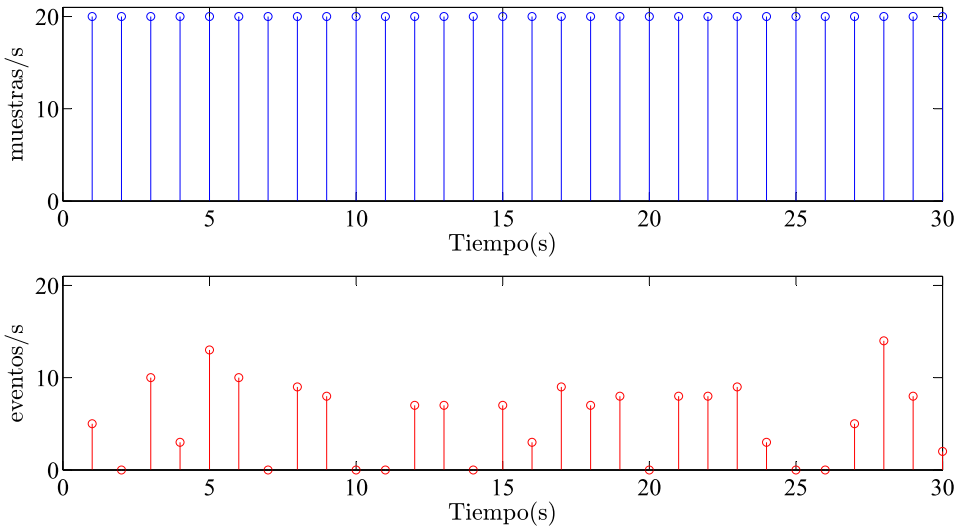

Figura 7.10: Actividad del controlador en el algoritmo EO. En azul el escenario 1 y en rojo el escenario 2.

como se muestra en la Figura 7.11, la actividad acumulada durante el tiempo que dura el experimento (30 s), en el sistema discreto son 600 muestras, en cambio el

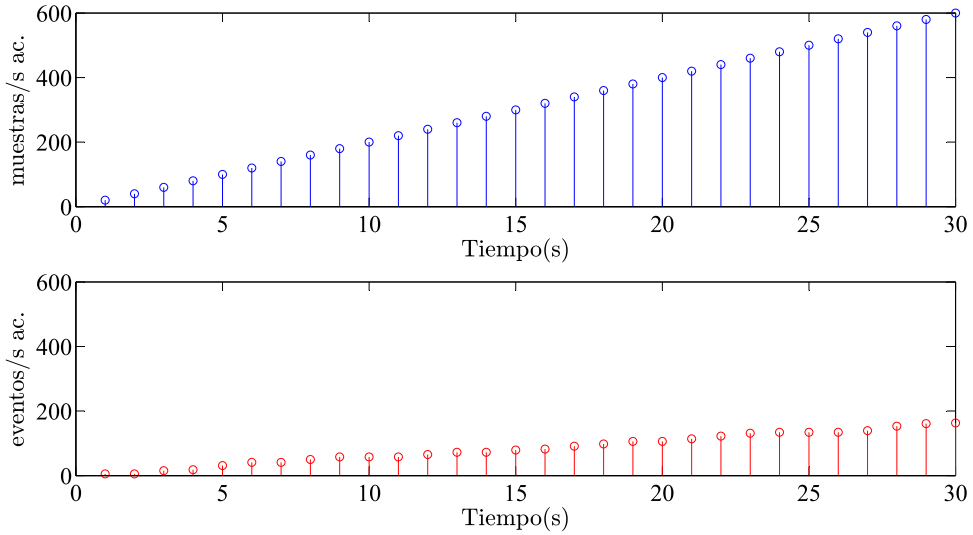

Figura 7.11: Actividad acumulada para el algoritmo EO. En azul el escenario 1 y en rojo el escenario 2.

sistema basado en eventos ha generado en total 163 eventos. Con estos valores se obtiene que el sistema discreto ha necesitado $3,7$ veces $(600/163)$ más muestras que el sistema propuesto basado en eventos.

7.3. Algoritmos para el Seguimiento de Paredes - SP

En la Sección 3.3.5 se propuso un algoritmo de seguimiento de paredes (SP) para la arquitectura de control basada en eventos. La condición de evento de dicho algoritmo (Ecuación 3.31), está parametrizada por la distancia a la que se quiere que el robot navegue respecto a la pared, w y el *event threshold*, $\bar{e}_{SP}$. Por otro lado, el controlador se configura también fijando dos parámetros. El primero de ellos es la velocidad estacionaria v, que se aplica a ambas ruedas del robot cuando este se encuentra exactamente a la distancia w de la pared. Y un segundo parámetro v_w, que fija la velocidad de aproximación/alejamiento de la pared cuando el robot está en estado transitorio. Como en los casos anteriores, para analizar el comportamiento de este algoritmo en un entorno real de robots móviles se han definido tres escenarios:

- Escenario 1. Sistema discreto con el mismo algoritmo en el controlador que

el sistema de eventos. El algoritmo del robot, en cada periodo de muestreo $\frac{1}{f_s}$, envía la información de los sensores de obstáculos al controlador.

- Escenario 2. Sistema basado en eventos propuesto en este trabajo con un *event threshold* $\bar{e}_{SP} = 0,30$ cm.
- Escenario 3. El mismo sistema basado en eventos que el Escenario 2 con un *event threshold* menos restrictivo $\bar{e}_{SP} = 0,45$ cm.

En la Tabla 7.3 se presenta el resumen de los escenarios planteados.

Tabla 7.3: Escenarios de análisis experimental del algoritmo SP.

Escenario	Arquitectura	f_s(Hz)	w(cm)	v(cm/s)	v_w(1/s)	$\bar{e}_{SP}$(cm)
1	Discreto	10	1,5	12	6	-
2	Eventos	-	1,5	12	6	0,30
3	Eventos	-	1,5	12	6	0,45

Tras ejecutar cada uno de los escenarios planteados, el comportamiento del robot es capturado y analizado con la herramienta *Tracker*. En la Figura 7.12 se presenta la secuencia de fotogramas para el Escenario 1 (discreto). En la Figura 7.13 el sistema de eventos con $\bar{e}_{SP} = 0,30$ cm del Escenario 2 y en la Figura 7.14 se muestra el Escenario 3 con $\bar{e}_{SP} = 0,45$ cm.

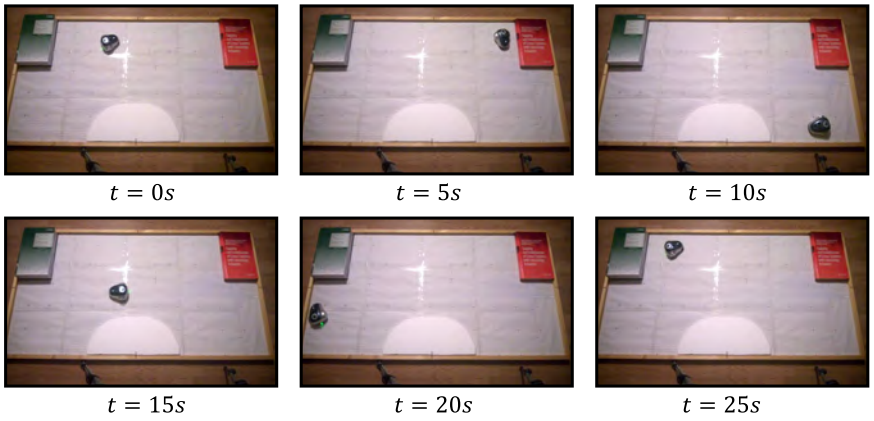

Figura 7.12: Respuesta del algoritmo SP en el escenario 1 (sistema discreto).

En la Figura 7.15 se presentan las trayectorias seguidas por el robot en cada uno de los escenarios. Tal como se observa en esta figura, las rutas que sigue el robot en cada uno de los escenarios analizados muestran que los tres controladores resuelven el problema de navegación de forma correcta, sin inestabilidades y con

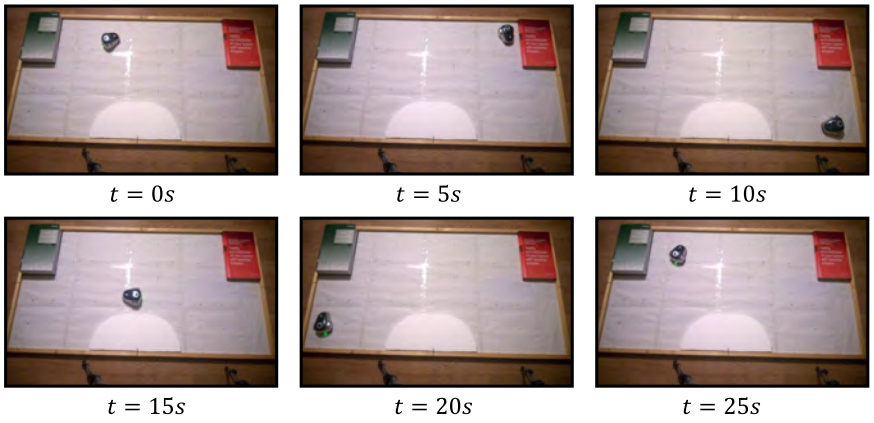

Figura 7.13: Respuesta del algoritmo SP en el escenario 2 (sistema basado en eventos con $\bar{e}_{SP} = 0,30$ cm).

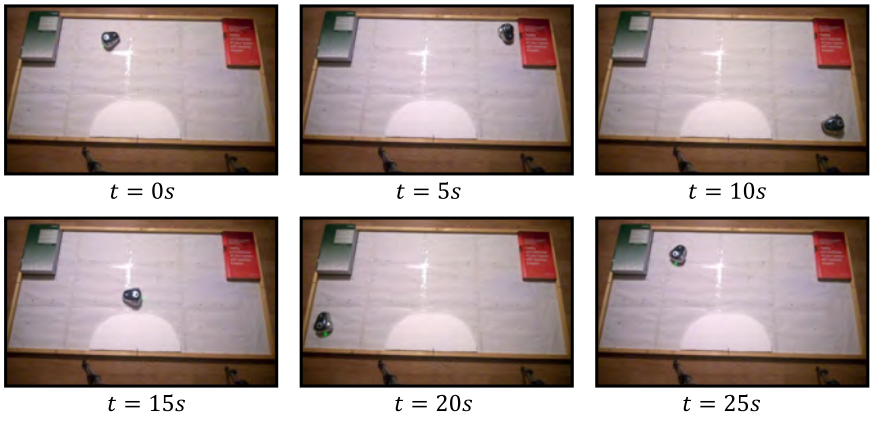

Figura 7.14: Respuesta del algoritmo SP en el escenario 3 (sistema basado eventos con $\bar{e}_{SP} = 0,45$ cm).

trayectorias que siguen las siluetas de las paredes de forma bastante aproximada. Es importante resaltar que prácticamente las tres soluciones tienen un resultado muy similar, incluso los dos controladores basados en eventos con *event threshold* diferentes.

Es interesante analizar la actividad del controlador en cada uno de los escenarios y las implicaciones que este comportamiento pueda tener en el consumo de recursos del sistema. En la Figura 7.16 se presenta la actividad instantánea de cada controlador (muestras/s o eventos/s) así como la actividad acumulada durante el tiempo que dura el experimento (30 s).

En la Figura 7.16 a) se muestra el número de invocaciones de la solución discreta, al trabajar este sistema con una frecuencia de muestreo $f_s = 10$ Hz genera 10 invocaciones por segundo de forma continua. Para el caso de los escenarios basados en eventos (Figuras 7.16 b) y c)), el número de eventos es siempre inferior a

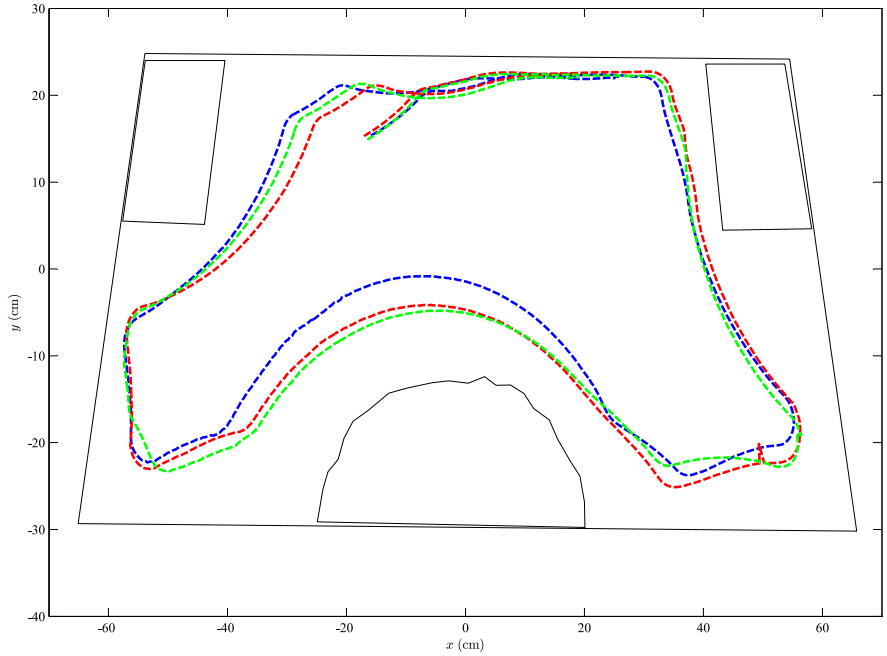

Figura 7.15: Análisis de las trayectorias para el algoritmo SP. En azul el escenario 1, en rojo el escenario 2 y en verde el escenario 3.

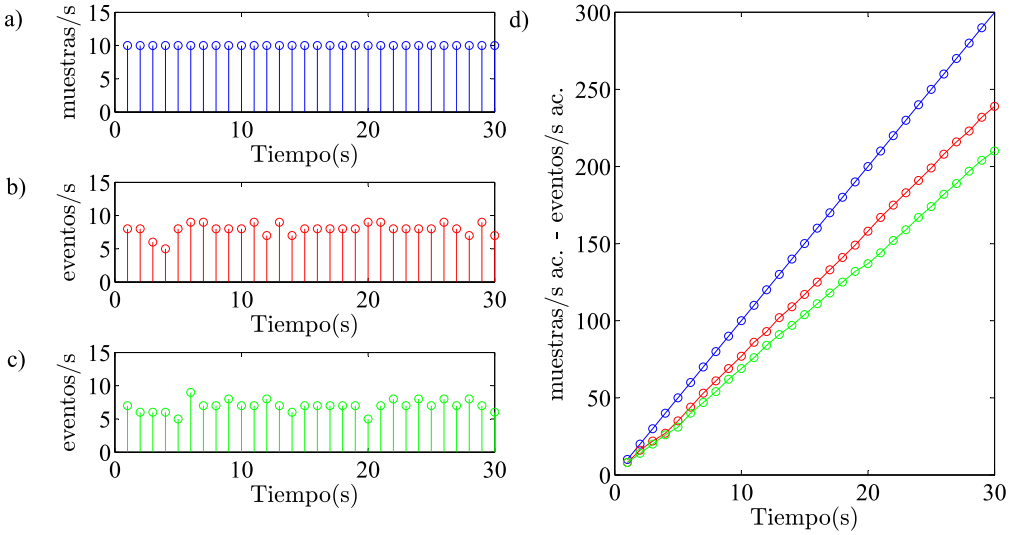

Figura 7.16: Actividad y actividad acumulada para los algoritmos SP. En azul el escenario 1, en rojo el escenario 2 y en verde el escenario 3.

10, incluso en los casos de mayor actividad del controlador. Como era de esperar, el Escenario 3 (Figura 7.16 c)), que tiene un *event threshold* menos restrictivo,

genera menos eventos que el Escenario 2 (Figura 7.16 b)).

Finalmente, para tener una visión global de la eficiencia de cada uno de los controladores, en la Figura 7.16 d) se presenta la actividad acumulada de cada uno de ellos. Como se muestra en dicha figura, el control discreto ha generado 300 invocaciones, el basado en eventos con umbral más restrictivo 239 eventos y el que posee el *event threshold* menos restrictivo 210. Con estos datos se obtiene que: el controlador discreto genera 1,25 veces más actividad que el controlador del Escenario 2 y 1,43 veces más que el controlador del Escenario 3. Dicho de otra forma, el controlador basado en eventos del Escenario 2 tiene un 20 % menos de actividad que su equivalente discreto y el del Escenario 3 una eficiencia aún mayor siendo en este caso un 30 % menor que su equivalente discreto.

7.4. Eficiencia de los Algoritmos Propuestos

En la Sección 3.4 se analizó la hipótesis de que los sistemas de control basados en eventos desarrollados en este trabajo presentan grandes eficiencias en el uso de recursos respecto a sus sistemas equivalentes en el dominio discreto. En ambos casos, los problemas de navegación se resuelven de forma satisfactoria y con unos resultados similares. Para el caso de los sistemas propuestos, éstos además tienen grandes eficiencias en el uso de los recursos de transmisión, computacionales e incluso en el consumo de energía.

Para analizar estas características de forma empírica, se ha planteado un escenario de análisis con un algoritmo de Evitación de Obstáculos - EO. En este escenario se utilizarán los mismos algoritmos que se analizaron en la Sección 7.2 de este capítulo y se harán trabajar dos robots de forma simultánea. Uno de los robots operará con control discreto para que nos sirva de marco de comparación y el otro con el algoritmo basado en eventos propuesto en este trabajo. En la Tabla 7.4 se presentan los parámetros más relevantes del escenario planteado.

Tabla 7.4: Escenario de análisis experimental sobre el consumo de recursos.

Robot	Arquitectura	f_s(Hz)	w(cm)	$\bar{e}_{EO}$(cm)
#1	Discreto	10	-	-
#2	Eventos	-	0,0	0,0

Para tener resultados representativos de este experimento, ambos algoritmos se hacen funcionar de forma continua durante un largo periodo de tiempo (20 mi-

nutos). Posteriormente se analiza y se compara el uso que se hace de los recursos en cada uno de los sistemas. En la Figura 7.17 se presentan las posiciones de los robots en diferentes instantes de tiempo del experimento. A lo largo del experi-

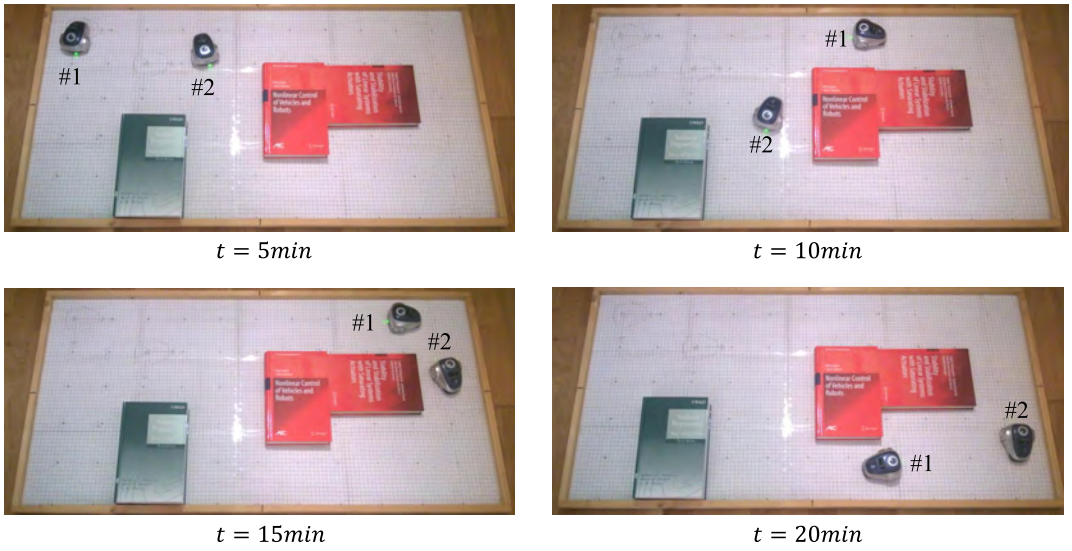

Figura 7.17: Respuesta del algoritmo EO para 20 min de análisis. Robot #1 con control discreto y robot #2 con control basado en eventos.

mento, ambos robots (el #1 con control discreto y el #2 con control basado en eventos) interactúan con el entorno y entre ellos, sin inestabilidades y resolviendo ambas arquitecturas el problema de navegación de forma satisfactoria.

En la Sección 3.4 de este documento se definió el concepto de ratio de eventos N_N, el ratio de actividad empírica de un recurso R (ancho de banda, carga computacional, etc.) como N_R, la eficiencia en el consumo de ese recurso $\eta_R(\,\%)$ y la eficiencia máxima teórica $\eta_N(\,\%)$ de un sistema de eventos respecto a su equivalente discreto. Estos serán los indicadores que se analizarán en este experimento de cara a evaluar la eficiencia en el consumo de recursos.

En la Figura 7.18 se presenta el resultado de la actividad en la arquitectura discreta utilizada como marco de referencia así como la basada en eventos desarrollada en este trabajo.

