Universidad de Alcalá
Escuela Politécnica Superior

Máster Universitario en Ingeniería de Telecomunicación
Especialidad en Tecnologías Espaciales y de Defensa

Trabajo Fin de Máster

Propuesta de estimación y control, remotos y aperiódicos, de unidad robótica P3-DX

Autor: Rubén Nieto Capuchino

Tutor: Felipe Espinosa Zapata

2016

UNIVERSIDAD DE ALCALÁ

ESCUELA POLITÉCNICA SUPERIOR

Máster Universitario en Ingeniería de Telecomunicación
Especialidad en Tecnologías Espaciales y de Defensa

Trabajo Fin de Máster

Propuesta de estimación y control, remotos y aperiódicos, de unidad robótica P3-DX

Autor: Rubén Nieto Capuchino

Director: Felipe Espinosa Zapata

Tribunal:

Presidente: José Luis Lázaro Galilea

Vocal 1º: Oscar Rodríguez Polo

Vocal 2º: Felipe Espinosa Zapata

Fecha: 6 de septiembre de 2016

A mis padres y hermana.

"Para alcanzar lo imposible, uno debe intentar lo absurdo."
Miguel de Cervantes

Agradecimientos

Quizás tenga que reconocer que esta parte es la que más veces habré repasado, porque no quiero olvidarme de nadie que a lo largo de estos dos últimos años han hecho posible que se acabe una nueva etapa de mi vida, pero sobre todo no quisiera olvidarme de nadie que haya aportado a este trabajo.

En primer lugar quisiera agradecer a mis padres todo su esfuerzo realizado a lo largo de todos estos años y que han hecho posible completar mi formación académica concluida (de momento) en este trabajo, que parte de él se o debo a ellos. Sin duda han sido un pilar fundamental a lo largo de tantos años y que agradeceré siempre.

Agradecer también a mi hermana Sandra su paciencia en ciertas ocasiones y por su infinita paciencia siempre que trataba de despejar la mente mientras estudiaba y trababa se saber que andaba haciendo.

No quiero olvidarme tampoco del resto de mi familia; mis abuelas, tíos, tías, primos y prima que siempre han mostrado interés por todos mis progresos y saber en qué líos estaba enredado en cada momento. Ahora ya podré decirles que ya hay un "teleco" oficialmente en la familia, para arreglar todos los ordenadores y cacharros que se han estropeando y espero que siempre pueda continuar intentando arreglar.

Tampoco debo olvidarme de las personas que han compartido mi camino a lo largo de estos años, pero en especial a mi compañera Ruth Pérez, que a lo largo de todo este tiempo hemos compartido muchas horas de laboratorio, de estudio, de frustración, de apoyo mutuo, de ánimos, de alegrías y de momentos únicos. Espero que a partir de ahora nuestros caminos puedan continuar juntos durante mucho tiempo.

Estos dos últimos años han sido muy importantes para mi, pero en parte se debe a todas las personas que he conocido. Hablo de mis compañeros del Fondo 33; David, Álvaro y Pinedo, de los compañeros del OL3; los Pablos y Carlos y de Diego, con los que he compartido ideas o distintos aspectos en los ratos de descanso mientras disfrutábamos de un maravilloso café. Pero en especial agradecer a Miguel y Carlos, quienes me han aconsejado, me han ayudado y me han enseñado todos los fundamentos que encierran sus trabajos y que forman parte de este trabajo.

Me gustaría también dedicar unas palabras en primer lugar al director de este proyecto, Felipe Espinosa Zapata que un día creyó en mí para realizar este trabajo y que su inestimable ayuda ha sido fundamental. Para mi ha sido un placer trabajar con él y agradeceré siempre todo lo que he aprendido de él, así como todo lo que me queda por aprender.

Otra de las personas que me ha ayudado a adquirir los conocimientos básicos de para poder usar las cámaras kinect 2, así como las bases en las que se sustenta la detección del robot necesario para este proyecto es Alfredo Gardel. Con el que ha sido un placer conocer dentro y fuera del aula, trabajando codo con codo para probar, reprobar y comprobar que las cámaras del demostrador funcionaban y realizaban la detección correctamente. Espero que el futuro nos depare poder seguir trabajando durante unos cuantos años más.

Agradecer también a Julio Pastor por ofrecerme una beca que me ha permitido aprender, gestionar y disfrutar de muchas de las actividades que realizábamos a distintos centros educativos de educación secundaria, que pese a los quebraderos de cabeza que nos pudieran llevar, en el momento de la verdad disfrutaba más que los asistentes.

Seguramente cuando cierre este capítulo, se me haya olvidado incluir en esta parte y poder agradecer de algún modo lo que han significado, no sólo para la realización de este proyecto, sino por hacer posible el cierre de esta etapa.

A todos, gracias.

Resumen

En los sistemas de control en red se dan casos en los que los distintos nodos consumen recursos compartidos, pero no aportan impacto al control de la planta. En ocasiones, los distintos sensores o las plantas a controlar hacen uso del canal de comunicaciones sin influir apenas en el comportamiento de la planta controlada.

El objetivo principal de este trabajo es evaluar las ventajas del control basado en eventos frente al basado en tiempo (periódico) cuando se dispone de un centro remoto (mini-PC Intel NUC), encargado de las labores de control y estimación de estados, comunicado de forma inalámbrica con la planta a controlar y un sensor externo a la planta. Proponiendo así los mecanismos de disparo que permitan al centro remoto mandar información a un robot P3-DX de forma eficiente, pero también entre centro remoto y Microsoft Kinect 2, de forma que esta solo envíe medida cuando el estimador de estados basado en eventos (EBSE) crea que es necesario en base la covarianza del error en la estimación.

De esta forma se pretende obtener un uso más eficiente del canal de comunicación frente a la solución periódica, sin degradar de forma significativa el control del robot.

Palabras clave: event-based control, event-triggered control, Lyapunov based controller (LBC), seguimiento de trayectorias, estimador de estados basado en eventos (EBSE).

Abstract

In the network control systems in cases where different nodes consume shared resources, but do not provide impact control of the plant are given. Sometimes, different sensors or control plants make use of the communication channel without hardly influence in the behavior of the plant.

The main target of this work is to evaluate the advantages of event-based control versus time-based (periódico) when it has a remote center (mini-PC Intel NUC), responsible for carrying out control and state estimation, statement wirelessly with the plant control and an external sensor to the plant. Thus proposing triggered mechanisms that allow the remote center to send information to a P3-DX robot efficiently, but also between remote center and Microsoft Kinect 2 so it only send as when the state estimator based on events (EBSE) believes is necessary based on the error covariance estimation.

This is the way to obtain a more efficient channel of communication against the periodic solution without significantly degrading the robot control use.

Keywords: event-based control, event-triggered control, Lyapunov based controller (LBC), path tracking, event-based state estimation (EBSE).

Índice general

Índice de figuras

Índice de tablas

Parte I

Memoria

Capítulo 1

Introducción

Este capítulo introductorio trata de describir el contexto donde se desarrolla este Trabajo Fin de Máster (TFM). Se realizará una breve descripción del proyecto ALCOR, el cual va a marca los objetivos de este trabajo, así como la descripción de las unidades robóticas utilizadas. Además, se enumerarán las motivaciones del propio TFM.

1.1 Proyecto ALCOR

El proyecto ALCOR está vinculado dentro de la actividad del grupo de investigación GEINTRA (Grupo de Ingeniería Electrónica Aplicada a Espacios Inteligentes y Transporte) del Departamento de Electrónica de la Universidad de Alcalá. Las líneas de investigación abordadas como indica en sus siglas comprenden un conjunto de áreas de trabajo relacionadas con el uso de la tecnología electrónica en los espacios inteligentes y los sistemas de transporte.

Se puede definir [1] un entorno inteligente como el espacio físico dónde la tecnología de la información, así como diferentes tecnologías de computación se fusionan para lograr alcanzar objetivos específicos para un usuario, el entorno o ambos. Bajo esta definición para poder realizar la fusión entre el espacio físico y los distintos elementos que proporcionan la inteligencia al entorno es necesario que exista una capa de sensores y actuadores entre ambos para recoger información del entorno, así como poder realizar una determinada acción sobre el mismo.

En la figura 1.1 se muestra un esquema donde se representa el mundo físico que se describía anteriormente. Los sensores pueden ser simples transductores o bien ser dispositivos que tengan cierta capacidad de proceso y/o comunicación. La inteligencia puede estar distribuida en diferentes sistemas empotrados o centralizada en un ordenador o servidor. Por último destacar que la red de comunicaciones que une los distintos elementos puede ser cableada, inalámbrica o híbrida.