En este caso, se observa que el Ratio de Eventos N_N se encuentra en un valor promedio en torno al 40 %, por tanto, la eficiencia máxima que se podrá obtener del controlador basado en eventos será ese valor $\eta_N(\,\%) = (1 - N_N)100 \approx 60\,\%$. Como se ha comentado anteriormente, dependiendo de la magnitud sobre la que se quiera analizar la eficiencia, su valor será siempre igual o inferior a esta cota. En las siguientes secciones se analizarán en detalle la eficiencia para las

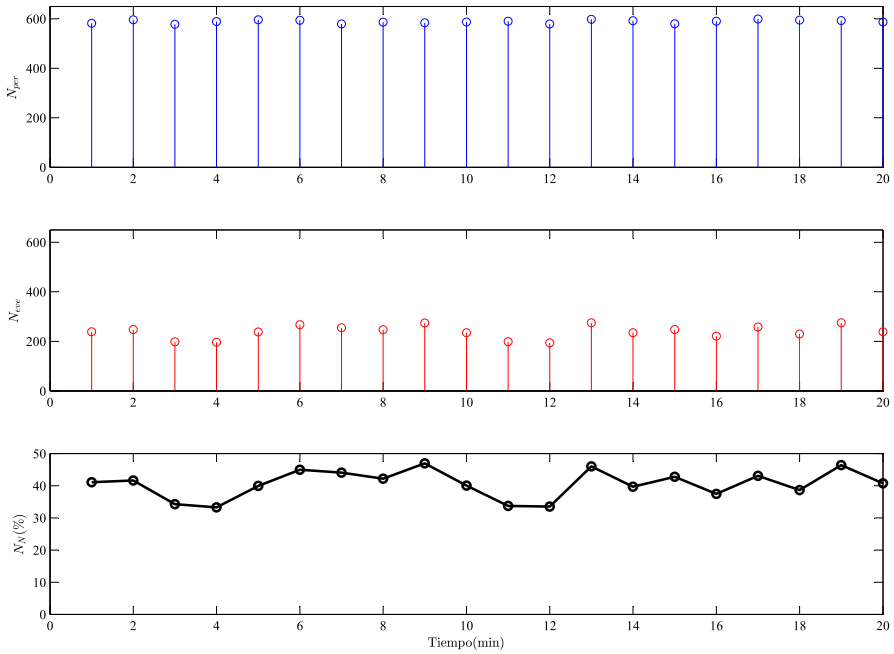

Figura 7.18: Actividad de los controladores y ratio de eventos. En azul control discreto y en rojo control basado en eventos.

magnitudes más características de estos sistemas como son el uso de recursos de comunicaciones, computacionales y de consumo de energía.

7.4.1. Eficiencia en Comunicaciones

Para medir la eficiencia en comunicaciones, se hará un análisis apoyado en la Figura 7.19. El modelo de comunicaciones mostrado en esta figura es aplicable

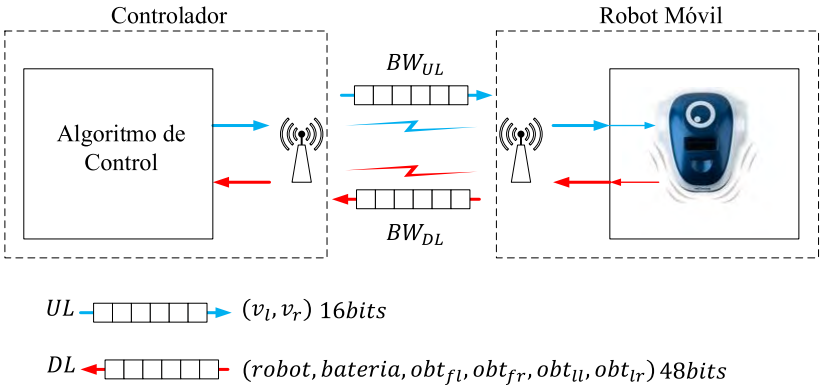

Figura 7.19: Modelo de comunicaciones para el experimento EO.

tanto al sistema discreto como al sistema basado en eventos. La diferencia principal entre ambos sistemas es que para el sistema discreto se invoca un proceso de comunicación *UpLink* (UL) y uno de *DownLink* (DL) en cada periodo de muestreo $\frac{1}{f_s}$. En cambio, para el sistema de eventos, esta comunicación se produce únicamente cuando se genera un evento k en el sistema. Por otro lado, la información que se intercambia es la misma en ambos esquemas. Para el caso del DL (desde el robot hacia el controlador) el sistema envía el indicador de robot, el nivel de batería y las medidas de los cuatro sensores de obstáculos, en total 48 bits. Para el caso del UL (controlador hacia el robot), se transmiten las velocidades a aplicar en cada una de las ruedas, esto supone un volumen de información de 16 bits.

Con estas consideraciones y midiendo el ancho de banda medio consumido en cada sentido de la comunicación, en la Figura 7.20 se muestran los resultados obtenidos. Tal como se muestra en esta figura, el ancho de banda en sentido DL

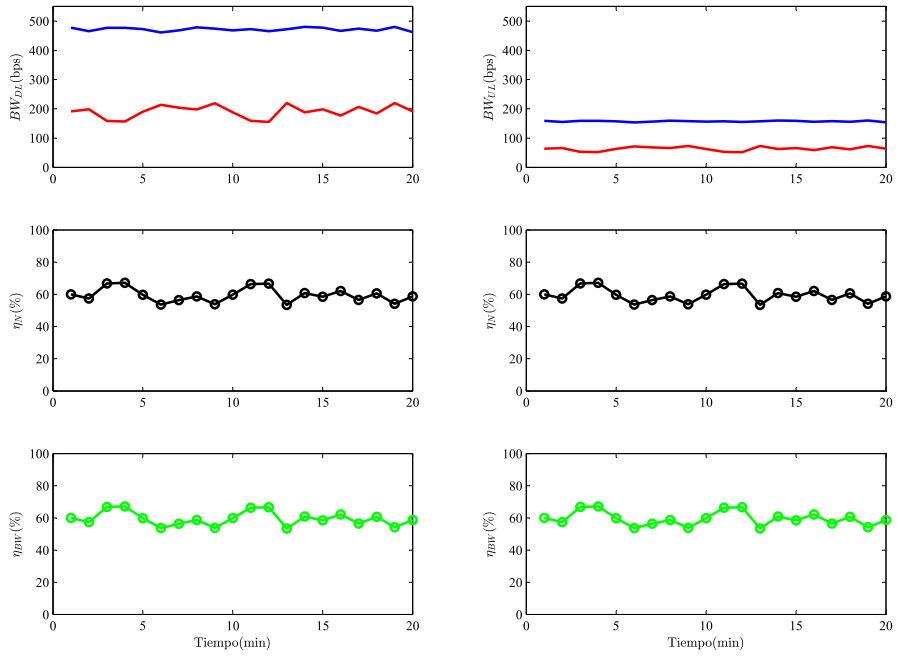

Figura 7.20: Consumo de ancho de banda. En azul control discreto y en rojo control basado en eventos.

es muy superior al UL como era de esperar, ya que en el sentido *DownLink* se transmite mucha más información que en el sentido opuesto. Por tanto, si se compara el sistema discreto con el basado en eventos, se observa un ahorro muy

notable de recursos de comunicación. Se observa también que $\eta_{BW}(\,\%) = \eta_N(\,\%)$ para ambos sentidos de la comunicación, lo que implica que para el caso de las comunicaciones la eficiencia es la máxima teórica, que en este caso ronda un valor medio del 60 %.

Como se ha visto, para los recursos de comunicaciones se puede conseguir una eficiencia igual a la máxima teórica. Para otras magnitudes como los recursos computacionales o el consumo de energía la situación no será tan favorable. Aún así, se consiguen importantes ahorros de recursos respecto a los sistemas discretos equivalentes como se describirá en las siguientes secciones.

7.4.2. Ahorro en Recursos Computacionales

Para poder analizar la eficiencia en el usos de recursos computacionales, hay que hacer un estudio exhaustivo de cómo están diseñados los algoritmos de control tanto del controlador como del robot. El algoritmo del controlador es muy similar tanto en el caso discreto como en el caso basado en eventos. En la Figura 7.21 se muestra un esquema compacto de ambos algoritmos. Tal como se muestra en la figura, el algoritmo que rige ambos sistemas es prácticamente el mismo, salvo la rama de la condición C_d que aplica solo al controlador discreto. Esta condición no se da en el caso de eventos ya que cuando el robot no detecta ningún obstáculo mantiene las velocidades que tiene en ese momento sin informar al controlador de este hecho.

Si se representa el algoritmo de los controladores mostrados en la Figura 7.21 como pseudocódigo según la Figura 7.22, se observa que el controlador discreto ejecuta 12 operaciones, en cambio el basado en eventos solo 11. Respecto al algoritmo que se ejecuta en los robots, también se presenta en la Figura 7.23 un esquema agrupado de ambas soluciones (discreta y eventos). Según el esquema de dicha figura, se observa que el algoritmo para el caso basado en eventos tiene dos secuencias de código en el robot:

- Una secuencia que trabaja de forma periódica en base a la frecuencia de muestreo nativa f_s del sistema. Esta parte del código comprueba la información de los sensores y chequea la condición de evento.

- El segundo bloque, es un código que actúa de forma eventual. Este algoritmo se ejecuta solo cuando se produce un evento en el sistema. Su misión principal es enviar la información de los sensores al controlador, recibir de

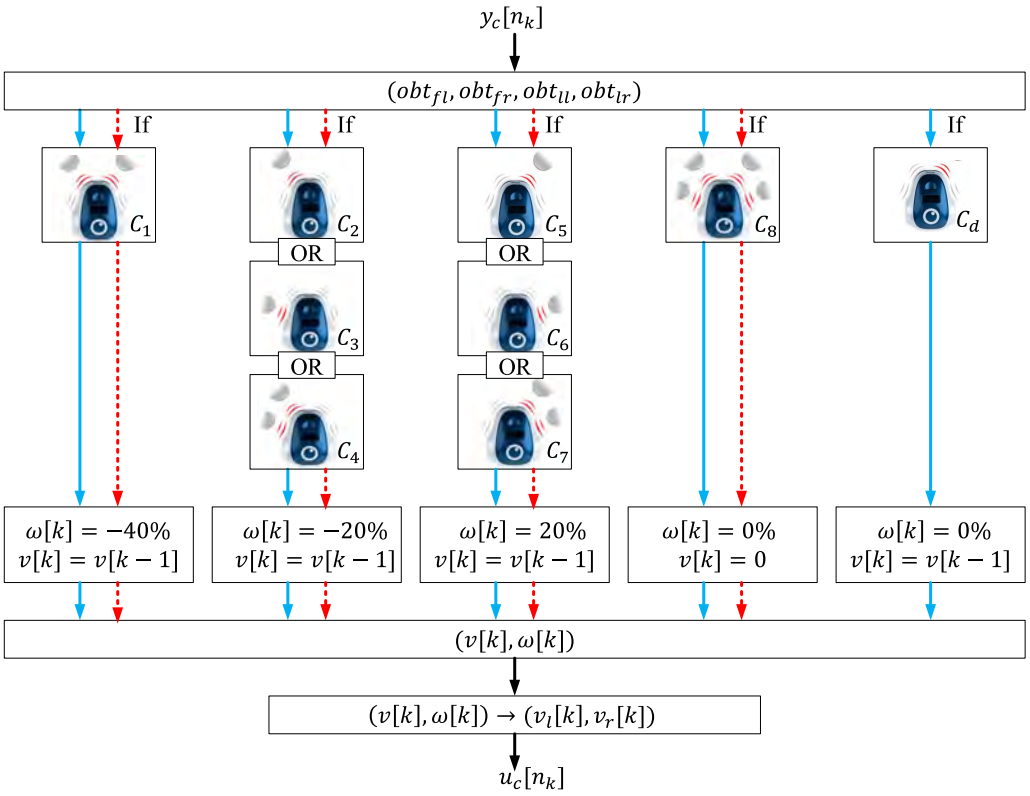

Figura 7.21: Algoritmo del controlador EO. En azul procesos del controlador discreto y en rojo procesos del controlador basado en eventos.

```
//Código Controlador
1-recibir (obtfl,obtfr,obtll,obtlr)
2-comprobar condición C1
3-comprobar condición C2 OR C3 OR C4
4-comprobar condición C5 OR C6 OR C7
5-comprobar condición C8
6-comprobar condición Cd
7-asignar V
8-asignar W
9-transformar (V,W)->vl
10-transformar (V,W)->vr
11-transmitir(vl,vr)
12-FIN
```

Figura 7.22: Pseudocódigo del controlador EO.

éste las velocidades de control y finalmente aplicarlas a los motores del robot.

Como se ha descrito anteriormente, el código de eventos del robot tiene una parte que se ejecuta de forma periódica y otra solo cuando hay eventos en el sistema. Por tanto, si el esquema de algoritmos de los robots mostrado en la

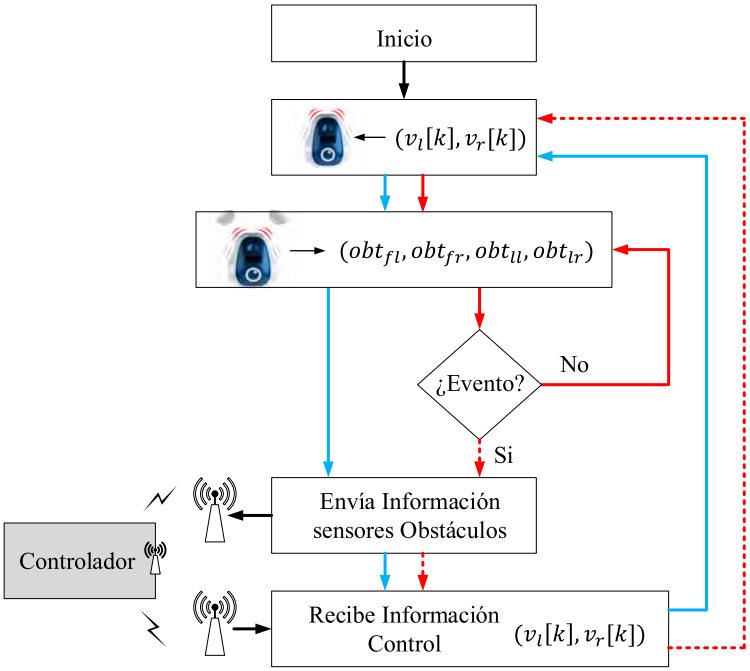

Figura 7.23: Algoritmo EO en el robot. En azul procesos del sistema discreto y en rojo procesos del sistema basado en eventos.

Figura 7.23 se representa en forma de pseudocódico las instrucciones para el robot con arquitectura discreta son las que se muestran en la Figura 7.24 y para el basado en eventos las de la Figura 7.25. Analizando los algoritmos presentados

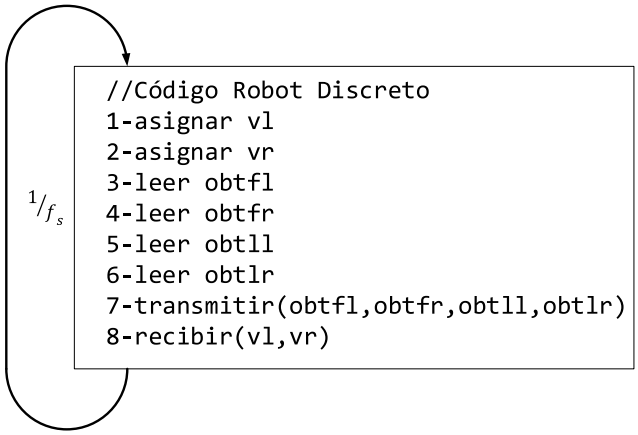

Figura 7.24: Pseudocódigo del robot discreto.

en estas figuras se deduce fácilmente que el robot trabajando en tiempo discreto ejecuta 8 operaciones cada periodo de muestreo $1/f_s$. En cambio, el robot del sistema basado en eventos ejecuta 9 operaciones de forma periódica y 5 cada vez que hay un evento en el sistema.

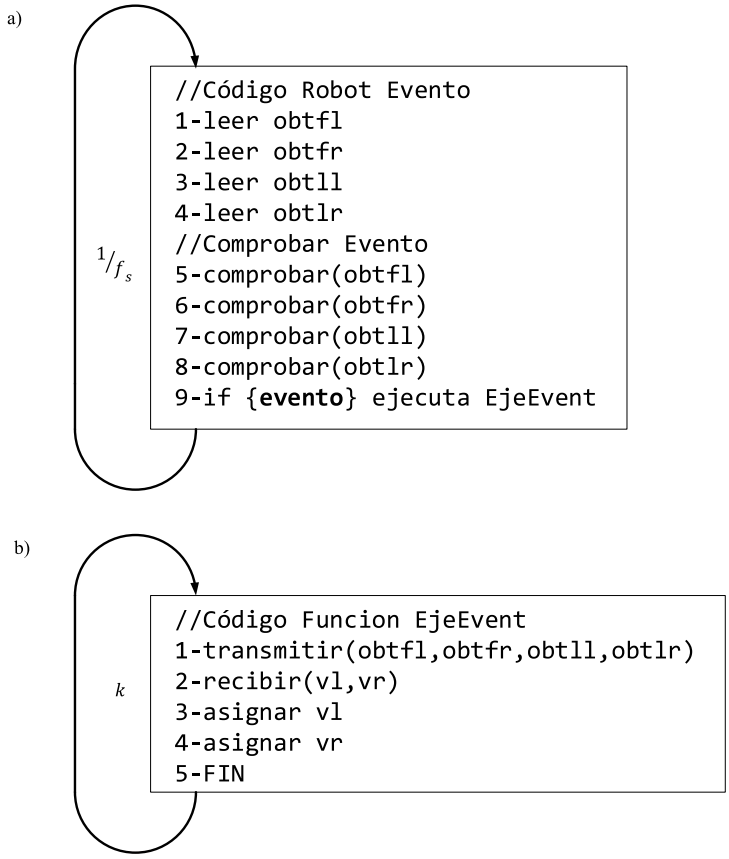

Figura 7.25: Pseudocódigo del robot basado en eventos. a) Código que se ejecuta de forma periódica y b) código que se ejecuta de forma eventual.

En la Figura 7.26 se presentan los resultados obtenidos en este experimento referente a la carga computacional. Tal como se muestra en la figura, la carga computacional del sistema discreto CC_{dis} es superior al sistema basado en eventos CC_{eve}. También se puede comprobar que el peso computacional del robot en el caso del sistema basado en eventos es superior al discreto (en la Figura 7.26 las barras azules representan la carga computacional del controlador y las barras rojas la carga computacional del robot). Esta mayor carga en el robot de eventos se debe a que ejecuta dos secuencias de código y por tanto mayor número de operaciones. En este caso, como era de esperar, la eficiencia en el consumo de recursos computacionales $\eta_{CC}(\%)$ es inferior a la máxima teórica $\eta_N(\%)$, como quedó demostrado en la Sección 3.4.2. Finalmente, y aunque no se haya alcanzado la eficiencia máxima, en este experimento se obtiene un ahorro de un 22 % en el uso de los recursos computacionales.

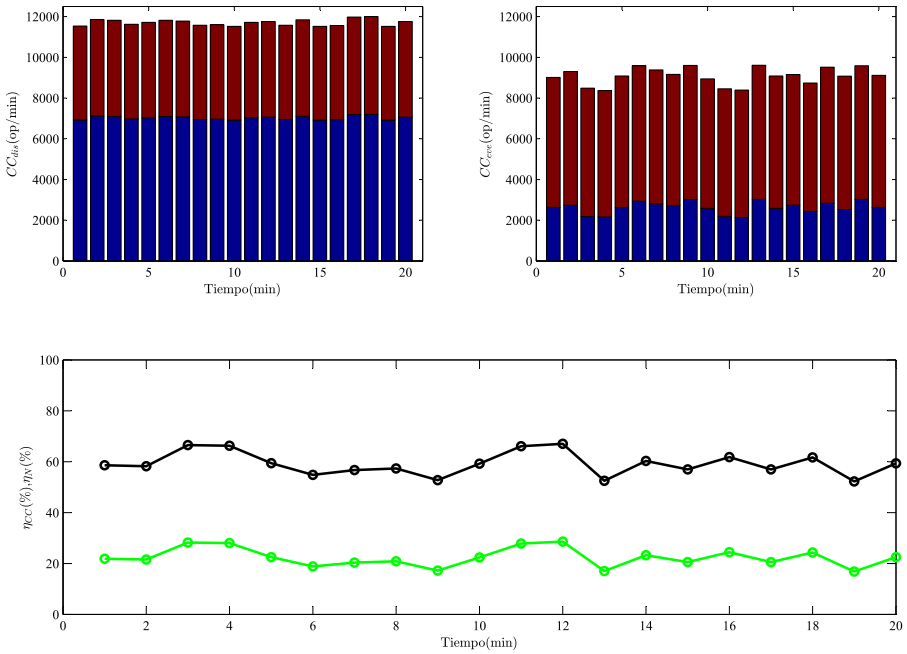

Figura 7.26: Resultados del consumo de recursos computacionales.

7.4.3. Ahorro Energético

En la Sección 3.4.3 se presentó un modelo para analizar el efecto que tienen la técnicas de control basadas en eventos en el consumo energético tanto del controlador como del robot. Para este experimento, solo se ha medido el consumo de energía en el robot. Se ha obviado el consumo del controlador ya que éste se implementa en un ordenador de propósito general e intentar extraer el consumo relativo solo a las funciones de controlador es muy complicado por la gran cantidad de *hardware* y *software* que este equipo posee.

En la Figura 7.27 se muestran los resultados de este experimento. Como se observa en los datos mostrados en la figura, el consumo de batería del sistema basado en eventos es menor que el consumo en el sistema discreto. Tal como se mencionó en la Sección 3.4.3, la eficiencia de este consumo depende fundamentalmente de dos factores:

1. El peso (parámetro β) que tiene en el consumo total del robot las funciones del módem para transmitir y recibir información del enlace inalámbrico.

2. La actividad del controlador basado en eventos. A medida que el número

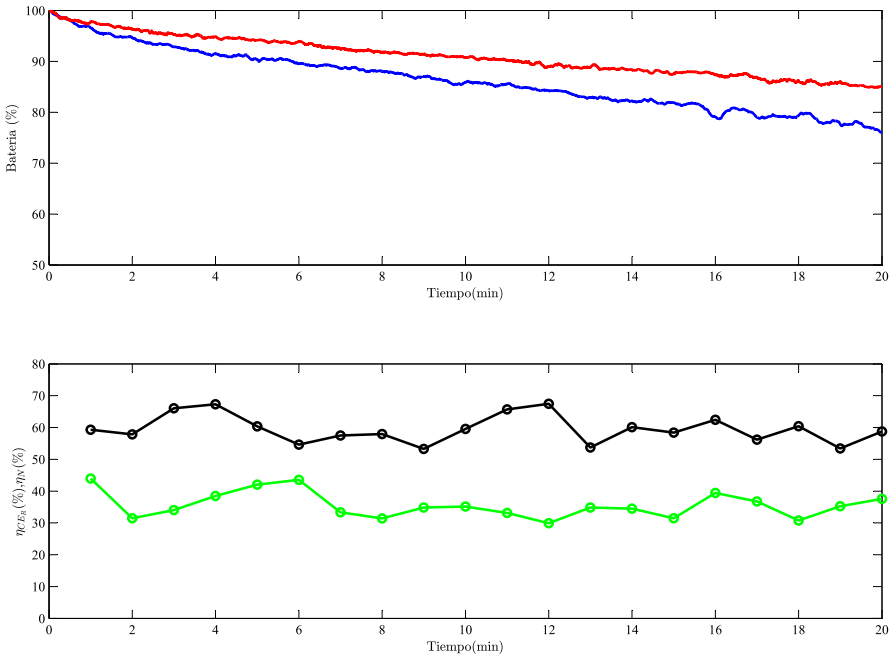

Figura 7.27: Resultados del consumo energético. En azul sistema discreto y en rojo sistema basado en eventos.

de eventos disminuye, el sistema debería de consumir menos energía.

En este caso, la eficiencia máxima teórica $\eta_N(\,\%)$ que es un parámetro general del sistema de eventos ronda el 60 % tal como se analizó previamente. Para el caso de la eficiencia en el consumo de energía $\eta_{CE_R}(\,\%)$, en este experimento se han alcanzado unos valores medios del 36 %. Por tanto, se puede concluir, que el sistema de control basado en eventos propuesto para robots móviles aporta grandes eficiencias en el consumo de energía de los dispositivos que trabajen bajo este esquema de control.

7.5. Modelado de Perturbaciones

En el Capítulo 5 (Sección 5.4) se ha definido una metodología para estimar las perturbaciones $P[n]$ que afectan a las medidas de los sensores. La filosofía que rige este método se basa en utilizar un patrón de medida conocido $S[n]$ y suponer que la perturbación $P[n]$ que afecta a estas medidas es un ruido aditivo. Posteriormente se debe estimar a partir de las señales que llegan al controlador

$SR[n]$ una realización del ruido $r[n]$ restando ambas señales $r[n] = SR[n] - S[n]$. Finalmente, de esta realización $r[n]$ se estimará qué distribución de probabilidad se ajusta mejor a estos datos, obteniendo así el modelo de ruido del sensor.

Por otro lado, si se analizan las perturbaciones que afectan a los sensores de la plataforma mOway utilizada en la experimentación, se observa lo siguiente:

- Los sensores inerciales (el acelerómetro) tienen un ruido bastante elevado que afecta de forma considerable a las medidas de estos sensores.
- Los sensores de posición tienen un ruido que puede considerarse despreciable en relación a los errores relativos a la estimación de posición basada en técnicas odométricas. Por tanto, en los experimentos que utilicen los sensores de posición se considerará que el ruido de este sensor es nulo.
- Por último, los sensores de distancia basados en infrarrojos, también pueden considerarse que tienen un ruido nulo dado que la perturbación que les afecta es muy inferior a su precisión. Dado que la precisión es baja, tener una estimación de este ruido aporta poco valor a estas medidas de distancia.

Por tanto, considerando las premisas anteriores, en las siguientes secciones se modelará primero el ruido del acelerómetro y posteriormente el ruido de los sensores de distancia. De cara a analizar el comportamiento de las estructuras de ajuste de umbrales de forma dinámica propuestas en este trabajo, se utilizará un método alternativo para modelar el ruido de los sensores de distancia. Dado que estos sensores tienen un ruido poco relevante en relación a los errores de odomentría, para probar las estructuras de umbrales dinámicos se contaminarán estas medidas con ruido procedente de otros sensores como es el micrófono que contiene la plataforma de robots mOway.

7.5.1. Modelo de Ruido del Acelerómetro

El robot mOway con el que se trabaja en la experimentación contiene el acelerómetro MMA7455L de Freescale Semicinductors de tres ejes. Dicho sensor tiene una sensibilidad que va desde $-2g$ hasta $+2g$ (donde $g = 9,81$ m/s^2) para cada uno de los ejes y cuya codificación en los registros internos del robot tiene un rango que van desde 0 (para una aceleración de $-2g$) hasta 255 (para $+2g$). Por otro lado, en la Figura 7.28 se muestra el criterio para representar la aceleración en cada uno de los ejes del sensor.

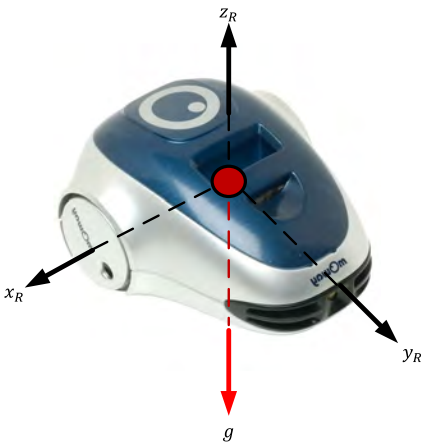

Figura 7.28: Referencia para las medidas del acelerómetro.

Este sensor es sensible a cualquier aceleración (lineal, centrípeta, gravitatoria, etc.) a la que es sometida el robot. Tras analizar las aceleraciones del robot en varios movimientos de cambio de velocidad y trayectoria, se concluye que la componente que predomina es la gravitatoria. Por tanto, en este experimento se supone que, salvo la aceleración gravitatoria, el resto de componentes de las aceleraciones en el sensor del robot no tienen influencia.

El acelerómetro es capaz de medir la aceleración en cada uno de los ejes respecto al sistema de referencia (x_R, y_R, z_R) situado en el c.g. del robot. Las medidas de dichas aceleraciones se codifican en el sistema por valores numéricos que están comprendidos en un rango que va desde 0 hasta 255 y que representan una aceleración entre $-2g$ hasta $+2g$ como se explicó previamente. Se define cada componente de la aceleración que mide el sensor como a_{xs}, a_{ys} y a_{zs}, cuyos valores pueden estar entre 0 y 255. Para expresar estas aceleraciones en $g's$ se utilizarán las Ecuaciones 7.1 - 7.3.

$$a_{xg} = -(a_{xs} - 127{,}5)\frac{4}{256} \tag{7.1}$$

$$a_{yg} = -(a_{ys} - 127{,}5)\frac{4}{256} \tag{7.2}$$

$$a_{zg} = -(a_{zs} - 127{,}5)\frac{4}{256} \tag{7.3}$$

donde a_{xg}, a_{yg} y a_{zg} son las aceleraciones que mide el sensor expresadas en $g's$.

Las medidas de aceleración a_{xg}, a_{yg} y a_{zg} así obtenidas, son medidas sin calibrar, por tanto, antes de que puedan utilizarse en el sistema debe realizarse un

proceso de calibrado. Por otro lado, en la metodología de estimación propuesta, se debe utilizar un patrón conocido $S[n]$ para estimar el ruido. Se puede entonces utilizar un mismo patrón para realizar ambas operaciones al mismo tiempo, es decir, calibrado y estimación de ruido. El procedimiento seguido es el siguiente:

1. Robot estático sobre un plano nivelado, donde las aceleraciones que mediría el sensor deberían ser $a_{xg} = 0$, $a_{yg} = 0$ y $a_{zg} = -1$.

2. Robot en movimiento aleatorio sorteando obstáculos sobre un plano nivelado donde las aceleraciones medidas también deberían ser $a_{xg} = 0$, $a_{yg} = 0$ y $a_{zg} = -1$. Se considera también este segundo caso para contemplar el ruido eléctrico y mecánico que produce el movimiento del robot.

Con las medidas anteriores se obtienen los parámetros de calibrado según las Ecuaciones 7.4 - 7.6.

$$a_{xbias} = \sum_{i=1}^{n} a_{xs}[i] \tag{7.4}$$

$$a_{ybias} = \sum_{i=1}^{n} a_{ys}[i] \tag{7.5}$$

$$a_{zbias} = \sum_{i=1}^{n} a_{zs}[i] \tag{7.6}$$

En ambos casos (robot estático y en movimiento), los parámetros de calibración obtenidos son casi similares, por lo que se concluye que el movimiento del robot no influye en la calibración de este sensor. Los valores obtenidos han sido $a_{xbias} = 120,6477$, $a_{ybias} = 105,6574$ y $a_{zbias} = 197,2968$. Por tanto, considerando estos parámetros de calibración, las medidas del sensor calibradas se pueden expresar mediante las Ecuaciones 7.7 - 7.9.

$$a_x = -(a_{xg} - a_{xbias})\frac{4}{256} \tag{7.7}$$

$$a_y = -(a_{yg} - a_{ybias})\frac{4}{256} \tag{7.8}$$

$$a_z = -(a_{zg} - a_{zbias} + 64)\frac{4}{256} \tag{7.9}$$

Con este procedimiento, las medidas obtenidas ya calibradas se presentan en la Figura 7.29.

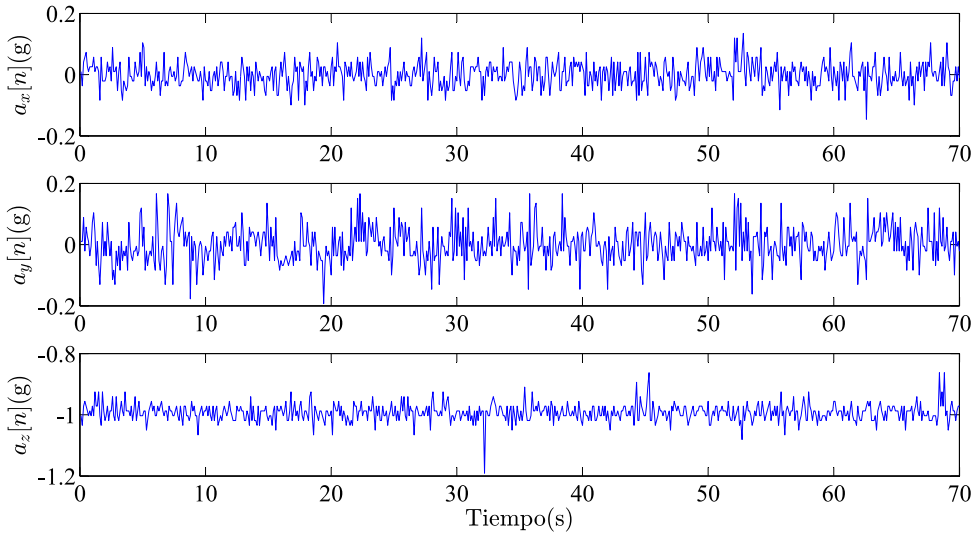

Figura 7.29: Medidas calibradas del acelerómetro.

Por otro lado, el patrón utilizado en este caso para calibrar el sensor $(0, 0, -1)$ es el mismo que el que se utilizará para obtener la realización del ruido. Con este criterio, una realización del ruido de medidas para cada una de las componentes del sensor $n_{ax}[n]$, $n_{ay}[n]$ y $n_{az}[n]$ durante el periodo de medida de 70s, puede obtenerse según las Ecuaciones 7.10 - 7.12.

$$n_{ax}[n] = a_x[n] \tag{7.10}$$

$$n_{ay}[n] = a_x[n] \tag{7.11}$$

$$n_{az}[n] = a_x[n] + 1 \tag{7.12}$$

Una vez obtenida una realización del ruido para cada una de las componentes del sensor, se aplica la metodología desarrollada en la Sección 5.4. Dentro del conjunto de distribuciones a estudiar, existe un grupo que no se pueden aplicar al conjunto de datos bajo análisis (p.e aquellas distribuciones que solo aceptan valores positivos) aunque el sistema desarrollado las soporte. Por tanto, y tras una análisis exhaustivo, el conjunto de distribuciones a analizar se recogen en la Tabla 7.5. Con el objetivo de poder representar los resultados de una manera

Tabla 7.5: Distribuciones analizadas para obtener el modelo de perturbación.

Grupo	Distribución (i)	Color Representativo	parámetro 1	parámetro 2
I	Uniforme	azul	a	b
I	Normal	rojo	μ	σ
I	Rayleigh	verde	b	-
II	Valor Extremo	azul	μ	σ
II	Geométrica	rojo	p	-
II	Logística	verde	μ	σ

ordenada, a cada distribución se le ha asignado un grupo y un color característico. En esa misma tabla también se muestran los parámetros que definen dichas distribuciones. En el Apéndices D se muestran los detalles de cada una de las distribuciones analizadas.

Siguiendo la metodología, una vez obtenidos el ajuste de los datos bajo análisis mediante el método de máxima verosimilitud, se presentan las funciones de distribución acumuladas $F_i(x)$ de cada una de ellas y se compara con la función de la realización de ruido bajo análisis $F_r(x)$. En las Figuras 7.30 - 7.32 se muestran los resultados obtenidos.

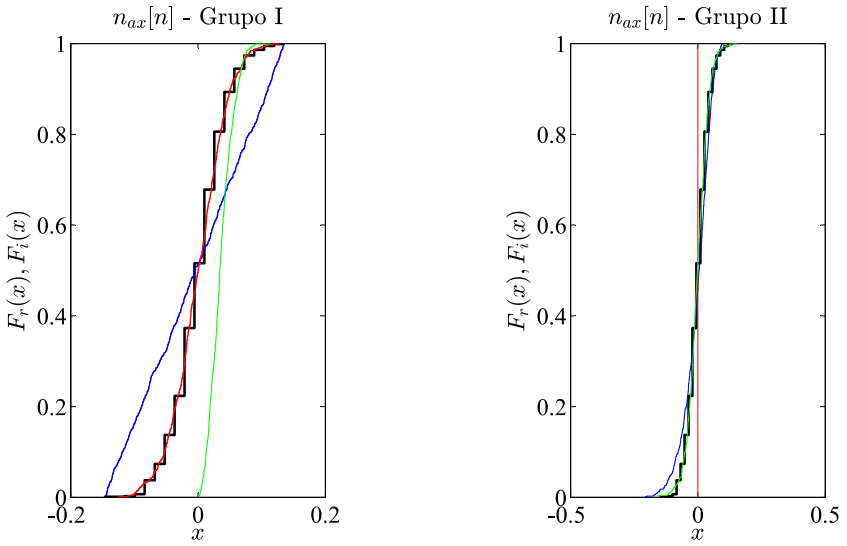

Figura 7.30: Funciones de distribución acumulada para el ruido $n_{ax}[n]$.

Analizando los datos presentados en la Figuras 7.30 - 7.32 se observa que que el ruido de cada una de las componentes $n_{ax}[n]$, $n_{ay}[n]$ y $n_{az}[n]$ tienen tres distribuciones posibles que se ajustan a los datos de la realización de ruido bajo

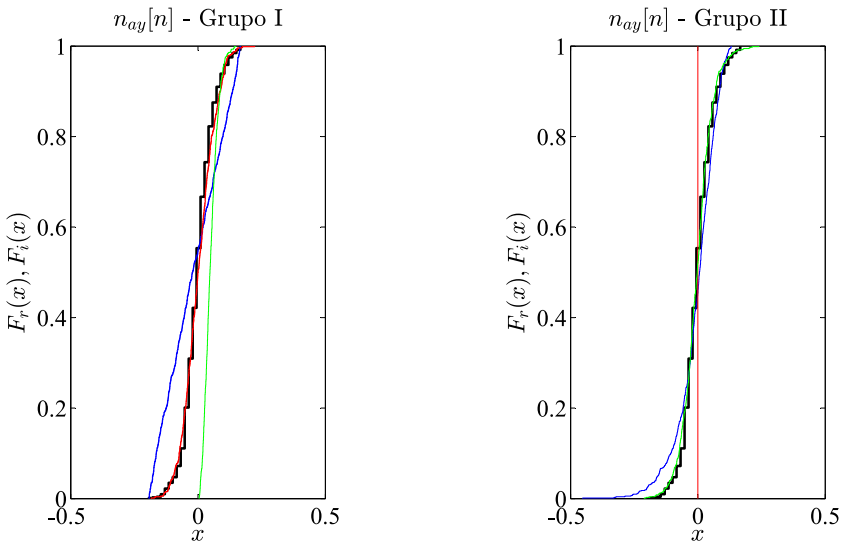

Figura 7.31: Funciones de distribución acumulada para el ruido $n_{ay}[n]$.

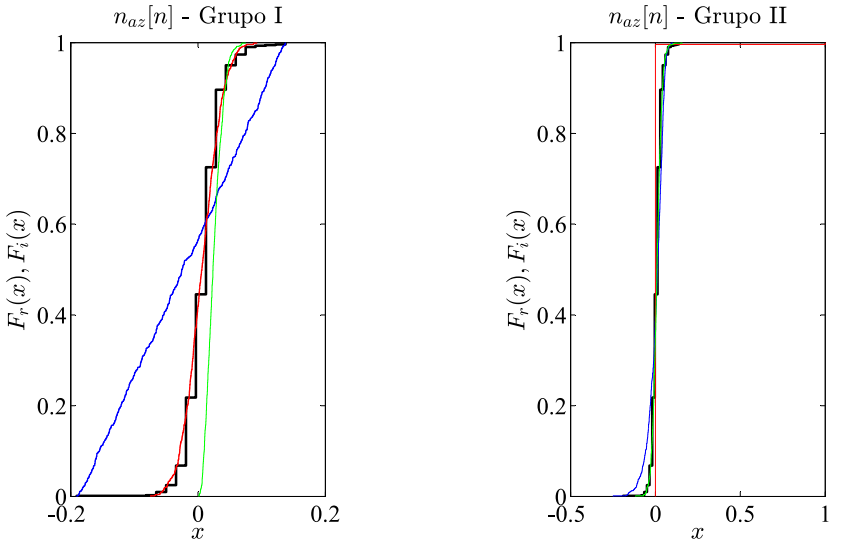

Figura 7.32: Funciones de distribución acumulada para el ruido $n_{az}[n]$.

estudio. Por tanto, puede concluirse que el ruido de este sensor puede modelarse con cualquiera de las distribuciones mostradas en la Tabla 7.6.

7.5.2. Modelo de Perturbación para los Sensores de Posición

Tal como se ha comentado anteriormente, las perturbaciones en los sensores de posición de la plataforma de experimentación utilizada en este trabajo tiene poco

Tabla 7.6: Parámetros del modelo de perturbación del acelerómetro.

Perturbación	Distribución (i)	μ(g)	σ(g)
$n_{ax}[n]$	Normal	$0,0000$	$0,0409$
$n_{ax}[n]$	Valor Extremo	$0,0206$	$0,0418$
$n_{ax}[n]$	Logística	$0,0000$	$0,0232$
$n_{ay}[n]$	Normal	$0,0000$	$0,0578$
$n_{ay}[n]$	Valor Extremo	$0,0296$	$0,0613$
$n_{ay}[n]$	Logística	$-0,0017$	$0,0321$
$n_{az}[n]$	Normal	$0,0078$	$0,0277$
$n_{az}[n]$	Valor Extremo	$0,0221$	$0,0349$
$n_{az}[n]$	Logística	$0,0073$	$0,0144$

peso respecto a los errores de odometría. Por tanto, y de cara a poder simular un entorno con altos niveles de ruido en los sensores de posición, se añadirá de forma artificial ruido procedente del micrófono del robot a dichos sensores tal como se ilustra en la Figura 7.33.

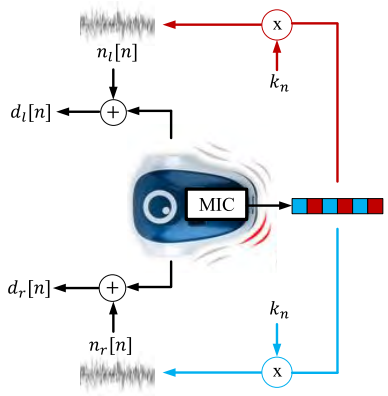

Figura 7.33: Generador de ruido para los sensores de posición.

Dado que el micrófono recoge tanto el ruido de los motores del robot, el sonido ambiente y el propio ruido inherente de este sensor, esta perturbación puede ser excesiva para que el sistema tenga una respuesta estable. Para controlar esta situación, esta perturbación se modula con una constante k_n y que permite tener un nivel de ruido que no sea excesivo. Para el experimento que se detallará a continuación este parámetro se ha fijado a $k_n = 1/80$.