Entre los objetivos del proyecto ALCOR se encuentra la optimización de una red de sensores inalámbricos y sistemas de control en red para la cooperación de unidades móviles en entornos inteligentes. Para ello, el reto está en optimizar los recursos de cada una de las unidades móviles (UM) y de los módulos sensoriales (MS), y también de los compartidos.

Con lo anterior expuesto, este trabajo se centra en la aplicación de una solución de control y estimación basada en eventos, y ejecutada en el centro remoto, para el control de una unidad móvil, utilizando una cámara para actualizar la informaición de la pose del móvil.Dentro del desarrollo del proyecto ALCOR.

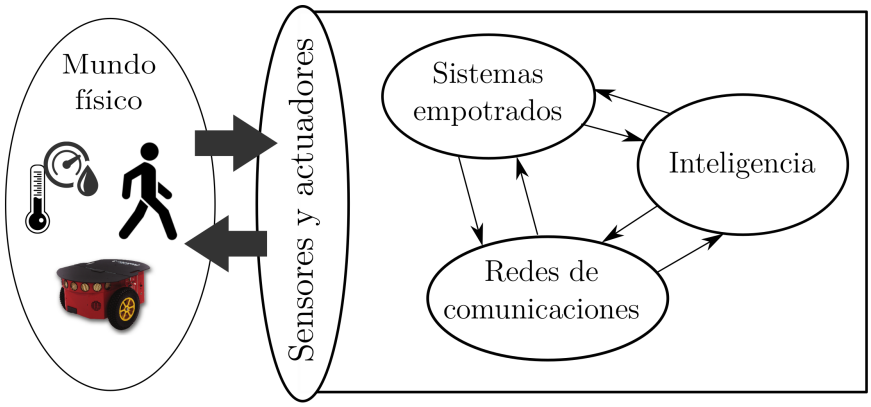

Figura 1.1: Esquema entorno inteligente.

De las alternativas de control basado en eventos se ha elegido la de auto-disparo (self-triggered), y de las existentes para estimación basada en eventos se ha optado por la que se implementa en el controlador (centro remoto) a partir del cálculo de la covarianza del error de estimación. Finalmente se propondrá una propuesta de fusión de control y estimación basada en eventos. La figura 1.2 resume los retos planteados en el proyecto.

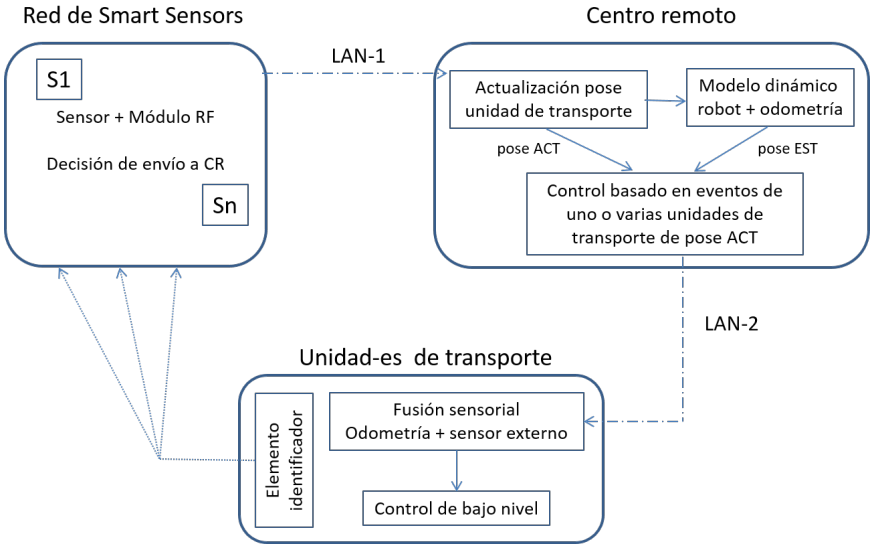

Figura 1.2: Esquema resumen de los retos planteados en el proyecto.