En este experimento se trata de analizar como afecta esta perturbación al parámetro $\alpha[n]$ en un algoritmo DD. Una vez se tenga modelada esta perturba-

ción, en la siguiente sección se analizará otro experimento de cómo puede utilizarse este modelo para fijar el *event threshold* $\bar{e}_\alpha$ del sistema. Por otro lado, si analiza el modelo cinemático presentado en la Sección 2.2.1, se observa que la perturbación en los sensores de posición $n_l[n]$ y $n_r[n]$ se traslada al parámetro $\alpha[n]$ mediante un factor de escala $\frac{n_r[n]-n_l[n]}{L}$, donde L es la distancia entre las ruedas del robot. Con estas premisas, en la Figura 7.34 se presenta una realización de 40 s de la perturbación en los sensores de posición así como la correspondiente perturbación en el parámetro $\alpha[n]$ que en este caso se denota por $n_\alpha[n]$.

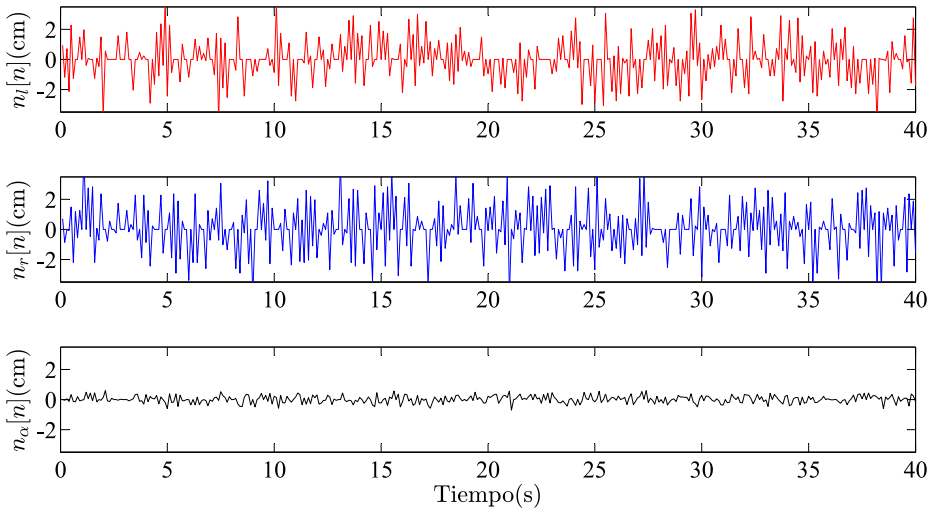

Figura 7.34: Perturbaciones en los sensores de posición y en el error de orientación.

Como siguiente paso, se aplica nuevamente la metodología presentada en la Sección 5.4 con el conjunto de distribuciones de la Tabla 7.5, obteniéndose los resultados de la Figura 7.35. Tal como se observa en la figura, las dos distribuciones que mejor modelan esta perturbación son la Normal y la Logística. En la Figura 7.36 se presentan solo las funciones de distribución acumulada de estas dos distribuciones con la de los datos bajo análisis para verificar que realmente se ajustan de forma muy precisa a esta perturbación. Por tanto, tras la aplicación de este método, ya se dispone del modelo de ruido que afecta al parámetro $\alpha[n]$ y que pueden describirse con cualquiera de las dos distribuciones comentadas. En la Tabla 7.7 se muestran los parámetros de dichos modelos.

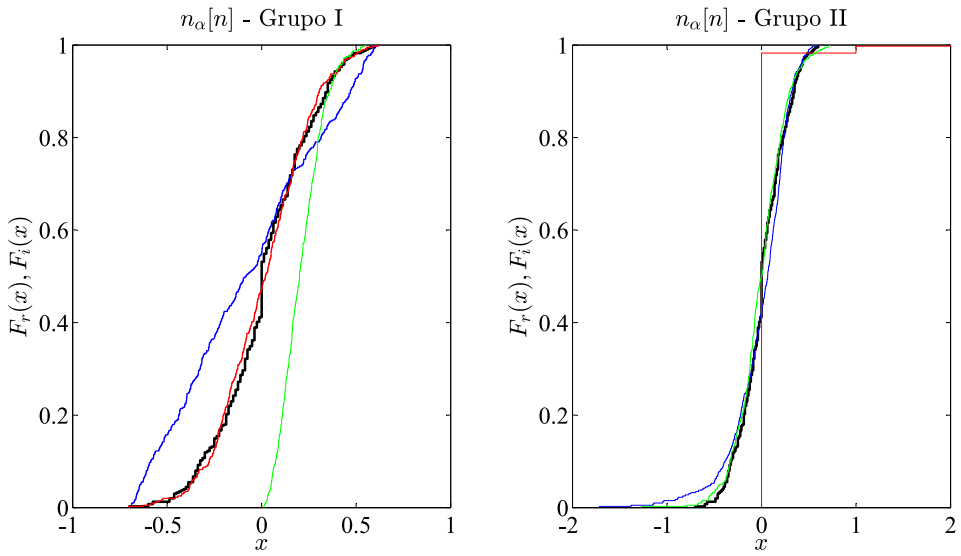

Figura 7.35: Funciones de distribución acumulada para el ruido en el error de orientación $n_\alpha[n]$.

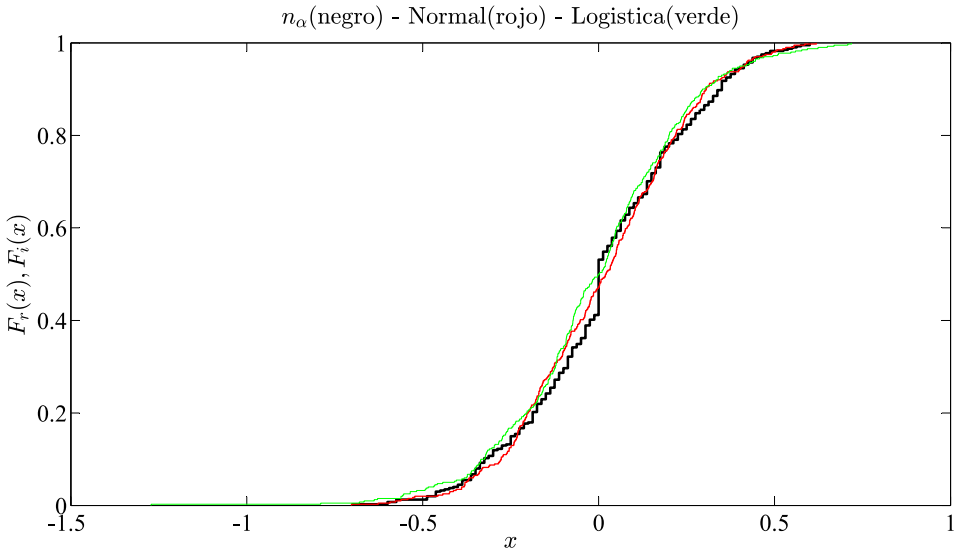

Figura 7.36: Funciones de distribución acumulada para las distribuciones Normal y Logística.

Tabla 7.7: Parámetros del modelo de perturbación del error de orientación $\alpha[n]$.

Perturbación	Distribución (i)	μ(rad)	σ(rad)
$n_\alpha[n]$	Normal	$0,0167$	$0,2414$
$n_\alpha[n]$	Logística	$0,0179$	$0,1383$

7.6. Mecanismos de Ajuste Dinámico de Umbrales

En el Capítulo 4, Sección 4.6.2, se propuso una arquitectura que permite a partir de la señal de salida del sistema estimar el ruido de medida y fijar un umbral dinámico en el sistema. Se demostró con escenarios de simulación que esta solución tenía una mejor respuesta en cuanto al número de eventos generados que su equivalente con umbrales estáticos. Posteriormente, en el Capítulo 5, Sección 5.2, se desarrolló otra alternativa para fijar umbrales en el sistema conociendo la distribución estadística que modela el ruido que afecta a los sensores.

En este experimento se plantea analizar el algoritmo DD de control basado en eventos con ruido en los sensores de posición. La perturbación que se añade a los sensores de posición es la misma que se ha analizado en la Sección 7.5.2 de este capítulo. Para realizar este análisis, se plantean tres escenarios según se muestra en la Tabla 7.8 cuya señal de consigna es $x_g = 50$ cm e $y_g = 40$ cm en cada uno de ellos.

Tabla 7.8: Escenarios de análisis experimental para el algoritmo DD en un entorno con ruido de medida.

Escenario	Arquitectura	K_ρ(1/cm)	K_α(1/s)	l	$\bar{e}_\alpha$ (rad)	e_ρ (cm)
1	U. Estático	0,2	3,8	-	0,2	8
2	C. Umbral	0,2	3,8	5	0,2+$\bar{e}_{\alpha d}[n]$	8
3	E. Perturbación	0,2	3,8	-	0,2+$\bar{e}_{\alpha d}$	8

En los tres escenarios planteados se utilizan los mismos parámetros del controlador K_ρ y K_α. En estos escenarios y debido al ruido, la precisión para alcanzar el destino disminuye de forma considerable. Por tal motivo, se pone una condición de parada e_ρ menos restrictiva para asegurar la convergencia del algoritmo. En el Escenario 1 el sistema trabaja con un umbral estático. El calculador de umbral planteado en la Sección 4.6.2 se aplica en el Escenario 2. Finalmente, en el Escenario 3 se obtiene la parte dinámica del umbral tal como se definió en Sección 5.2 mediante la Ecuación 7.13. En este último caso se ha considerado una realización $r[n]$ del modelo de perturbación que se obtuvo en la Sección 7.5.2.

$$\bar{e}_{\alpha d} = Per(abs(r[n]), p) \tag{7.13}$$

Si se considera una realización de este modelo de ruido con cada una de las distribuciones (Normal y Logística) y se aplica la Ecuación 7.13 con un valor de $p = 70\,\%$ (percentil 70) se obtienen las componentes dinámicas para el umbral del Escenario 3. Los resultados obtenidos se muestran en la Tabla 7.9. Tal como

Tabla 7.9: Realizaciones del modelo de perturbación para el error de orientación $\alpha[n]$.

Distribución	μ(rad)	σ(rad)	$length(r[n])$	$Per(abs(r[n]), 70\,\%)$
Normal	0,0167	0,2414	6.000	0,25
Logística	0,0179	0,1383	6.000	0,24

se observa en la tabla, la componente dinámica del umbral obtenida para cada uno de los modelos es prácticamente la misma $(0,25$ vs. $0,24)$ por lo que se puede utilizar cualquiera de las dos indistintamente.

Una vez fijados los parámetros para cada uno de los escenarios, se ejecutan éstos sobre la plataforma de experimentación obteniéndose los resultados mostrados en la Figura 7.37. Tal como se observa en la figura, en los tres escenarios planteados se resuelve el problema de navegación de forma satisfactoria. Como es lógico y debido a las perturbaciones, cada uno de los experimentos termina en un punto diferente pero siempre dentro del margen marcado por la condición de parada e_ρ (circulo rojo).

En la Figura 7.38 se muestran los parámetros de posicionamiento y orientación para cada uno de los escenarios, como se ilustra en esta figura, los tres escenarios tienen una respuesta muy parecida. Aunque en la fase transitoria hay grandes variaciones sobre todo en la orientación, cuando los robots llegan al destino tanto las posiciones como las orientaciones son muy similares.

En cada escenario analizado se ha aplicado un umbral diferente y por tanto la actividad de cada controlador dependerá en gran medida de este parámetro. Cuanto más ajustado esté el umbral a la envolvente de ruido que afecta a las medidas de los sensores, mejor se comportará el controlador en cuanto a la generación de eventos. Para el experimento bajo análisis, en la Figura 7.39 se presentan los valores de los umbrales, la evolución del error de orientación $\alpha[n]$ y el número de eventos generados. Observando estos resultados se comprueba que la solución que mejor se comporta desde el punto de vista de la generación de eventos es la que fija el umbral a través de la estimación de un modelo de la perturbación. Le sigue la arquitectura que utiliza el calculador de umbral y finalmente la de peor comportamiento es la que utiliza un umbral estático.

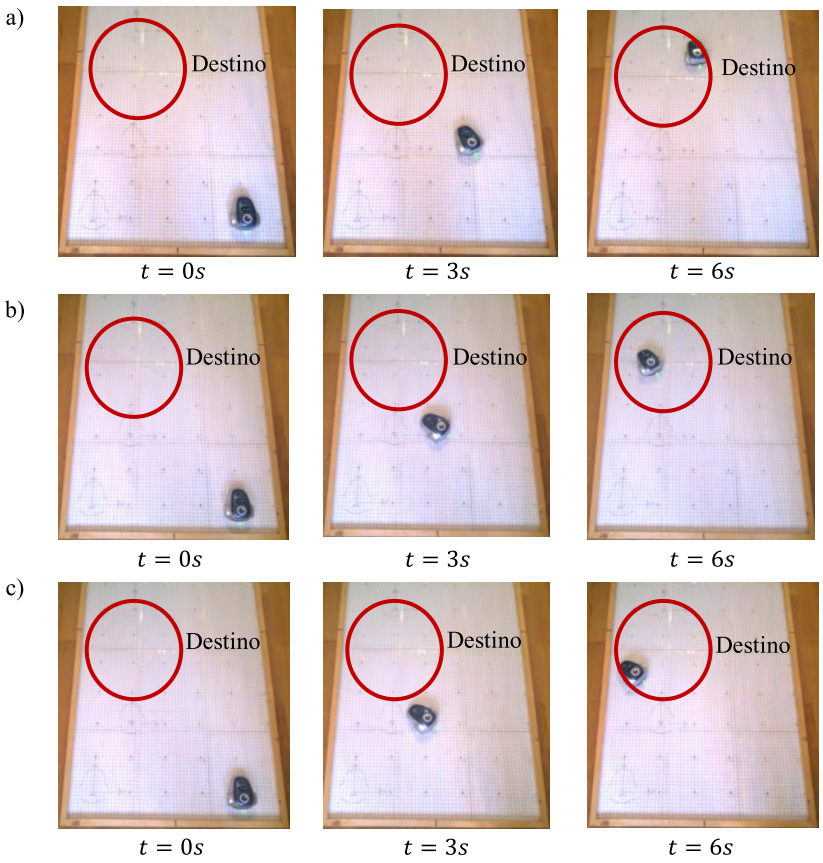

Figura 7.37: Respuesta del algoritmo DD con ruido en los sensores de posición. a) Escenario 1, b) escenario 2 y c) escenario 3.

Para tener una visión más clara de este efecto, en la Figura 7.40 se muestra el volumen de eventos acumulados para cada uno de los escenarios analizados. Observando estos resultados se tiene que la solución basada en umbrales estáticos genera 34 eventos, la del calculador de umbral 25 eventos y la que se apoya en hacer una estimación del modelo de perturbación 13 eventos. A la luz de estos resultados se comprueba que el sistema con el calculador de umbral genera un 26 % menos de eventos respecto al sistema con umbrales estáticos. Por último, el sistema que aplica un umbral en base a una estimación del ruido genera un 62 % menos de eventos que el sistema con umbral estático.

7.7. Posicionamiento de Robots Móviles

En el Capítulo 6 se han planteado nuevas ideas de como mejorar el posicionamiento de un robot mediante un nuevo algoritmo que se apoya en medidas de

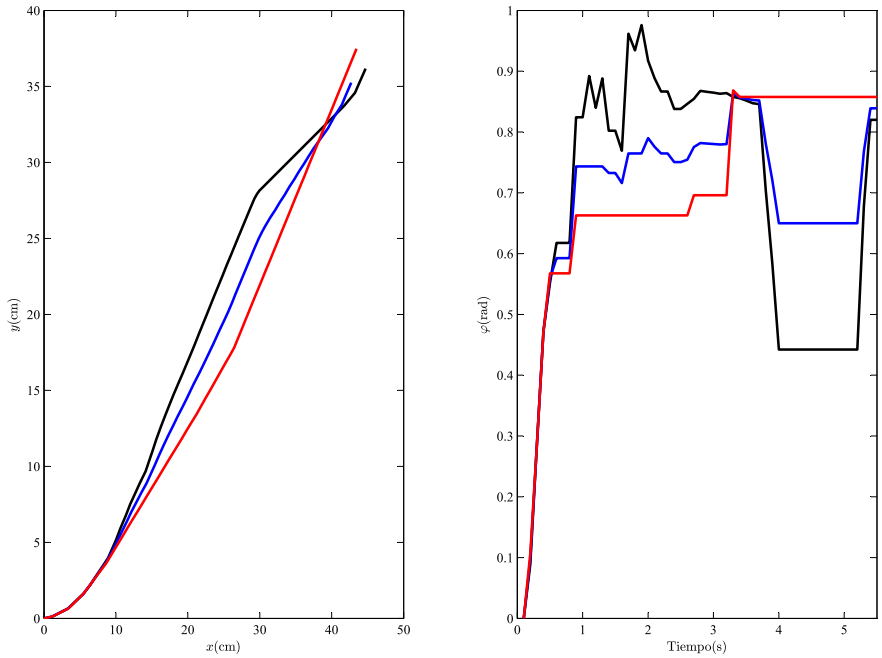

Figura 7.38: Posición y orientación. En trazo negro el escenario 1, en azul el escenario 2 y en rojo el escenario 3.

inclinación. La idea general se basa en un algoritmo controlado por una máquina de estados finitos (MEF) que aplica unos procedimientos de cálculo u otros en función de la inclinación del robot. Dicha inclinación, que se obtiene a través del acelerómetro, se combina con técnicas odométricas para estimar así el posicionamiento en 3D del robot. Por otro lado, y apoyado también en las medidas de aceleración, se propuso un sistema para sincronizar la orientación del robot y reducir los errores que producen las técnicas basadas en odometría.

Con el objetivo de analizar la respuesta de este algoritmo, en los siguientes experimentos se analizará el comportamiento de sus diferentes partes. En primer lugar se investigará el comportamiento de la MEF descrita en la Sección 6.6.3. Posteriormente, se verificarán los métodos de estimación de posicionamiento planteados en la Sección 6.6.4. Finalmente, se hará un análisis de la respuesta del sistema de sincronización propuesto en la Sección 6.7.

Una vez calibrado el acelerómetro y los sensores de distancia que obtienen las medidas para la estimación basada en odometría, se fijan los parámetros del algoritmo, cuyos valores se presentan en la Tabla 7.10.

A continuación se analizan los resultados obtenidos de cada uno de los bloques

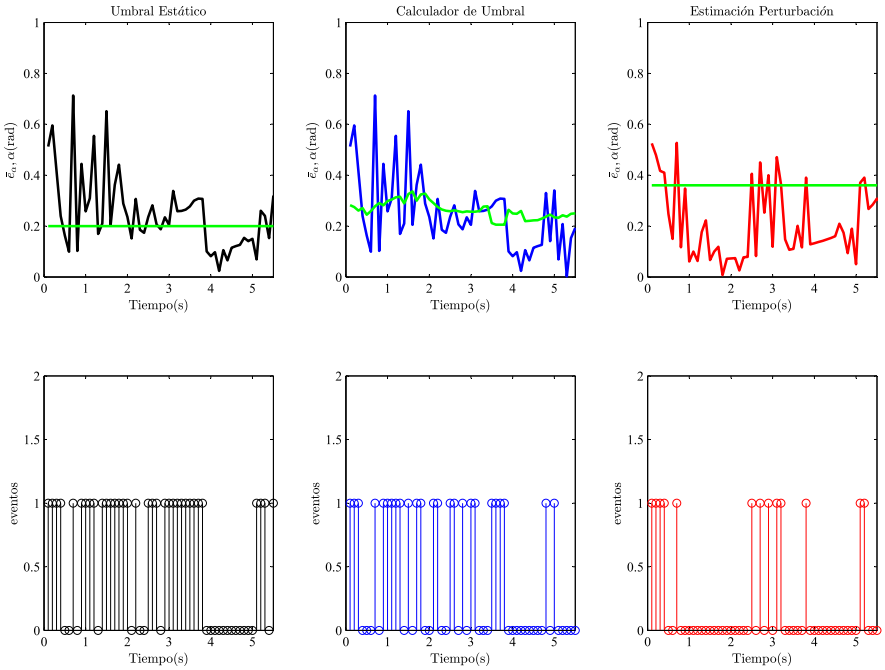

Figura 7.39: Error de orientación, umbral y eventos generados. En trazo negro el escenario 1, en azul el escenario 2 y en rojo el escenario 3.

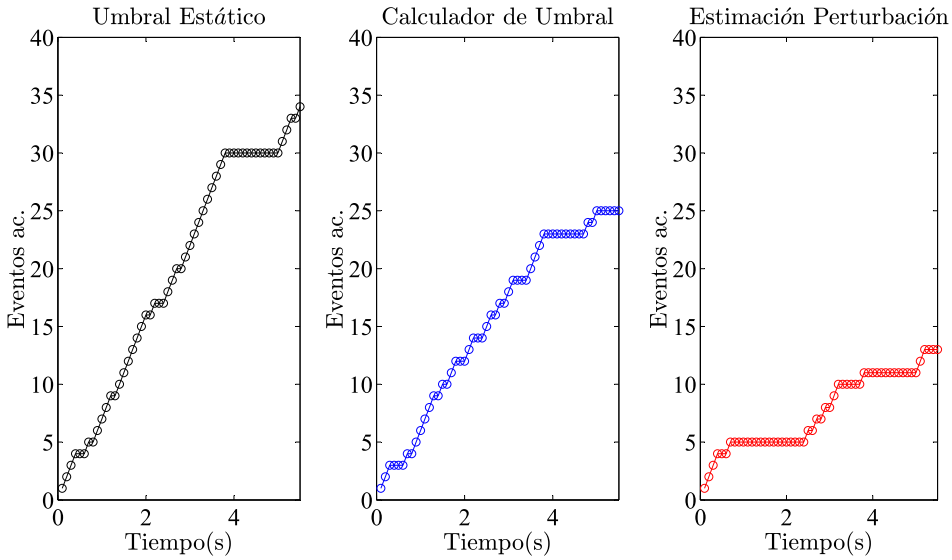

Figura 7.40: Eventos acumulados de los escenarios con ruido.

del algoritmo comentados anteriormente.

Tabla 7.10: Parámetros del algoritmo de estimación de posicionamiento.

Parámetro	Valor
Constante del Filtro	$\alpha = 0,9$
Longitud del Calculador de Umbral	$l = 20$
Bandas de Guarda	$H = 0,2°$, $h = 0,1°$
Tiempo Sistema Referencia	$t_{min} = 1$s

7.7.1. Comportamiento de la Máquina de Estados Finitos

Para comprobar el comportamiento de la MEF, el robot se ha puesto sobre una superficie móvil con un movimiento en línea recta y velocidad constante de $10,3$ cm/s. Seguidamente se aplicaron a la superficie diferentes grados de inclinación (elevación θ y alabeo ψ). El ángulo de elevación se ha variado desde $-8°$ hasta $+8°$. De la misma manera se ha modificado el ángulo de alabeo desde $-8°$ hasta $+8°$. En la Figura 7.41 se muestran los resultados obtenidos. Tal como se muestra

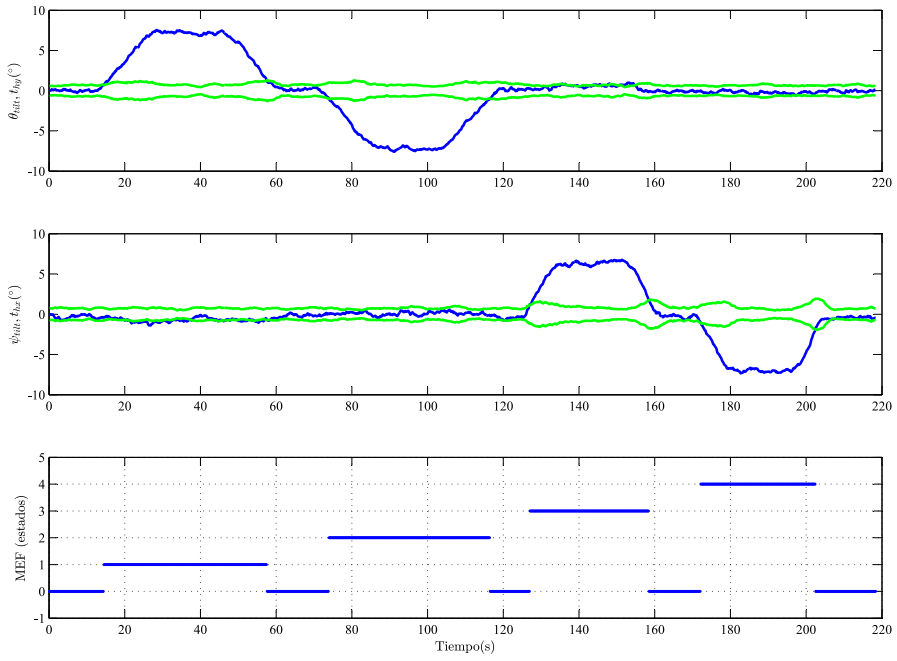

Figura 7.41: Comportamiento de la MEF. En trazo azul los ángulos de inclinación y en trazo verde los umbrales. En trazo azul recto se presentan los estados de la MEF.

en la gráfica, el comportamiento de los estados de la MEF es acorde al ángulo de inclinación al que es sometido el robot. Cuando los ángulos de inclinación son

cercanos a cero, la MEF está en estado 0. La MEF está en el Estado 1 o 2 según sea el valor de θ_{tilt} y en los Estados 3 o 4 dependiendo de lo que valga ψ_{tilt}. Finalmente, es importante destacar que el sistema es capaz de conmutar entre estados con un error inferior a 1°.

7.7.2. Estimación de Posición y Orientación en 3D

Una vez analizada la MEF, el siguiente paso es comprobar la respuesta del sistema para el posicionamiento en 3D. Para ello se han realizado dos experimentos:

- *Experimento 1.* El robot navega en línea recta con una velocidad constante de $10,3$ cm/s sobre una superficie con una elevación de $\theta = 15°$.
- *Experimento 2.* En este caso, el robot describe una trayectoria circular (velocidad de la rueda izquierda $17,5$ cm/s y velocidad rueda derecha $10,3$ cm/s) sobre una superficie con un ángulo de alabeo de $\psi = -20°$.

Ambos experimentos se han ejecutado durante 10 s describiendo los robots las trayectorias mostradas en la Figura 7.42.

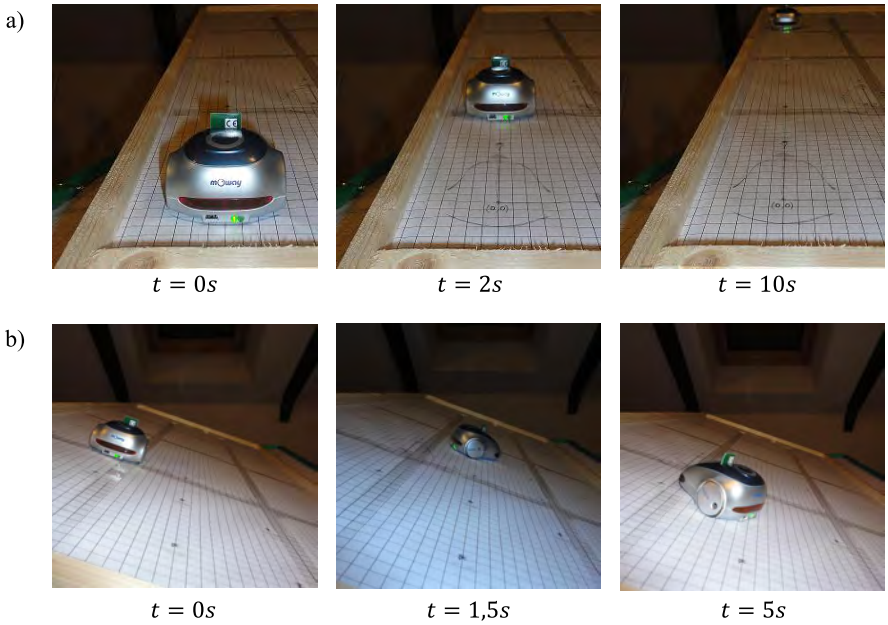

Figura 7.42: Trayectorias en 3D. a) Experimento 1, $\theta = 15°$ y $\psi = 0°$. b) Experimento 2, $\theta = 0°$ y $\psi = -20°$.

Para comprobar la precisión del algoritmo, las posiciones reales del robot $(x,\ y,\ z,\ \varphi,\ \theta,\ \psi)$ se han obtenido mediante el sistema de vídeo *Tracker*. Unos

parámetros se obtienen de forma directa a través de la herramienta y otros se calculan a través de las condiciones de contorno del problema. Los parámetros de estimación de posición ($\hat{x}$, $\hat{y}$, $\hat{z}$, $\hat{\varphi}$, $\hat{\theta}$, $\hat{\psi}$) los obtiene el algoritmo propuesto. Para el Experimento 1 se comparan ambas magnitudes en la Figura 7.43. Para el Experimento 2 se presentan los mismo parámetros en la Figura 7.44.

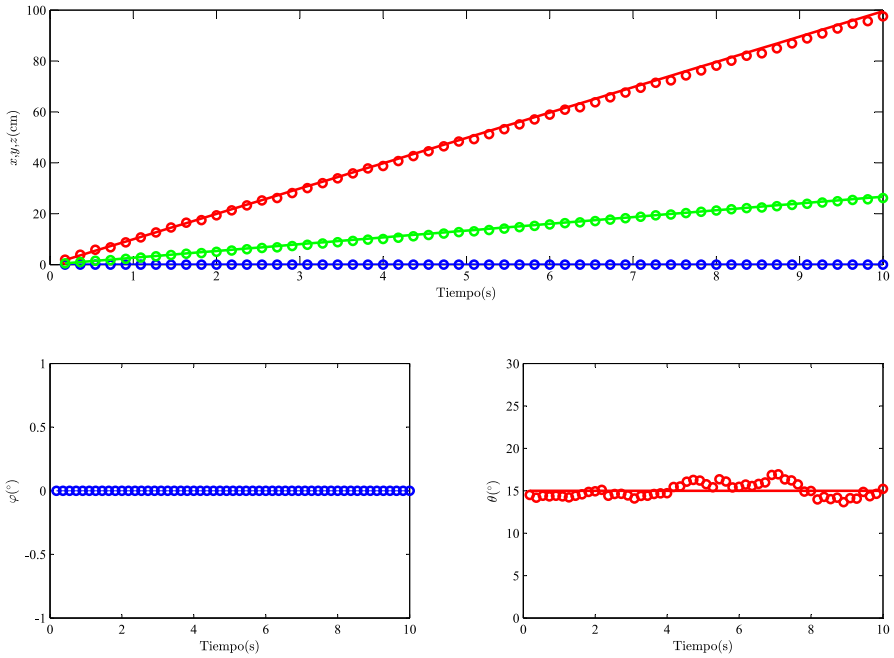

Figura 7.43: Posicionamiento 3D del experimento 1. En trazo continuo parámetros reales y en círculos la estimación mediante el algoritmo propuesto.

Tal como se muestra en las Figuras 7.43 y 7.44 el algoritmo propuesto estima de forma adecuada y sin inestabilidades la posición 3D del robot. Tanto la estimación del ángulo de elevación como del ángulo de alabeo no tienen errores acumulativos debido a que se obtienen directamente de las medidas del acelerómetro. El resto de parámetros, inevitablemente, tendrán este tipo de errores como consecuencia de las técnicas de odometría utilizadas para su estimación. Estos errores acumulativos se procurará minimizarlos con el sistema de sincronización planteado en este trabajo el cual se analizará en la siguiente sección.

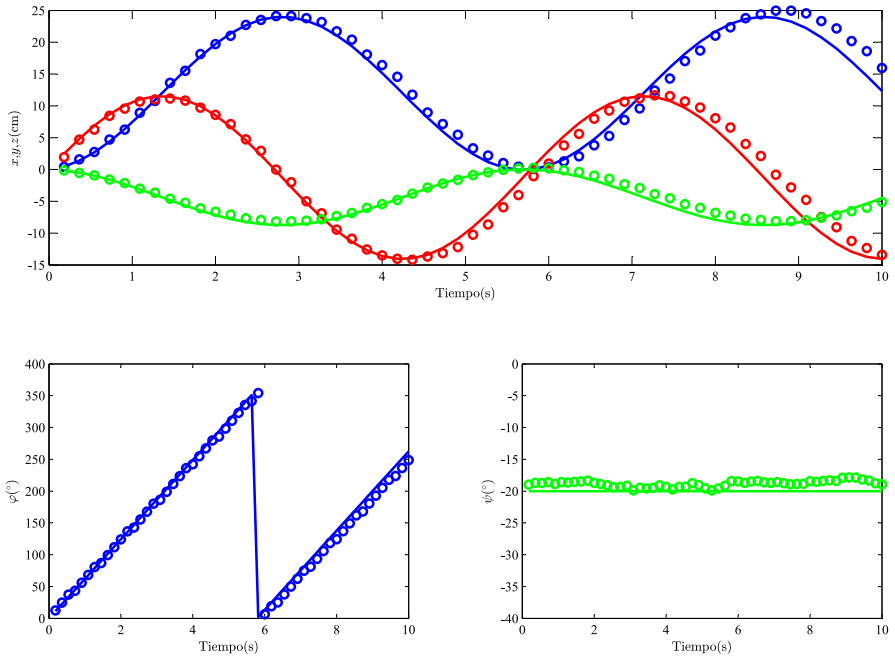

Figura 7.44: Posicionamiento 3D del experimento 2. En trazo continuo parámetros reales y en círculos la estimación mediante el algoritmo propuesto.

7.7.3. Sistema de Sincronización de la Orientación

A continuación se realizará un nuevo experimento para analizar cómo el sistema de sincronización propuesto puede mejorar la estimación del posicionamiento 3D del robot. En este caso, se ha programado el robot con la trayectoria mostrada en la Figura 7.45. Nuevamente la trayectoria real del robot se ha obtenido con el sistema de visión *Tracker* y la estimación del posicionamiento mediante el algoritmo propuesto. Para la estimación basada en el algoritmo se han usado dos métodos diferentes:

- Método 1. Estimación de la posición con el sistema de sincronización desactivado.
- Método 2. Estimación con el sistema de sincronización activo.

El hecho de hacer las estimaciones con y sin el sistema de sincronización, servirá de marco de comparación y poder analizar así qué ventajas aporta este nuevo elemento al sistema global.

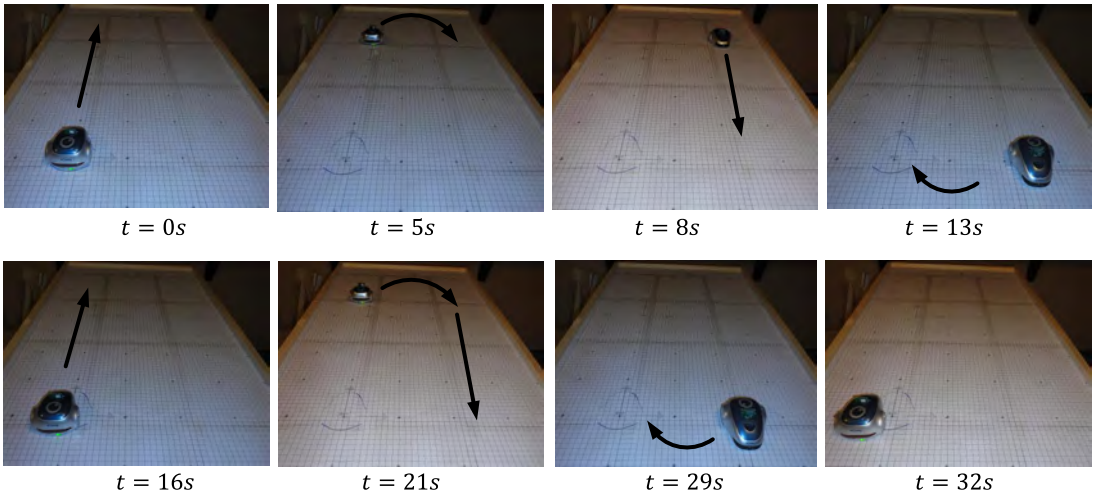

Figura 7.45: Trayectoria del experimento de sincronización.

El experimento se ha ejecutado durante 32 segundos, obteniéndose los ángulos de orientación mostrados en la Figura 7.46, donde se presenta tanto la orientación real obtenida con el *Tracker* como las dos estimaciones considerando y sin considerar el sistema sincronización.

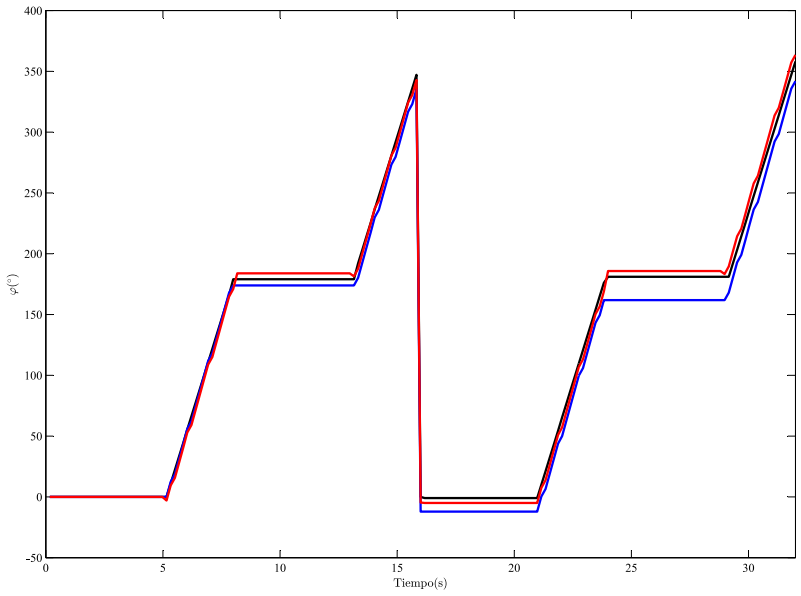

Figura 7.46: Estimación del ángulo de orientación. En trazo negro la orientación real, en trazo azul orientación sin sincronización y en trazo rojo con el sistema de sincronización activo.

Tal como se observa en la Figura 7.46, el error en la estimación del ángulo de orientación se incrementa a lo largo del tiempo cuando no está activo el sistema de

sincronización. En cambio, cuando este sistema se activa, el error en la estimación del ángulo de orientación del robot permanece constante en el tiempo.

Posteriormente se analizan los parámetros de posición del robot, tanto los capturados con el sistema *Tracker* (x, y, z) como los estimados por el algoritmo bajo análisis ($\hat{x}$, $\hat{y}$, $\hat{z}$), en la Figura 7.47 se muestran los resultados obtenidos sobre el plano $Oxyz$. En la Figura 7.48 se presentan los resultados si se considera solo el plano Oxy.

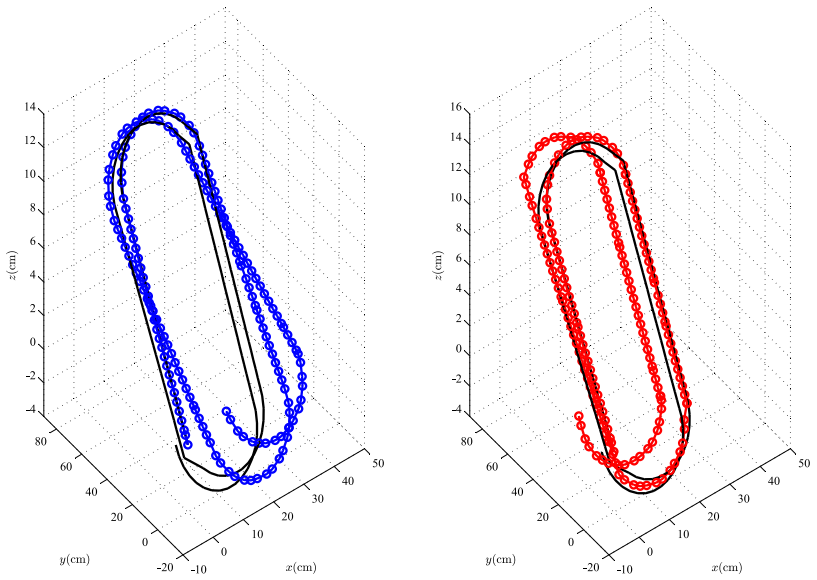

Figura 7.47: Posición en 3D. En trazo negro posición real, círculos azules sin sistema de sincornización y círculos rojos con el sistema de sincronización activo.

Tal como se muestra en las Figuras 7.47 y 7.48, cuando los parámetros de posición son estimados con el sistema de sincronización activo, el error cometido en la estimación de la posición se reduce considerablemente. Con el objeto de evaluar el error cometido con cada una de las estimaciones, se puede definir el error de posicionamiento P_e según la ecuación:

$$P_e = \|\mathbf{p} - \hat{\mathbf{p}}\| \tag{7.14}$$

donde $\mathbf{p} = (x, y, z)^T$ es la posición del robot obtenida con el sistema *Tracker* y $\hat{\mathbf{p}} = (\hat{x}, \hat{y}, \hat{z})^T$ la estimación de la posición obtenida por el algoritmo propuesto. En la Figura 7.49 se muestra el error de posicionamiento considerando $\hat{\mathbf{p}}$ con el sistema de sincronización desactivado y posteriormente con este sistema activo.

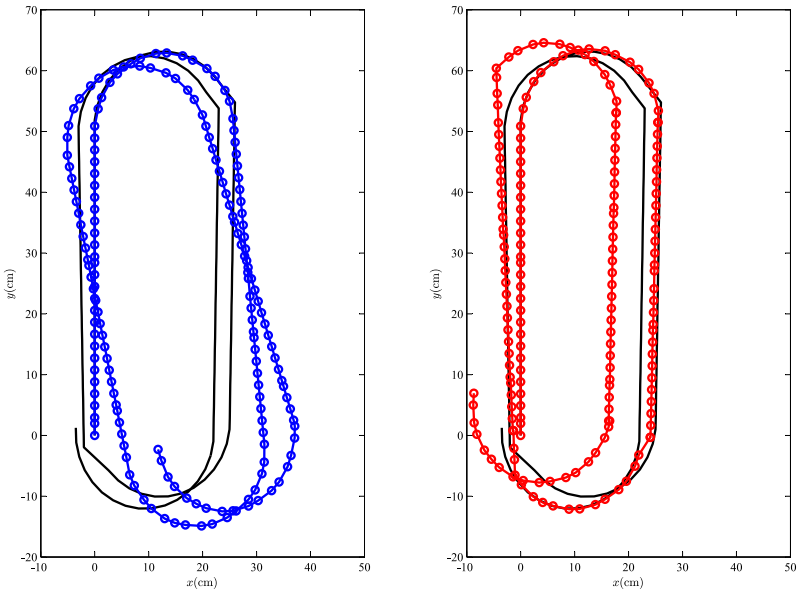

Figura 7.48: Posición en 2D. En trazo negro posición real, círculos azules sin sistema de sincornización y círculos rojos con el sistema de sincronización activo.