1.2 Objetivos

El objetivo principal de este TFM es disponer de un demostrador que permita evaluar las ventajas de control y estimación basados en eventos, y por tanto no periódicos, aplicado al seguimiento remoto de trayectorias de un robot P3-DX. Tanto el sensor (cámara) como el robot están conectadas en red con el centro remoto de procesamiento (estimación y control).

La principal razón de evaluar las ventajas del control basado en eventos es que en un sistema de control clásico, existe un periodo de muestreo que marca la actualización de las tareas. Cuando el sensado y el control es remoto, estas tareas consumen recursos del enlace de comunicación de forma periódica sin apenas producirse cambios en el lazo de control. Es decir, utilizan la red de comunicación sin que mejore el comportamiento de la planta controlada, lo que supone un ineficiente uso de los recursos compartidos.

Figura 1.3: Prototipo de robot utilizado en el trabajo.

Para mitigar esta situación se propone una técnica de intercambio de información entre la unidad móvil, módulo sensorial y centro remoto sólo cuando sea necesario.

Por tanto, los objetivos concretos de este trabajo son los siguientes:

- Propuesta de mecanismo de disparo de los eventos producidos en el lazo de control.
- Propuesta de mecanismo de disparo para la estimación (corrección de la pose) a partir de las medidas recogidas por un módulo sensorial externo al robot, que en este caso es una cámara.
- Validación mediante simulación de la estrategia comparándola con una solución periódica.
- Validación de la propuesta mediante un demostrador de laboratorio utilizando un robot P3DX, una cámara Kinect 2.0 y un PC como controlador, todos ellos conectados a través de una red WiFi.

1.3 Motivación

Aprovechando la experiencia del grupo de investigación GEINTRA en el control remoto periódico de unidades móviles capaces de realizar taras individuales y/o cooperativas, se pretende avanzar en esta línea dejando de lado la solución de control periódica y adoptando una solución de control aperiódica que permita reducir el uso del canal de forma eficiente sin que afecte a la planta a controlar.

Además, se pretende realimentar el lazo de control con la pose del robot obtenida a través de una cámara Kinect 2.0 mediante el uso de visión artificial. Pero el sensor sólo enviará medida cuando el centro remoto la precise, realizando una petición a través del canal de comunicación.

Las unidades móviles a controlar son robots P3DX, los cuales incorporan una placa base Via-Epia (mini-ITX), conectada de forma inalámbrica a través de un convertidor Ethernet-Wireless. Todo ello ejecuta un sistema operativo de tiempo real Ubuntu kernel 2.6.23 con RTAI. En la figura 1.3 se muestra una imagen del prototipo del robot utilizado en este TFM.

El centro remoto encargado de las labores de control, hasta el momento de plantear este TFM, se ejecutaban en un ordenador con mejores características que las de la Via-Epia con un sistema operativo

de tiempo real Ubuntu con RTAI. Se decidió estudiar la posibilidad de migrar la versión de Ubuntu, que estaba en la 10.04-LTS a una más actual garantizado que el kernel está parcheado con RTAI, con el fin de actualizar en parte la infraestructura desplegada hasta el momento.

Como la actualización era posible, se decidió realizar la actualización en un mini PC Intel NUC con un procesador a 2.1GHz y 4GB de memoria RAM. Instalando la versión de Ubuntu a la 12.04-LTS kernel 3.4-9-rtai, todo el proceso de instalación está recogido en el apéndice B.

Capítulo 2

Revisión del estado del arte

2.1 Introducción

En este capítulo se va a situar al lector dentro del marco teórico que engloba este trabajo.

Se comenzará este capitulo desarrollando el estado del Arte de los trabajos que se han desarrollado o que se están desarrollando relacionados con el objetivo de este TFM.

Se continuará con los apartados más teóricos relacionados con los fundamentos de control no lineal basado en eventos (sección A.1), así como los fundamentos de estimación basada en eventos (sección 2.3). Por último, centrados más en el caso del trabajo, se realizará un ligero repaso a la parte de las comunicaciones inalámbricas que se llevan a cabo entre el centro remoto y las unidades móviles (sección 2.4).

2.2 Revisión control basado en eventos

Este capítulo es un resumen del capítulo Introducción de la tesis doctoral de Carlos Santos [2].

Las soluciones de control tradicionales por lo general ignoran los aspectos de implementación en la fase de diseño considerando estas fases independientes. Sin embargo, debido a la proliferación actual de los sistemas de control empotrados y en red (NCS), el criterio de implementación cobra especial relevancia ya que en estos sistemas los recursos suelen ser compartidos por múltiples tareas de control. Por lo tanto, la optimización de los recursos computacionales y de comunicación es de suma importancia en estos sistemas. Esta problemática justifica la aparición de las técnicas aperiódicas, destacando entre ellas las técnicas *Event-triggered* control (ETC) [3–5] y *Self-triggered* control (STC) [6–8].

2.2.1 Consideraciones sobre el periodo de muestreo en los sistemas de control

Actualmente la mayor parte de los controladores son implementados en plataformas digitales,por lo tanto las tareas del controlador sólo pueden ser ejecutadas en intervalos de tiempo discretos.En estos casos, si se sigue el planteamiento clásico, se hace necesario fijar un periodo constante de muestreo (Ts). Este periodo se fija basándose en la dinámica del sistema a controlar, indicando cada cuanto tiempo se realizan actualizaciones en el controlador y se obtienen medidas de los distintos sensores, véase figura 2.1. Durante cada periodo de muestreo las señales en el controlador permanecen constantes con técnicas zero-order

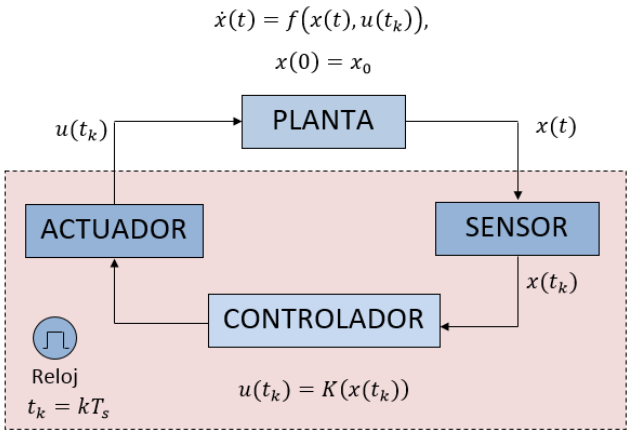

Figura 2.1: Sistema de control digital con muestreo periódico.

hold. Este tipo de estrategia de control es denominada como time-scheduled, debido a que la planificación se realiza de forma periódica en el tiempo.

El gran inconveniente del control time-scheduled [9] es que el periodo de muestreo tiene que fijarse para garantizar la estabilidad del sistema en el peor de los casos, lo que conlleva que esta estrategia de control resulte conservadora desde el punto de vista de número de actualizaciones del controlador. Resulta obvio, que el reducir el número de actuaciones de control es ventajoso, siempre que se garantice un índice de comportamiento adecuado en el sistema realimentado, ya que de esta forma se reduce el consumo de energía y el uso de procesador. Además la reducción de actuaciones de control en los sistemas en red, reduce el tráfico de datos entre nodos con las ventajas asociadas que ello supone.

Por ello, frente al planteamiento conservador de time-scheduled se han desarrollado nuevas técnicas en las que el periodo de muestreo no es constante sino que se adapta a la sucesión de eventos (event-scheduled) que modifican el estado del sistema controlado, optimizando de esta forma el uso de los distintos recursos. Estas novedosas técnicas de control son: control activado por eventos (*Event-triggered control* -ETC-) y control autoactivado (*Self-triggered control* -STC-).

La idea básica del ETC es que los eventos del sistema son disparados cuando la medida de una señal de error sobrepasa un cierto nivel previamente ajustado. La estrategia STC puede considerarse una extensión natural de la anterior, pero en lugar de requerir un sensado continuo del estado de la planta se realiza una estimación de cuándo ha de producirse el siguiente evento. La solución STC evita la actividad continua de los sensores pero no se tiene información de la planta entre evento y evento.

En este TFM se ha optado por utilizar la técnica ETC.

2.2.1.1 Aspectos clave de la técnica de control *Self-triggered*

Los sistemas STC se caracterizan por calcular en cada instante de actualización, tanto las leyesde control como el siguiente instante de muestreo (Figura 2.2).