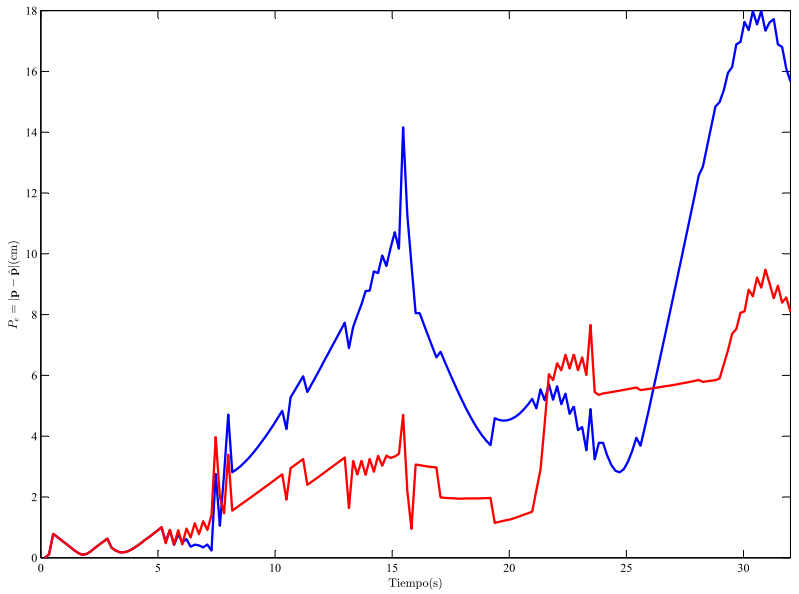

Figura 7.49: Error de posicionamiento. En trazo azul sin sistema de sincronización y en trazo rojo con el sistema de sincronización activo.

Si se analizan los resultados presentados en la Figura 7.49, se observa que, aunque ambas metodologías tienen un error de posicionamiento acumulativo, cuando

se usa el sistema de sincronización propuesto, este error se reduce considerablemente.

7.8. Conclusiones

En el presente capítulo se han recopilado los resultados experimentales de las ideas propuestas en esta tesis doctoral. Se ha verificado experimentalmente que, la arquitectura de control basada en eventos propuesta y los algoritmos desarrollados sobre este esquema de control tienen una respuesta equivalente a los sistemas clásicos de tiempo discreto. Se ha contrastado también que dichos algoritmos resuelven el problema de navegación de forma precisa y sin inestabilidades, presentando además mejoras considerables en el consumo de recursos respecto a las soluciones clásicas. Tras analizar el sistema en presencia de perturbaciones, se verifica que al utilizar las metodologías de modelado de ruido y los esquemas de estimación propuestos, los sistemas de control desarrollados en esta tesis mejoran la robustez frente al ruido. Por último, se ha analizado de forma experimental el algoritmo de posicionamiento planteado encontrándose que mejora de forma considerable las estimaciones de posición y orientación de los robots respecto a los esquemas tradicionales basados en odometría.

Capítulo 8

Conclusiones y Líneas Futuras

Se exponen a continuación las conclusiones obtenidas en el desarrollo de esta tesis, así como las posibles líneas de investigación futuras que se derivan de este trabajo.

8.1. Conclusiones

Como conclusiones principales de esta tesis se enumeran las siguientes:

- Los sistemas de control clásicos aplicados a problemas de navegación en robots móviles tiene un campo de mejora en cuando a la reducción del consumo de recursos aplicando técnicas de control basadas en eventos.
- Los esquemas de control propuestos en esta tesis resuelven el problema de navegación en robots móviles de forma satisfactoria, sin inestabilidades y con una respuesta equivalente a los sistemas clásicos. Por tanto, los soluciones planteadas son claras sustitutas a los esquemas tradicionales de control.
- Desde un punto de vista del uso de los recursos, las arquitecturas desarrolladas presentan grandes eficiencias en el consumo de ancho de banda, carga computacional y gasto energético. Por otro lado, la sintonía de estos nuevos esquemas de control se hace de forma sencilla e intuitiva. En esta tesis, se presentan las reglas para sintonizar estos esquemas en algoritmos de navegación convencional y su posible extensión en algoritmos más complejos.
- En este trabajo se han desarrollado tanto de forma teórica como experimental los esquemas de control basados en eventos para robots móviles

diferenciales. Las ideas aquí planteadas pueden extenderse de forma sencilla e intuitiva a cualquier otro tipo de robot móvil cuyos esquemas de control tendrán características y prestaciones similares a los analizados en esta tesis.

- Se ha estudiado que el ruido en los sistemas de control basados en eventos impacta de forma negativa en el consumo de recursos, por tanto, en presencia de estas perturbaciones estos esquemas pierden las propiedades diferenciales frente a los esquemas tradicionales. En este trabajo, se han planteado esquemas y metodologías de sintonía que minimizan estos efectos, además, se consigue que las arquitecturas planteadas aumenten su inmunidad al ruido de medida.

- Las soluciones planteadas para sintonizar los esquemas de control propuestos en presencia de ruido tiene también una aplicación directa en el modelado de los sensores. Con estas técnicas se obtiene de forma sencilla un modelo teórico de los sensores, con ello, se permite mejorar de forma considerable cualquier aplicación que se desarrolle sobre estas plataformas.

- Un aspecto crucial en las aplicaciones de navegación de robots móviles es disponer de un sistema de posicionamiento preciso. En esta tesis, se proponen mecanismos de posicionamiento que mejoran de forma considerable las metodologías tradicionales, consiguiendo así, unos resultados más precisos en los algoritmos de navegación planteados en este trabajo.

8.2. Líneas de Trabajo Futuras

A partir del trabajo realizado en esta tesis se pueden plantear distintas líneas de investigación a desarrollar en trabajos futuros:

- Investigar la aplicación de los algoritmos de navegación planteados en esta tesis sobre otras plataformas de robots móviles o vehículos más complejos como pueden ser los robots móviles con patas (LMRs), los vehículos aéreos no tripulados (UAVs) o los vehículos autónomos acuáticos (AUVs).

- Las arquitecturas de control basadas en eventos aquí planteadas se han analizado experimentalmente sobre una plataforma que trabaja de forma

nativa en tiempo discreto. Este aspecto hace que el sistema no manifieste efectos tipo Zeno y que por tanto, no sea necesario aplicar mecanismos para corregirlos. Se plantea en futuros trabajos estudiar los algoritmos aquí desarrollados sobre plataformas que no estén condicionadas a un periodo de reloj, y desarrollar así, mecanismos que garanticen mejor la estabilidad de estos sistemas.

- La estabilidad de los sistemas de control aquí planteados se garantiza mediante la aplicación de metodologías clásicas de control continuo y discreto. En este sentido, queda por cubrir un amplio campo de investigación en el aspecto teórico para desarrollar metodologías que garanticen la estabilidad de estos sistemas con un enfoque de control basado en eventos.

- Los análisis de perturbaciones han estado centrados en el ruido presente en los sensores. Se plantea analizar también las perturbaciones que afectan al comportamiento de la planta, abriendo así, una nueva línea de investigación para la sintonía de estos controladores.

- Finalmente, los mecanismos de posicionamiento propuestos se apoyan en sistemas de bajo coste y plataformas con un número limitado de sensores. Queda por tanto desarrollar estas metodologías en sistemas más complejos, con un mayor número de sensores y con mayores frecuencias de muestreo. Por tanto, se espera con estas nuevas condiciones conseguir una mayor precisión en las estimaciones obtenidas.

Apéndice A

Laboratorio de Experimentación

Para analizar de forma experimental las ideas planteadas en esta tesis, se ha desarrollado un laboratorio cuya estructura se presenta en la Figura A.1.

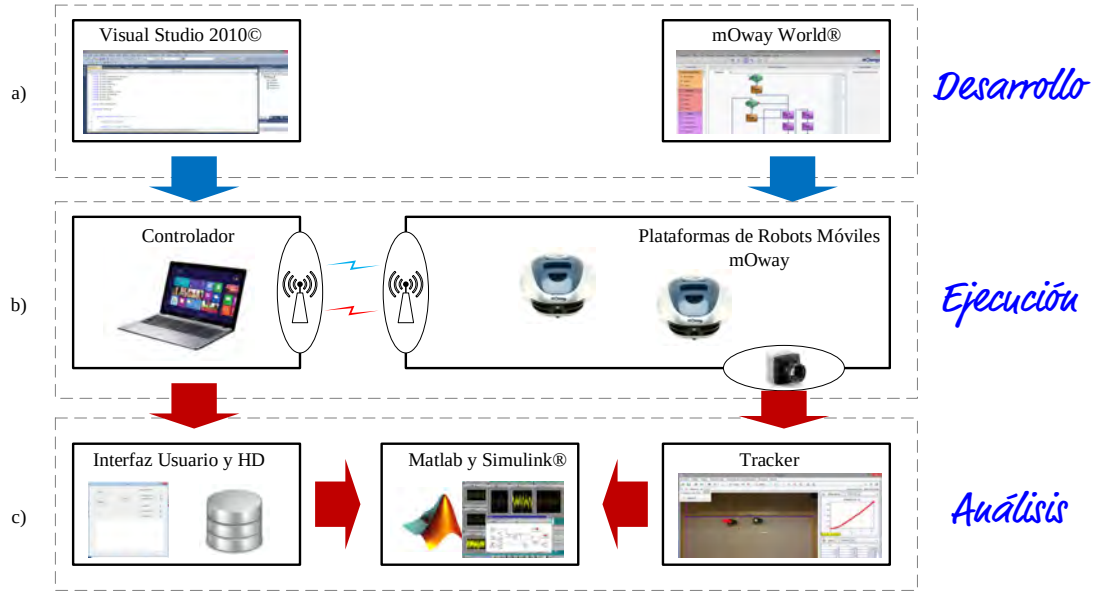

Figura A.1: Estructura funcional del laboratorio de experimentación. a) Módulo de desarrollo, b) módulo de ejecución y c) módulo de análisis.

Esta plataforma de experimentación se compone de tres módulos funcionales:

1. *Módulo de desarrollo.* Este bloque se compone de dos entornos de desarrollo diferentes. El primero se utiliza para la programación de los algoritmos del controlador y el segundo para el desarrollo de los algoritmos que se ejecutan en la plataformas de robots mOway (véase Figura A.1 a)). En este entorno, la plataforma *Visual Studio 2010* de Microsoft es la elegida para desarrollar

el código del controlador. Para el caso de las plataforma de robots, se ha usado el entorno *mOway World* del fabricante de la plataforma de robots mOway.

2. *Módulo de ejecución*. Este elemento consta del código del controlador ejecutándose sobre un ordenador de propósito general bajo sistema operativo *Windows* y de la plataformas de robots con el código correspondiente desarrollado en el entorno *mOway World* (véase Figura A.1 b)). Ambos elementos, el controlador y las plataformas de robots, se comunican mediante un enlace inalámbrico sobre la banda no licenciada de $2,4$ GHz (Figura A.2).

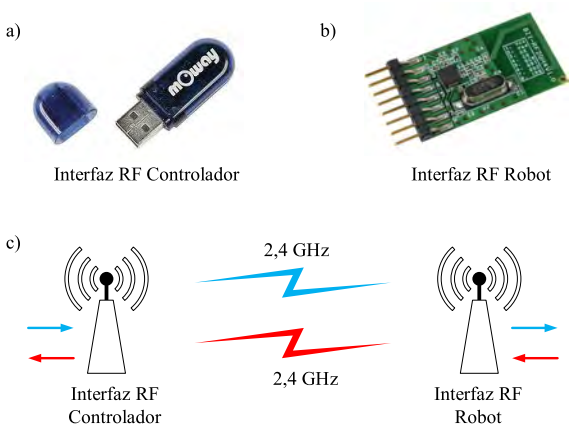

Figura A.2: Interfaz RF. a) Módem del controlador, b) módem de las plataformas de robots mOway y c) esquema funcional del sistema de comunicación inalámbrica.

3. *Módulo de análisis*. Este bloque se compone de la interfaz de usuario, una cámara de vídeo, el sistema *Tracker* y el entorno *Matlab / Simulink* (véase la Figura A.1 c)). La interfaz de usuario del controlador sirve para ejecutar los algoritmos de control y al mismo tiempo, todos los datos que procesa el controlador se almacenan en un disco duro (HD) para su posterior análisis. Por otro lado, mientras se está ejecutando un experimento, una cámara de vídeo captura las trayectorias de los robots en tiempo real para su posterior procesado. Estas imágenes son tratadas por el software *Tracker* para obtener las trayectorias reales seguidas por los robots en cada experimento. Por último, tanto los datos procesados por *Tracker* como los datos que captura el controlador, alimentan al entorno *Matlab / Simulink* donde son analizados y presentados los resultados de los experimentos.

Por último, en la Figura A.3 se presenta el esquema real de la plataforma de experimentación desarrollada en esta tesis.

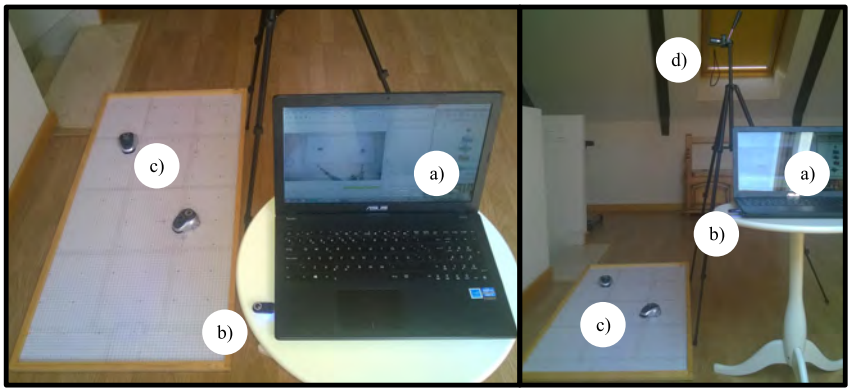

Figura A.3: Laboratorio de experimentación. a) Ordenador de propósito general, b) módem del controlador, c) plataforma de robots móviles con interfaz de RF y d) sistema de captura de vídeo.

Los elementos principales de esta plataforma son:

- *Ordenador de propósito general.* El ordenador contiene los dos módulos del bloque de desarrollo (*Visual Studio 2010* y *mOway World*), la interfaz de usuario (junto con el HD), el sistema *Tracker* y el entorno de desarrollo *Matlab / Simulink* (véase Figura A.3 a) y Figura A.1).

- *Módem del controlador.* Este elemento es módulo con una interfaz USB que implementa la comunicación inalámbrica con el robot en la banda de 2,4 GHz (véase Figura A.3 b) y Figura A.2).

- *Plataforma de robots mOway con interfaz RF.* Este se compone de varios robots mOway equipados con un módem RF (véase Figura A.3 c)).

- *Sistema de captura de vídeo.* Contiene una cámara de vídeo HD y un trípode para capturar las trayectorias que siguen los diferentes robots en los experimentos (véase Figura A.3 d)).

Apéndice B

Plataformas de Robots Móviles mOway

La plataforma de robot mOway desarrollada por la empresa Bizintek Innova, S.L. son unos pequeños robots autónomos programables diseñados principalmente para realizar aplicaciones prácticas de robótica móvil. El robot mOway está dotado de una serie de sensores que le ayudarán a desenvolverse en un entorno real. A su vez cuenta con un grupo motor que le permitirá desplazarse sobre el terreno. Todos estos periféricos están conectados a un microcontrolador que será el encargado de gobernar el robot. Este pequeño robot cuenta además con opciones de ampliación a través de un bus de expansión. En él se puede conectar, por ejemplo, un módulo de comunicaciones inalámbricas, una cámara de video, una tarjeta de prototipos o cualquier otro dispositivo que se considere interesante para el desempeño de una tarea.

Esta plataforma de robots se compone principalmente de los robots propiamente dichos, diferentes elementos de expansión (para este trabajo de tesis solo se ha utilizado el módulo de comunicación RF) y el entorno de desarrollo *mOway World* (véase Figura B.1).

Estos robots (Figura B.1 a)) contienen los siguientes subsistemas:

- PIC18F86J50 como microcontrolador principal.
- Grupo motor con control de trayectoria comandado por I2C.
- Sensores infrarrojos anticolisión.
- Sensor de intensidad de luz direccional.

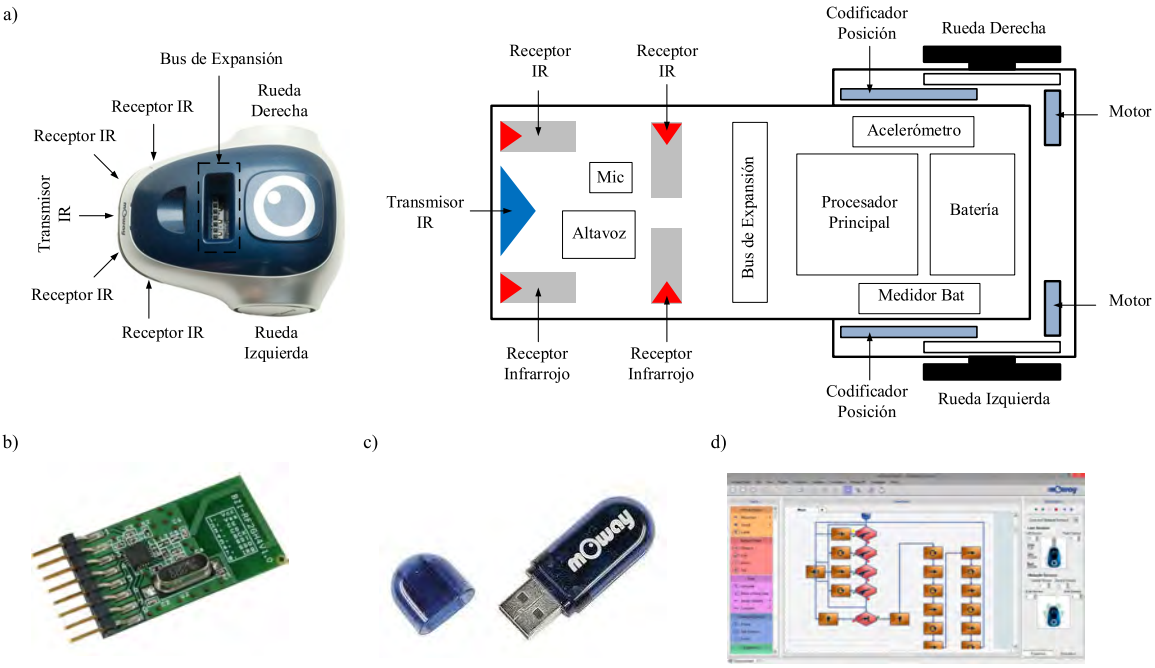

Figura B.1: Plataforma de robots mOway. a) Elementos del robot, b) módem RF del robot, c) módem RF del ordenador y d) entorno de desarrollo *mOway World*.

- Sensores optorreflectivos infrarrojos para el suelo.
- Indicador luminoso superior bicolor.
- Led frontal.
- Leds rojos traseros.
- Sensor de temperatura.
- Acelerómetro de 3 ejes.
- Micrófono.
- Altavoz.
- Bus de expansión SPI/I2C para tarjetas electrónicas.
- Módulo de radiofrecuencia para comunicación inalámbrica.
- Batería LI-PO recargable por USB.
- Autonomía de 2 horas.

- Entorno para robótica colaborativa.

Por otro lado, en el laboratorio de experimentación desarrollado para esta tesis también se han utilizado los módulos de comunicación inalámbrica para el robot y el ordenador (véase Figura B.1 b) y Figura B.1 c) respectivamente) y el entorno de desarrollo *mOway World* que suministra el fabricante de estos robots (véase Figura B.1 d)).

Apéndice C

Identificación y Calibración de la Plataforma mOway

C.1. Parámetros Geométricos y Dinámicos del Robot mOway

En la Figura C.1 se presentan los parámetros geométricos y dinámicos de la plataforma mOway utilizada en los experimentos de esta tesis. Este robot se

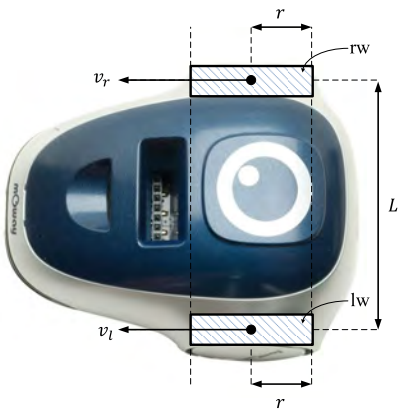

Figura C.1: Robot mOway. Parámetros geométricos y dinámicos.

compone de dos ruedas motrices, la rueda izquierda (lw) y la rueda derecha (rw). Los parámetros geométricos que modelan este sistema son el radio de las ruedas (r) y la distancia entre éstas (L). Por otro lado, los parámetros dinámicos son las velocidades lineales de cada una de las ruedas. En este caso, la velocidad de la rueda izquierda se denota por v_l y la de la rueda derecha por v_r. En la Tabla C.1 se presentan los valores numéricos de todos estos parámetros.

Tabla C.1: Valores números de los parámetros del robot mOway.

Parámetro	Valor numérico
r	$1,6$ cm
L	$6,6$ cm
v_l	min 0 cm/s; max $\pm 17,5$ cm/s
v_r	min 0 cm/s; max $\pm 17,5$ cm/s

C.2. Identificación de la Velocidad Lineal

El entorno del programación del robot mOway está diseñado de tal manera que para modificar la velocidad lineal hay que ejecutar un comando para cada una de las ruedas de forma independiente. Estos comandos tienen como parámetro de entrada la velocidad lineal que se aplica a cada rueda expresada en unidades relativas (%). Los valores de este parámetro varían entre un mínimo igual al 0 % hasta un máximo de ± 100 %. Por tanto, para poder trabajar con velocidades absolutas se ha hecho un proceso de identificación entre las velocidades lineales relativas (en %) y las velocidades lineales absolutas (en cm/s) y viceversa. En las Figuras C.2 y C.3 se presentan los resultados obtenidos.