Este tiempo es calculado en función del estado actual de la planta y la dinámica de la misma, siendo normalmente el tiempo entre eventos mayor que el periodo de muestreo de las implementaciones periódicas. Además presenta otra gran ventaja: entre muestra y muestra no hay actividad de sensado. En el caso del STC sólo se toman medidas en los instantes de actualización de la ley de control, pudiendo en este caso tener los sensores en modo de ahorro de energía hasta que llegue el siguiente evento. Sin embargo esto, a su vez, puede suponer un problema pues el sistema de control trabaja en lazo abierto

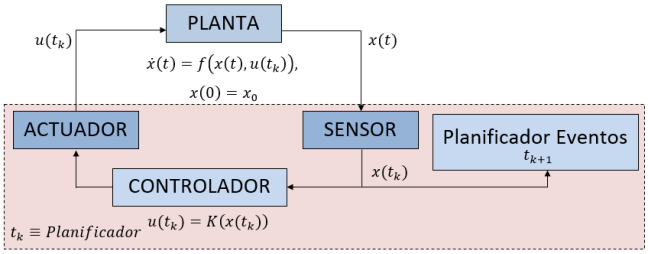

Figura 2.2: Estructura de sistema de control *Self-triggered*.

entre muestras, ya que no se dispone de información de la planta, con lo que se reduce la robustez frente a perturbaciones externas.

Uno de los primeros trabajos presentados en el que se describe de forma simple y clara esta técnica de control es [10]. En él se diseña un controlador PID disparado por eventos, demostrando a través de numerosas simulaciones cómo es posible reducir de forma significativa la tasa de uso de la CPU con una degradación mínima del comportamiento del controlador. El mayor problema encontrado en este trabajo, es la dificultad de verificar y garantizar la estabilidad del controlador, ya que no se disponía de una teoría de sistemas basados en eventos lo suficientemente amplia como para realizar un análisis detallado del sistema de control resultante, como se hizo posteriormente en [7, 11].

2.2.1.2 Aspectos clave de la técnica de control *Event-triggered*

En los sistemas ETC es el acontecimiento de un evento, en lugar del cumplimiento de una condición temporal, el que fija cuándo debe ser realizada la actualización del controlador (Figura 2.3).

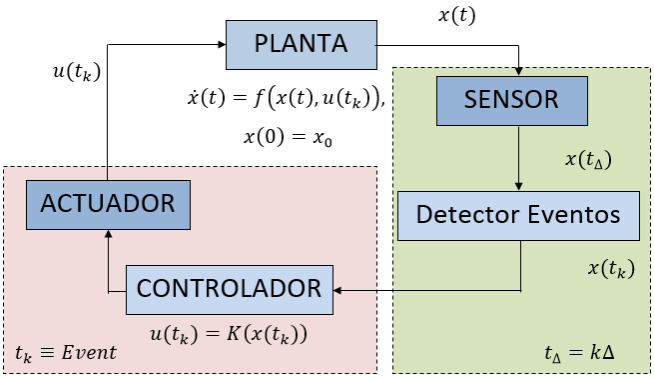

Figura 2.3: Estructura de sistema de control *Event-triggered*.

La naturaleza del evento puede variar, en general se suele considerar el instante en el que la variable sensada $x(t)$ difiere de la última enviada $x(t_{k-1})$ por encima de un umbral. La naturaleza del muestreo (Δ) en el sistema ETC puede ser intrínseca tanto al método de medida utilizado, como a la naturaleza física del sistema a controlar.

2.3 Fundamentos de estimación basada en eventos

Hasta ahora se ha centrado el estudio en la parte de control, donde se ha definido un controlador basado en Lyapunov, que a incluye un servosistema de velocidad lineal y angular de la unidad robótica, dónde se

incluía un estimador basado en Kalman, pero se utilizaba para realimentar aquellos estados que no eran medibles.

Centrando el estudio en los trabajos publicados en la linea de estimación basada en eventos, con la finalidad de estimar la posición de un robot en interiores, se encuentra el trabajo [12], donde se muestra la aplicación de la corrección de la posición de un robot a través de un Filtro de Kalman Extendido (EKF) el cual recibe la información de una cámara conectada a través del centro remoto y comunica a través del robot usando la tecnología Bluetooth.