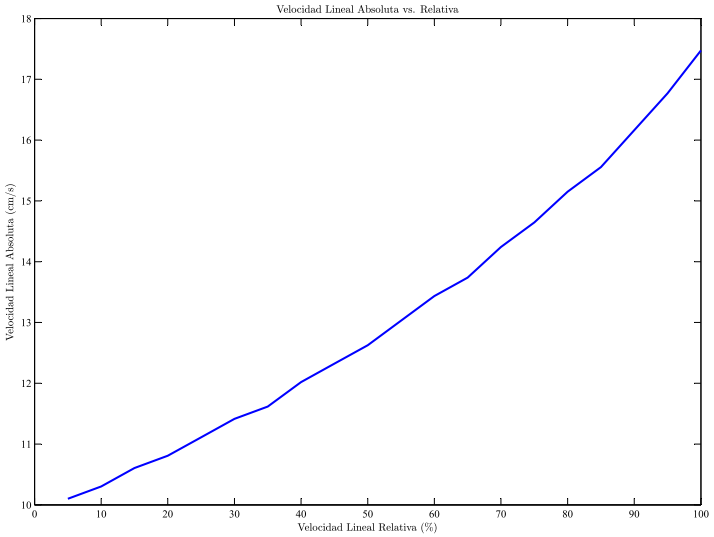

Figura C.2: Relación entre la velocidad lineal absoluta y la velocidad lineal relativa del robot mOway.

Este proceso de identificación se ha realizado a través del ajuste de un polinomio $p(x)$ (Ecuación C.1 donde x es la variable independiente) de grado n mediante

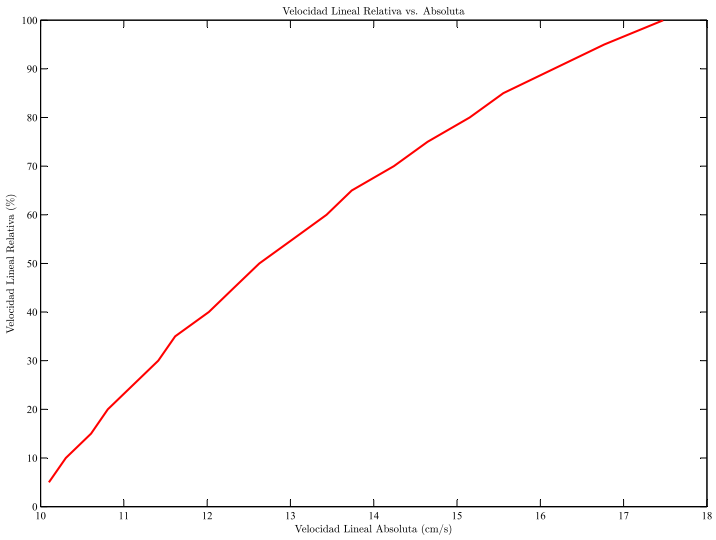

Figura C.3: Relación entre la velocidad lineal relativa y la velocidad lineal absoluta del robot mOway.

el método de mínimos cuadrados. En este caso, se han realizado varios experimentos fijando las velocidades lineales relativas vl_r y midiendo sus correspondientes velocidad lineales absolutas vl_a tal como se muestra en la Tabla C.2.

$$p(x) = p_1 x^n + p_2 x^{n-1} + \cdots + p_n x + p_{n+1} \tag{C.1}$$

A partir de los valores de la Tabla C.2 se han obtenido los polinomios correspondientes. En este caso, el polinomio $p_{ra}(x)$ que determina la velocidad lineal absoluta vl_a a partir de la velocidad lineal relativa vl_r mediante la expresión $vl_a = p_{ra}(vl_r)$. En cambio, para el proceso inverso, se ha obtenido también el polinomio $p_{ar}(x)$ que obtiene dichas velocidades mediante la ecuación $vl_r = p_{ar}(vl_a)$. Los coeficientes de dichos polinomios se muestran en las Tablas C.3 y C.4 respectivamente.

C.3. Identificación de los Sensores de Obstáculos

Los sensores de obstáculos como la mayoría de los sensores utilizados en robótica suelen tener una respuesta no lineal. En este caso, el proceso de identificación y calibración se ha hecho a partir de utilizar patrones conocidos y ajustar éstos

Tabla C.2: Relación entre las velocidades lineales absolutas y relativas durante el proceso de identificación.

Vel. Lineal Relativa - $vl_r(\%)$	Vel. Lineal Absoluta - vl_a (cm/s)
0	0,000
5	10,101
10	10,303
15	10,606
20	10,808
25	11,111
30	11,414
35	11,616
40	12,020
45	12,323
50	12,626
55	13,030
60	13,434
65	13,737
70	14,242
75	14,646
80	15,152
85	15,556
90	16,162
95	16,768
100	17,475

mediante polinomios. La metodología seguida para este proceso ha sido como en la sección anterior el método de los mínimos cuadrados. De esta forma, la medida de distancia calibrada d_c de un sensor de obstáculos se puede modelar mediante el polinomio que la relaciona con la medida tomada directamente del sensor d_{sc} (medida sin calibrar) a través de la expresión $d_c = p(d_{sc})$, donde $p(x)$ es el polinomio que las relaciona.

Para la plataforma de robots mOway se han utilizado dos polinomios diferentes. El primer polinomio, $p_f(x)$ para los sensores frontales del robot y el polinomio, $p_l(x)$ para los dos sensores laterales. En las Tablas C.5 y C.6 se presentan los coeficientes obtenidos para dichos polinomios.

Tabla C.3: Coeficientes del polinomio p_{ra}.

p_i	Valor
p_1	$2,4911 \cdot 10^{-11}$
p_2	$-6,6384 \cdot 10^{-9}$
p_3	$6,7781 \cdot 10^{-7}$
p_4	$-3,1889 \cdot 10^{-5}$
p_5	$9,5479 \cdot 10^{-4}$
p_6	$0,0362$
p_7	$9,8940$

Tabla C.4: Coeficientes del polinomio p_{ar}.

p_i	Valor
p_1	$-0,0115$
p_2	$0,6753$
p_3	$-15,5528$
p_4	$176,7816$
p_5	$-769,7338$

C.4. Calibración de las Medidas del Acelerómetro

El robot mOway utilizado en la plataforma de experimentación contiene el acelerómetro MMA7455L de Freescale Semiconductors de tres ejes. Dicho sensor tiene una sensibilidad que va desde $-2g$ hasta $+2g$ (donde $g = 9,81$ m/s^2) para cada uno de los ejes y cuya codificación en los registros internos del robot tiene un rango que van desde 0 (para una aceleración de $-2g$) hasta 255 (para $+2g$) (véase Figura C.4).

Para expresar las medidas de aceleración de este sensor en unidades de $g's$ deben utilizarse las Ecuaciones C.2-C.4.

$$a_{xg} = -(a_{xs} - 127{,}5)\frac{4}{256} \tag{C.2}$$

$$a_{yg} = -(a_{ys} - 127{,}5)\frac{4}{256} \tag{C.3}$$

$$a_{zg} = -(a_{zs} - 127{,}5)\frac{4}{256} \tag{C.4}$$

Tabla C.5: Coeficientes del polinomio p_f.

p_i	Valor
p_1	$-4,5317 \cdot 10^{-9}$
p_2	$2,0910 \cdot 10^{-6}$
p_3	$-2,9015 \cdot 10^{-4}$
p_4	$-0,0096$
p_5	$5,9836$

Tabla C.6: Coeficientes del polinomio p_l.

p_i	Valor
p_1	$2,4600 \cdot 10^{-10}$
p_2	$-1,4775 \cdot 10^{-7}$
p_3	$3,7629 \cdot 10^{-5}$
p_4	$-0,0119$
p_5	$2,0000$

donde a_{xs}, a_{ys} y a_{zs} son las aceleraciones que mide el sensor y a_{xg}, a_{yg} y a_{zg} son dichas medidas expresadas en $g's$. Tras el proceso de calibración basado en utilizar un patrón conocido, las medidas calibradas (a_x, a_y, a_z) se obtienen mediante las Ecuaciones C.5-C.7. En estas expresiones las variables a_{xbias}, a_{ybias} y a_{zbias} son los parámetros que se obtienen del proceso de calibrado del acelerómetro. Los valores numéricos de dichos parámetros son los siguientes:

- $a_{xbias} = 120,6477$
- $a_{ybias} = 105,6574$
- $a_{zbias} = 197,2968$

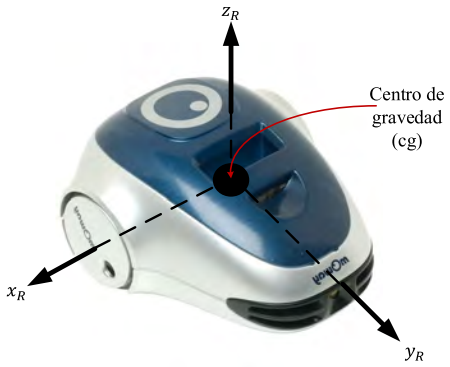

Figura C.4: Sistema de referencia para las medidas del acelerómetro.

$$a_x = -(a_{xg} - a_{xbias})\frac{4}{256} \tag{C.5}$$

$$a_y = -(a_{yg} - a_{ybias})\frac{4}{256} \tag{C.6}$$

$$a_z = -(a_{zg} - a_{zbias} + 64)\frac{4}{256} \tag{C.7}$$

Apéndice D

Distribuciones de Probabilidad

En los experimentos sobre modelado de perturbaciones se han utilizado varias distribuciones para encontrar el modelo que describa los datos bajo análisis. La metodología desarrollada consiste en ajustar los datos a una distribución de probabilidad por el método de máxima verosimilitud, para posteriormente, escoger aquella distribución que tenga un función de distribución acumulada $F(x)$ más cercana a los datos bajo estudio. A continuación se detallan las funciones de masa (VA discreta) o de densidad de probabilidad (VA continua) de las distribuciones analizadas en esta tesis.

D.1. Distribución Uniforme

$$f(x|a,b) = \begin{cases} \frac{1}{(b-a)} & ; a \leq x \leq b \\ 0 & ; \text{resto} \end{cases} \tag{D.1}$$

D.2. Distribución Normal

$$f(x|\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right) \tag{D.2}$$

D.3. Distribución Rayleigh

$$f(x|b) = \frac{x}{b^2} \exp\left(\frac{-x^2}{2b^2}\right) \tag{D.3}$$

D.4. Distribución Valor Extremo

$$f(x|\mu,\sigma) = \frac{1}{\sigma}\exp\left(\frac{x-\mu}{\sigma}\right)\exp\left(-\exp\left(\frac{x-\mu}{\sigma}\right)\right) \tag{D.4}$$

D.5. Distribución Geométrica

$$f(x|p) = p(1-p)^x \tag{D.5}$$

D.6. Distribución Logística

$$f(x|\mu,\sigma) = \frac{\exp\left(\frac{x-\mu}{\sigma}\right)}{\sigma\left(1+\exp\left(\frac{x-\mu}{\sigma}\right)\right)^2} \tag{D.6}$$

D.7. Función de Distribución Acumulada

Para cada una de las distribuciones anteriores, la función de distribución acumulada $F(x)$ se obtiene según la Ecuación D.7 para VA discretas y mediante la Ecuación D.8 para VA continuas.

$$F(x) = \sum_{x_i \leq x} f(x_i) \tag{D.7}$$

$$F(x) = \int_{-\infty}^{x} f(u) \cdot du \tag{D.8}$$

Bibliografía

[1] Tanveer Abbas, Muhammad Arif, and Waqas Ahmed. Measurement and correction of systematic odometry errors caused by kinematics imperfections in mobile robots. En *SICE-ICASE, 2006. International Joint Conference*, págs. 2073–2078. IEEE, 2006.

[2] Michele Aicardi, Giuseppe Casalino, Antonio Bicchi, and Aldo Balestrino. Closed loop steering of unicycle like vehicles via lyapunov techniques. *Robotics & Automation Magazine, IEEE*, 2(1):27–35, 1995.

[3] James C Alexander and John H Maddocks. On the kinematics of wheeled mobile robots. En *Autonomous robot vehicles*, págs. 5–24. Springer, 1990.

[4] Joseph A Angelo. *Robotics: a reference guide to the new technology*. Libraries Unlimited, 2007.

[5] Adolfo Anta and Paulo Tabuada. On the minimum attention and anytime attention problems for nonlinear systems. En *Decision and Control (CDC), 2010 49th IEEE Conference on*, págs. 3234–3239. IEEE, 2010.

[6] Adolfo Anta and Paulo Tabuada. To sample or not to sample: Self-triggered control for nonlinear systems. *Automatic Control, IEEE Transactions on*, 55(9):2030–2042, 2010.

[7] José Araújo. Design, implementation and validation of resource-aware and resilient wireless networked control systems. 2014.

[8] Khaled Arisha, Moustafa Youssef, and Mohamed Younis. Energy-aware tdma-based mac for sensor networks. En *System-level power optimization for wireless multimedia communication*, págs. 21–40. Springer, 2002.

[9] Karl-Erik Årzén. A simple event-based pid controller. En *Proc. 14th IFAC World Congress*, tomo 18, págs. 423–428. 1999.

[10] Alessandro Astolfi. Exponential stabilization of a wheeled mobile robot via discontinuous control. *Journal of dynamic systems, measurement, and control*, 121(1):121–126, 1999.

[11] Karl J Åström. B. wittenmark computer-controlled systems: Theory and design. 1990.

[12] Karl J Åström. Event based control. En *Analysis and design of nonlinear control systems*, págs. 127–147. Springer, 2008.

[13] Karl Johan Åström and Bo Bernhardsson. Comparison of riemann and lebesque sampling for first order stochastic systems. En *Proceedings of the 41st IEEE Conference on Decision and Control, 2002*, tomo 2, págs. 2011–2016. IEEE, 2002.

[14] Karl Johan Åström and Tore Hägglund. *Advanced PID control*. ISA-The Instrumentation, Systems, and Automation Society; Research Triangle Park, NC 27709, 2006.

[15] K auf der Heide, K Janschek, and A Tkocz. Synergy in power and momentum management for spacecraft using double gimbaled solar arrays. 2004.

[16] Joao P Barreto and Helder Araujo. Issues on the geometry of central catadioptric image formation. En *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, tomo 2, págs. II–422. IEEE, 2001.

[17] Randal W Beard and Timothy W McLain. *Small unmanned aircraft: Theory and practice*. Princeton university press, 2012.

[18] Alberto Bemporad, Maurice Heemels, and Mikael Johansson. *Networked control systems*, tomo 406. Springer, 2010.

[19] José Miguel Betancort, Luis Gómez, and Rafael Socas. Modelado y diseño de un sistema de control para la dinámica de vuelo de un quadrotor. En *Proyecto Fin de Grado*. Departamento de Electrónica y Automática. Universidad de Las Palmas de Gran Canaria, 2016.

[20] Johann Borenstein and Liqiang Feng. Correction of systematic odometry errors in mobile robots. En *Intelligent Robots and Systems 95.'Human*

Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on, tomo 3, págs. 569–574. IEEE, 1995.

[21] Julio H Braslavsky, Rick H Middleton, and Jim S Freudenberg. Feedback stabilization over signal-to-noise ratio constrained channels. En *American Control Conference, 2004. Proceedings of the 2004*, tomo 6, págs. 4903–4908. IEEE, 2004.

[22] Wolfram Burgard, Dieter Fox, Daniel Hennig, and Timo Schmidt. Estimating the absolute position of a mobile robot using position probability grids. En *Proceedings of the national conference on artificial intelligence*, págs. 896–901. Citeseer, 1996.

[23] Anton Cervin and Karl Johan Åström. On limit cycles in event-based control systems. En *Decision and Control, 2007 46th IEEE Conference on*, págs. 3190–3195. IEEE, 2007.

[24] Anton Cervin and Toivo Henningsson. Scheduling of event-triggered controllers on a shared network. En *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*, págs. 3601–3606. IEEE, 2008.

[25] DD Chaudhary, SP Nayse, and LM Waghmare. Application of wireless sensor networks for greenhouse parameter control in precision agriculture. *International Journal of Wireless & Mobile Networks (IJWMN) Vol*, 3(1):140–149, 2011.

[26] Frédéric Chenavier and James L Crowley. Position estimation for a mobile robot using vision and odometry. En *Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on*, págs. 2588–2593. IEEE, 1992.

[27] Marieke BG Cloosterman, Nathan Van de Wouw, WPMH Heemels, and Hendrik Nijmeijer. Stability of networked control systems with uncertain time-varying delays. *Automatic Control, IEEE Transactions on*, 54(7):1575–1580, 2009.

[28] Michael V Cook. *Flight dynamics principles: a linear systems approach to aircraft stability and control*. Butterworth-Heinemann, 2012.

[29] D De Bruin and PPJ van den Bosch. Measurement of the lateral vehicle position with permanent magnets. En *IFAC Workshop on Intelligent Components for Vehicles*, págs. 9–14. 1998.

[30] Manuel de la Sen. Non-periodic and adaptive sampling. a tutorial review. *Informatica*, 7(2):175–228, 1996.

[31] David DeVon and Timothy Bretl. Kinematic and dynamic control of a wheeled mobile robot. En *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, págs. 4065–4070. IEEE, 2007.

[32] MCF Donkers, Paulo Tabuada, and WPMH Heemels. Minimum attention control for linear systems. *Discrete Event Dynamic Systems*, 24(2):199–218, 2014.

[33] RC Dorf, MC Farren, and CA Phillips. Adaptive sampling frequency for sampled-data control systems. *Automatic Control, IRE Transactions on*, 7(1):38–47, 1962.

[34] S Dormido, M de la Sen, and M Mellado. Criterios generales de determinación de leyes de maestro adaptivo. *Revista de Informática y Automática*, 38:13–29, 1978.

[35] S Dormido and M Mellado. A study on fixed-difference sampling scheme. *Applications and Research in Information Systems and Sciences*, 2:496–500, 1977.

[36] Sebastián Dormido, J Sánchez, and Ernesto Kofman. Muestreo, control y comunicación basados en eventos. *Revista Iberoamericana de Automática e Informática Industrial RIAI*, 5(1):5–26, 2008.

[37] Neil A Duffie. An approach to the design of distributed machinery control systems. *Industry Applications, IEEE Transactions on*, (4):435–442, 1982.

[38] Sylvain Durand and Nicolas Marchand. An event-based pid controller with low computational cost. En *8th International Conference on Sampling Theory and Applications (SampTA'09)*, págs. Special–session. 2009.

[39] Nicola Elia and Sanjoy K Mitter. Stabilization of linear systems with limited information. *Automatic Control, IEEE Transactions on*, 46(9):1384–1400, 2001.

[40] Phillip H Ellis. Extension of phase plane analysis to quantized systems. *Automatic Control, IRE Transactions on*, 4(2):43–54, 1959.

[41] Philip H Enslow. What is a"distributed"data processing system? *Computer*, 11(1):13–21, 1978.

[42] Lyndon Evans. The large hadron collider: a marvel of technology. fundamental sciences. 2009.

[43] HR Everett. *Sensors for mobile robots: theory and application*. AK Peters, Ltd., 1995.

[44] Fabio Fagnani and Sandro Zampieri. Stability analysis and synthesis for scalar linear systems with a quantized feedback. *Automatic Control, IEEE Transactions on*, 48(9):1569–1584, 2003.

[45] Mohammed Faisal, Ramdane Hedjar, Mansour Alsulaiman, Khalid Al-Mutabe, and Hassan Mathkour. Robot localization using extended kalman filter with infrared sensor. En *Computer Systems and Applications (AICCSA), 2014 IEEE/ACS 11th International Conference on*, págs. 356–360. IEEE, 2014.

[46] D Flippo and DP Miller. Advantages of anisotropic wheels in skid steer turns. En *Volume1-Issue1 - 2016*, págs. 1(1): 1–5. International Journal of Advanced Robotics and Automation, 2016.

[47] William R. Franta, E. Douglas Jensen, Richard Y. Kain, and George D. Marshall. Real-time distributed computer systems. *Advances in Computers*, 20:39–82, 1981.

[48] Christopher Michael Geyer. *Catadioptric Projective Geometry: theory and applications*. Tesis Doctoral, University of Pennsylvania, 2003.

[49] Xiaojin Gong. *Omnidirectional vision for an autonomous surface vehicle*. Tesis Doctoral, Virginia Polytechnic Institute and State University, 2008.

[50] J Gorgas, N Cardiel, and J Zamorano. Estadística básica para estudiantes de ciencias. *Fac. de CC Físicas. Universidad complutense de Madrid*, 2009.

[51] Luca Greco, Daniele Fontanelli, and Antonio Bicchi. Design and stability analysis for anytime control via stochastic scheduling. *Automatic Control, IEEE Transactions on*, 56(3):571–585, 2011.

[52] María Guinaldo, Dimos V Dimarogonas, Karl H Johansson, José Sánchez, and Sebastián Dormido. Distributed event-based control strategies for interconnected linear systems. *Control Theory & Applications, IET*, 7(6):877–886, 2013.

[53] Vijay Gupta. On an anytime algorithm for control. En *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*, págs. 6218–6223. IEEE, 2009.

[54] WPMH Heemels and MCF Donkers. Model-based periodic event-triggered control for linear systems. *Automatica*, 49(3):698–711, 2013.

[55] WPMH Heemels, RJA Gorter, A van Zijl, PPJ Van den Bosch, Siep Weiland, WHA Hendrix, and MR Vonder. Asynchronous measurement and control: a case study on motor synchronization. *Control Engineering Practice*, 7(12):1467–1482, 1999.

[56] WPMH Heemels, JH Sandee, and PPJ Van Den Bosch. Analysis of event-driven controllers for linear systems. *International journal of control*, 81(4):571–590, 2008.

[57] WPMH Heemels and N van de Wouw. Stability and stabilization of networked control systems. En *Networked Control Systems*, págs. 203–253. Springer, 2010.

[58] Joao Hespanha, Antonio Ortega, and Lavanya Vasudevan. Towards the control of linear systems with minimum bit-rate. En *15th Int. Symp. Mathematical Theory of Networks and Systems (MTNS)*. 2002.

[59] HIROSHI Inose, T Aoki, and K Watanabe. Asynchronous delta-modulation system. *Electronics Letters*, 2(3):95–96, 1966.

[60] Abhishek Jha and Manoj Kumar. Two wheels differential type odometry for mobile robots. En *Reliability, Infocom Technologies and Optimization*

(ICRITO)(Trends and Future Directions), 2014 3rd International Conference on, págs. 1–5. IEEE, 2014.

[61] Yu Jianyong, Yu Shimin, and Wang Haiqing. Survey on the performance analysis of networked control systems. En *Systems, Man and Cybernetics, 2004 IEEE International Conference on*, tomo 6, págs. 5068–5073. IEEE, 2004.

[62] HW Ka, D Ding, and RA Cooper. Three dimensional computer vision-based alternative control method for assistive robotic manipulator. En *Volume1-Issue1 - 2016*, págs. 1(1): 1–6. International Journal of Advanced Robotics and Automation, 2016.

[63] Chung-Yao Kao and Bo Lincoln. Simple stability criteria for systems with time-varying delays. *Automatica*, 40(8):1429–1434, 2004.

[64] Hassan K Khalil. Nonlinear systems, third. 2002.

[65] Hassan K Khalil and JW Grizzle. *Nonlinear systems*, tomo 3. Prentice hall New Jersey, 1996.

[66] Alain Y Kibangou. Distributed estimation over unknown fading channels. En *2nd IFAC Workshop on Distributed Estimation and Control in Networked Systems (NecSys'10)*. 2010.

[67] Hermann Kopetz. Real time in distributed real time systems. En *Requirements Engineering*, págs. 240–247. Springer, 1983.

[68] Vangipuram Lakshmikantham, Srinivasa Leela, and Anatoliĭ Martynyuk. *Practical stability of nonlinear systems*.

[69] Yu-Cheol Lee and Seunghwan Park. Localization method for mobile robots moving on stairs in multi-floor environments. En *Systems, Man and Cybernetics (SMC), 2014 IEEE International Conference on*, págs. 4014–4020. IEEE, 2014.

[70] Alberto Leon-Garcia. *Probability, statistics, and random processes for electrical engineering*. Pearson/Prentice Hall, 2008.

[71] Shanbin Li, Zhi Wang, and Youxian Sun. Fundamental problems of networked control system from the view of control and scheduling. En *IECON 02 [Industrial Electronics Society, IEEE 2002 28th Annual Conference of the]*, tomo 3, págs. 2503–2508. IEEE, 2002.

[72] Xu-Guang Li, Arben Cela, Silviu-Iulian Niculescu, and Abdellatif Reama. Some problems in the stability of networked-control systems with periodic scheduling. *International Journal of Control*, 83(5):996–1008, 2010.

[73] Feng-Li Lian, John K Yook, Dawn M Tilbury, and James Moyne. Network architecture and communication modules for guaranteeing acceptable control and communication performance for networked multi-agent systems. *Industrial Informatics, IEEE Transactions on*, 2(1):12–24, 2006.

[74] Zhenying Liang and Chaoli Wang. Robust exponential stabilization of nonholonomic wheeled mobile robots with unknown visual parameters. *Journal of Control Theory and Applications*, 9(2):295–301, 2011.

[75] Daniel Liberzon and João P Hespanha. Stabilization of nonlinear systems with limited information feedback. *Automatic Control, IEEE Transactions on*, 50(6):910–915, 2005.

[76] Hugh HS Liu and Grantham KH Pang. Accelerometer for mobile robot positioning. *Industry Applications, IEEE Transactions on*, 37(3):812–819, 2001.

[77] L Liu and Q Zhang. Stability analysis of non-linear real-time network control systems. *JOURNAL-NORTHEASTERN UNIVERSITY NATURAL SCIENCE*, 29(3):305, 2008.

[78] Sergiusz Luczak, Waldemar Oleksiuk, and Maciej Bodnicki. Sensing tilt with mems accelerometers. *Sensors Journal, IEEE*, 6(6):1669–1675, 2006.

[79] Jan Lunze and Daniel Lehmann. A state-feedback approach to event-based control. *Automatica*, 46(1):211–215, 2010.

[80] Magdi S Mahmoud and Abdulla Ismail. Role of delays in networked control systems. En *Electronics, Circuits and Systems, 2003. ICECS 2003. Proceedings of the 2003 10th IEEE International Conference on*, tomo 1, págs. 40–43. IEEE, 2003.

[81] Sandeep Kumar Malu and Jharna Majumdar. Kinematics, localization and control of differential drive mobile robot. *Global Journal of Research and Engineering-GJRE-H*, 14(1), 2014.

[82] Riccardo Marino and Patrizio Tomei. *Nonlinear control design: geometric, adaptive and robust*. Prentice Hall International (UK) Ltd., 1996.

[83] Nuno C Martins and Munther A Dahleh. Feedback control in the presence of noisy channels:"bode-like" fundamental limitations of performance. *Automatic Control, IEEE Transactions on*, 53(7):1604–1615, 2008.

[84] Manuel Mazo, Adolfo Anta, and Paulo Tabuada. On self-triggered control for linear systems: Guarantees and complexity. En *Control Conference (ECC), 2009 European*, págs. 3767–3772. IEEE, 2009.

[85] Manuel Mazo Jr and Paulo Tabuada. On event-triggered and self-triggered control over sensor/actuator networks. En *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*, págs. 435–440. IEEE, 2008.

[86] Manuel Mazo Jr and Paulo Tabuada. Input-to-state stability of self-triggered control systems. En *CDC*, págs. 928–933. Citeseer, 2009.

[87] Pablo Millán, Luis Orihuela, Isabel Jurado, Carlos Vivas, and Francisco Rubio. Distributed estimation in networked systems under periodic and event-based communication policies. *International Journal of Systems Science*, 46(1):139–151, 2015.

[88] Pablo Millán, Luis Orihuela, Carlos Vivas, and Francisco Rubio. Improved delay-dependent stability criterion for uncertain networked control systems with induced time-varying delays. *IFAC Proceedings Volumes*, 42(20):346–351, 2009.

[89] Marek Miśkowicz. The event-triggered sampling optimization criterion for distributed networked monitoring and control systems. En *Industrial Technology, 2003 IEEE International Conference on*, tomo 2, págs. 1083–1088. IEEE, 2003.

[90] Marek Miskowicz. Improving the performance of the networked control system using event-triggered observations. *PDS'2004*, págs. 53–58, 2004.

[91] Marek Miskowicz. Send-on-delta concept: an event-based data reporting strategy. *sensors*, 6(1):49–63, 2006.

[92] Marek Miskowicz. Send-on-delta concept: an event-based data reporting strategy. *sensors*, 6(1):49–63, 2006.

[93] Marek Miskowicz. Asymptotic effectiveness of the event-based sampling according to the integral criterion. *Sensors*, 7(1):16–37, 2007.

[94] Adam Molin and Sandra Hirche. Adaptive event-triggered control over a shared network. En *Decision and Control (CDC), 2012 IEEE 51st Annual Conference on*, págs. 6591–6596. IEEE, 2012.

[95] Yi-Jen Mon, Chih-Min Lin, Imre J Rudas, et al. Wireless sensor network (wsn) control for indoor temperature monitoring. *Acta Polytechnica Hungarica*, 9(6):17–28, 2012.

[96] Luis A Montestruque and Panos J Antsaklis. On the model-based control of networked systems. *Automatica*, 39(10):1837–1843, 2003.

[97] George P Moustris and Spyros G Tzafestas. Switching fuzzy tracking control for mobile robots under curvature constraints. *Control Engineering Practice*, 19(1):45–53, 2011.

[98] Richard M Murray, Karl J Astrom, Stephen P Boyd, Roger W Brockett, and Gunter Stein. Future directions in control in an information-rich world. *IEEE Control Systems Magazine*, 23(2):20–33, 2003.

[99] Payam Naghshtabrizi and Joao P Hespanha. Designing an observer-based controller for a network control system. En *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC'05. 44th IEEE Conference on*, págs. 848–853. IEEE, 2005.

[100] Girish N Nair, Robin J Evans, Iven MY Mareels, and William Moran. Topological feedback entropy and nonlinear stabilization. *Automatic Control, IEEE Transactions on*, 49(9):1585–1597, 2004.

[101] Ulrich Nehmzow. *Robot behaviour: design, description, analysis and modelling*. Springer Science & Business Media, 2008.

[102] Ulrich Nehmzow. *Mobile robotics: a practical introduction*. Springer Science & Business Media, 2012.

[103] Vinh Hao Nguyen and Young Soo Suh. A modified multirate controller for networked control systems with a send-on-delta transmission method. En *Advanced Intelligent Computing Theories and Applications. With Aspects of Theoretical and Methodological Issues*, págs. 304–315. Springer, 2007.

[104] Katsuhiko Ogata. *Discrete-time control systems*, tomo 2. Prentice Hall Englewood Cliffs, NJ, 1995.

[105] Keitaro Ohno, Mitsuo Hirata, and Roberto Horowitz. A comparative study of the use of the generalized hold function for hdds. *Mechatronics, IEEE/ASME Transactions on*, 10(1):26–33, 2005.

[106] Luis Orihuela, Pablo Millán, Carlos Vivas, and Francisco Rubio. Reduced-order h 2/h inf distributed observer for sensor networks. *International Journal of Control*, 86(10):1870–1879, 2013.

[107] Paul G Otanez, James R Moyne, and Dawn M Tilbury. Using deadbands to reduce communication in networked control systems. En *American Control Conference, 2002. Proceedings of the 2002*, tomo 4, págs. 3015–3020. IEEE, 2002.

[108] A Pandey and DR Parhi. Multiple mobile robots navigation and obstacle avoidance using minimum rule based anfis network controller in the cluttered environment. En *Volume1-Issue1 - 2016*, págs. 1(1): 1–11. International Journal of Advanced Robotics and Automation, 2016.

[109] Andrzej Pawlowski, José Luis Guzmán, Francisco Rodríguez, Manuel Berenguel, J Sánchez, and Sebastián Dormido. The influence of event-based sampling techniques on data transmission and control performance. En *Emerging Technologies & Factory Automation, 2009. ETFA 2009. IEEE Conference on*, págs. 1–8. IEEE, 2009.

[110] Andrzej Pawlowski, Jose Luis Guzman, Francisco Rodríguez, Manuel Berenguel, José Sánchez, and Sebastián Dormido. Simulation of greenhouse climate monitoring and control with wireless sensor network and event-based control. *Sensors*, 9(1):232–252, 2009.

[111] Thongchai Phairoh and Keith Williamson. Autonomous mobile robots using real time kinematic signal correction and global positioning system control. En *Proceedings of IAJC-IJME International Conference on Industrial Technology, Nashville, TN, USA, 17-19 November*. 2008.

[112] Farzad Pourboghrat and Mattias P Karlsson. Adaptive control of dynamic mobile robots with nonholonomic constraints. *Computers & Electrical Engineering*, 28(4):241–253, 2002.

[113] S Ramaswamy and SN Balakrishnan. Formation control of car-like mobile robots: A lyapunov function based approach. En *American Control Conference, 2008*, págs. 657–662. IEEE, 2008.

[114] Saúl Robaina, Luis Gómez, and Rafael Socas. Diseño de un sistema electrónico para el pesaje automático y estimación del centro de gravedad de aviones ligeros. En *Proyecto Fin de Grado*. Departamento de Electrónica y Automática. Universidad de Las Palmas de Gran Canaria, 2015.

[115] Robert E Roberson and Richard Schwertassek. *Dynamics of multibody systems*. Springer Science & Business Media, 2012.

[116] Mariano Mellado Rodríguez, Sebastián Dormido Bencomo, J Ruiz Fernández, and José María Guillén Rubio. Sistemas de muestreo adaptativo mediante un criterio de diferencia de amplitudes constante. *Revista de automática*, 6(15):13–27, 1973.

[117] Julio Ariel Romero, Néstor J Pascual, Ignacio Peñarrocha, and Roberto Sanchis. Event-based pi controller with adaptive thresholds. En *Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2012 4th International Congress on*, págs. 219–226. IEEE, 2012.

[118] Anant Sahai. *Anytime information theory*. Tesis Doctoral, Massachusetts Institute of Technology, 2001.

[119] J Sánchez, M Guarnes, S Dormido, and A Visioli. Comparative study of event-based control strategies: An experimental approach on a simple tank. En *Control Conference (ECC), 2009 European*, págs. 1973–1978. IEEE, 2009.

[120] Jacobus Henk Sandee. Event-driven control in theory and practice. *Diss. PhD Thesis, Technische Universiteit Eindhoven, Netherlands*, 2006.

[121] Necip Sayiner, Henrik V Sorensen, and Thayamkulangara R Viswanathan. A level-crossing sampling scheme for a/d conversion. *Circuits and Systems II: Analog and Digital Signal Processing, IEEE Transactions on*, 43(4):335–339, 1996.

[122] Kurt Seifert and Oscar Camacho. Implementing positioning algorithms using accelerometers. *Freescale Semiconductor*, 2007.

[123] Georg S Seyboth, Dimos V Dimarogonas, and Karl H Johansson. Event-based broadcasting for multi-agent average consensus. *Automatica*, 49(1):245–252, 2013.

[124] Rahul C Shah and Jan M Rabaey. Energy aware routing for low energy ad hoc sensor networks. En *Wireless Communications and Networking Conference, 2002. WCNC2002. 2002 IEEE*, tomo 1, págs. 350–355. IEEE, 2002.

[125] Naim Sidek and Nilanjan Sarkar. Dynamic modeling and control of non-holonomic mobile robot with lateral slip. En *Systems, 2008. ICONS 08. Third International Conference on*, págs. 35–40. IEEE, 2008.

[126] Roland Siegwart, Illah Reza Nourbakhsh, and Davide Scaramuzza. *Introduction to autonomous mobile robots*. MIT press, 2011.

[127] Rafael Socas, Sebastián Dormido, and Raquel Dormido. Event-based controller for noisy environments. En *Complex Systems (WCCS), 2014 Second World Conference on*, págs. 280–285. IEEE, 2014.

[128] Rafael Socas, Sebastián Dormido, and Raquel Dormido. Event-based control strategy for the guidance of the aerosonde uav. En *Mobile Robots (ECMR), 2015 European Conference on*, págs. 1–6. IEEE, 2015.

[129] Rafael Socas, Sebastián Dormido, and Raquel Dormido. Optimal threshold setting for event-based control strategies. *IEEE Access*, 2017.

[130] Rafael Socas, Sebastián Dormido, Raquel Dormido, and Ernesto Fabregas. 3d positioning algorithm for low cost mobile robots. En *Informatics*

in Control, Automation and Robotics (ICINCO), 2015 12th International Conference on, tomo 2, págs. 5–14. IEEE, 2015.

[131] Rafael Socas, Sebastián Dormido, Raquel Dormido, and Ernesto Fabregas. Event-based control strategy for mobile robots in wireless environments. *Sensors*, 15(12):30076–30092, 2015.

[132] Rafael Socas, Sebastián Dormido, Raquel Dormido, and Ernesto Fabregas. Improving the 3d positioning for low cost mobile robots. En *Informatics in Control, Automation and Robotics 12th International Conference, ICINCO 2015 Colmar, France, July 21-23, 2015 Revised Selected Papers*, págs. 97–114. Springer, 2016.

[133] Raznav Solea, Adrian Filipescu, and Urbano Nunes. Sliding-mode control for trajectory-tracking of a wheeled mobile robot in presence of uncertainties. En *Proceedings of the 7th Asian Control Conference*, págs. 1701–1706. 2009.

[134] Eduardo D Sontag. Input to state stability: Basic concepts and results. En *Nonlinear and optimal control theory*, págs. 163–220. Springer, 2008.

[135] Tsu T Soong. *Fundamentals of probability and statistics for engineers*. John Wiley & Sons, 2004.

[136] Daniel Stonier, Se-Hyoung Cho, Sung-Lok Choi, Naveen Suresh Kuppuswamy, and Jong-Hwan Kim. Nonlinear slip dynamics for an omniwheel mobile robot platform. En *Robotics and Automation, 2007 IEEE International Conference on*, págs. 2367–2372. IEEE, 2007.

[137] Young Soo Suh. Send-on-delta sensor data transmission with a linear predictor. *Sensors*, 7(4):537–547, 2007.

[138] Paulo Tabuada. Event-triggered real-time scheduling of stabilizing control tasks. *Automatic Control, IEEE Transactions on*, 52(9):1680–1685, 2007.

[139] Yodyium Tipsuwan and Mo-Yuen Chow. Control methodologies in networked control systems. *Control engineering practice*, 11(10):1099–1111, 2003.

[140] Harry Trentelman, Anton A Stoorvogel, and Malo Hautus. *Control theory for linear systems*. Springer Science & Business Media, 2012.

[141] Sebastian Trimpe and Raffaello DÁndrea. Accelerometer-based tilt estimation of a rigid body with only rotational degrees of freedom. En *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, págs. 2630–2636. IEEE, 2010.

[142] Spyros G Tzafestas. *Introduction to mobile robot control*. Elsevier, 2013.

[143] Spyros G Tzafestas, KM Deliparaschos, and GP Moustris. Fuzzy logic path tracking control for autonomous non-holonomic mobile robots: Design of system on a chip. *Robotics and Autonomous Systems*, 58(8):1017–1027, 2010.

[144] Volodymyr Vasyutynskyy and Klaus Kabitzsch. Simple pid control algorithm adapted to deadband sampling. En *Emerging Technologies and Factory Automation, 2007. ETFA. IEEE Conference on*, págs. 932–940. IEEE, 2007.

[145] Jasmin Velagic, Bakir Lacevic, and Nedim Osmic. *Nonlinear motion control of mobile robot dynamic model*. INTECH Open Access Publisher, 2008.

[146] Manel Velasco, Josep Fuertes, and Pau Marti. The self triggered task model for real-time control systems. En *Work-in-Progress Session of the 24th IEEE Real-Time Systems Symposium (RTSS03)*, tomo 384. 2003.

[147] Keigo Watanabe, Takahiro Yamamoto, Kiyotaka Izumi, and Shoichi Maeyama. Underactuated control for nonholonomic mobile robots by using double integrator model and invariant manifold theory. En *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, págs. 2862–2867. IEEE, 2010.

[148] David K Wehe, Johann Borenstein, Liqiang Feng, and HR Everett. Mobile robot positioning: Sensors and techniques. 1997.

[149] Robert L Williams, Brian E Carter, Paolo Gallina, and Giulio Rosati. Dynamic model with slip for wheeled omnidirectional robots. *Robotics and Automation, IEEE Transactions on*, 18(3):285–293, 2002.

[150] Björn Wittenmark, Johan Nilsson, and Martin Törngren. Timing problems in real-time control systems. En *Proceedings of the American Control Conference*. Citeseer, 1995.

[151] Tao Wu and Qi Cheng. Distributed estimation over fading channels using one-bit quantization. *Wireless Communications, IEEE Transactions on*, 8(12):5779–5784, 2009.

[152] Fang Yang and Chaoli Wang. Adaptive tracking control for uncertain dynamic nonholonomic mobile robots based on visual servoing. *Journal of Control Theory and Applications*, 10(1):56–63, 2012.

[153] Tai C Yang. Networked control system: a brief survey. *IEE Proceedings Control Theory and Applications*, 153(4):403, 2006.

[154] John K Yook, Dawn M Tilbury, and Nandit R Soparkar. Trading computation for bandwidth: Reducing communication in distributed control systems using state estimators. *Control Systems Technology, IEEE Transactions on*, 10(4):503–518, 2002.

[155] Mei Yu, Long Wang, Tianguang Chu, and Fei Hao. An lmi approach to networked control systems with data packet dropout and transmission delays. En *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, tomo 4, págs. 3545–3550. IEEE, 2004.

[156] Dong Yue, Qing-Long Han, and James Lam. Network-based robust H∞ control of systems with uncertainty. *Automatica*, 41(6):999–1007, 2005.

[157] Dong Yue, Engang Tian, Yijun Zhang, and Chen Peng. Delay-distribution-dependent robust stability of uncertain systems with time-varying delay. *International Journal of Robust and Nonlinear Control*, 19(4):377–393, 2009.

[158] Li Zhang, Tao Cui, and Xianda Zhang. Distributed estimation for sensor networks with channel estimation errors. *Tsinghua Science & Technology*, 16(3):300–307, 2011.

[159] Wei Zhang, Michael S Branicky, and Stephen M Phillips. Stability of networked control systems. *Control Systems, IEEE*, 21(1):84–99, 2001.

[160] X Zhang, W Xie, SV Hoa, and R Zeng. Design and analysis of collaborative automated fiber placement machine. En *Volume1-Issue1 - 2016*, págs. 1(1): 1–14. International Journal of Advanced Robotics and Automation, 2016.

[161] Xiaomei Zhang, Yufan Zheng, and Guoping Lu. Stochastic stability of networked control systems with network-induced delay and data dropout. En *Decision and Control, 2006 45th IEEE Conference on*, págs. 5006–5011. IEEE, 2006.