El controlador basado en Lyapunov, tiene el proposito de realimentar con la pose del robot, donde esta pose del robot va a venir dada por un sensor externo, que en este caso particular es una cámara pero puede ser otro sensor cualquiera, como por ejemplo un sensor láser como se demuestra su uso de aplicación en [13]

Estos sensores externos, dependiendo de la tecnología usada, no siempre pueden dar una medida cuando lo precise el lazo de control. Por tanto, una de las razones de que se va a utilizar un estimador es para que proporcione una estimación al lazo de control cuando no se tenga una medida del sensor.

Un aspecto importante a tener en cuenta, es que cuando se tiene una red de sensores distribuidos, ya sea una red cableada o inalámbrica (WSN), es interesante como método de ahorro de energía y en el canal de comunicación, que los sensores envíen información cuando se supere un cierto umbral. Dependiendo dónde se sitúe el mecanismo de decisión de aportación de información a la red, ya sea en el sensor o en un centro remoto, se aplicará una determinada estrategia.

El problema de la estimación de la pose para el estudio que se está llevando a cabo se trata de detectar un patrón de cuatro círculos, uno de ellos con una tonalidad más clara, que se dispone encima del robot. La detección del patrón se va a realizar en toda imagen que recoge la cámara.

La pose del robot $[x_{rob}\ y_{rob}\ \theta_{rob}]^T$, donde x_{rob} e y_{rob} son las coordenadas de la posición y θ_{rob} es la orientación del robot en ángulo, se determina mediante procesamiento de imagen y técnicas de visión artificial a partir del reconocimiento del patrón anteriormente mencionado, como se recogen en [14–19]. Por tanto, la pose del robot cuando no se tenga medida será lo que se debe obtener en la salida del estimador y realimentar así el lazo de control.

2.3.1 Estimador basado en Unscented Kalman Filter (UKF)

El filtro de Kalman como estimador de estados periódico en sistemas con ruido de estado y ruido de medida, es idóneo para esta aplicación.

La cinemática que describe el movimiento del robot, dadas en tiempo discreto son:

$$\begin{cases} x(t+T) = x(t) + Tv(t)\cos\left(\theta(t) + \dfrac{T}{2}\omega(t)\right) \\ y(t+T) = y(t) + Tv(t)\sin\left(\theta(t) + \dfrac{T}{2}\omega(t)\right) \\ \theta(t+T) = \theta(t) + T\omega(t) \end{cases} \tag{2.1}$$

Donde x e y representan las coordenadas de la posición, θ la orientación en ángulo, v es la velocidad lienal y ω es la velocidad angular.

Dado el siguiente vector de estados:

$$\mathbf{x} = \begin{bmatrix} x & y & \theta \end{bmatrix}^{\mathrm{T}} \tag{2.2}$$

y su correspondiente vector de entrada:

$$\mathbf{u} = \begin{bmatrix} v & \omega \end{bmatrix}^{\mathrm{T}} \tag{2.3}$$

Sea $f(\mathbf{x}, \mathbf{u})$ la función vector que describe las expresiones 2.1. Por tanto, el modelo del sistema del robot en forma vectorial es de la forma:

$$\mathbf{x}((n+1)T) = f(\mathbf{x}(nT), \mathbf{u}(nT)) + \mathbf{w}(nT) \tag{2.4}$$

Donde $n \in \mathbb{N}$ y $\mathbf{w}$ representa el ruido Gaussiano aditivo debido a las perturbaciones externas, cuya covarianza de ruido se define:

$$E[\mathbf{w}(nT)\mathbf{w}^{\mathrm{T}}(nT)] = \mathbf{Q}_{\mathbf{w}} \tag{2.5}$$

En cuanto a las entradas del sistema se va a añadir un ruido a las variables de entrada, ya que sólo se conocen los comandos de velocidad que proporciona el centro remoto $\mathbf{u}_c = \begin{bmatrix} v_c & \omega_c \end{bmatrix}^T$, ya que la velocidad del controlador $\mathbf{u}_c$ y la velocidad que lleva realmente la unidad móvil $\mathbf{u}$ no es la misma debido a perturbaciones externas. Por tanto, la variables de entrada se van a definir como:

$$\begin{aligned} v &= v_c + \Delta v \\ \omega &= \omega_c + \Delta\omega \end{aligned} \tag{2.6}$$

El cual presenta una matrix de covarianza que viene dada:

$$E\Big[\begin{bmatrix}\Delta v & \Delta\omega\end{bmatrix}^{\mathrm{T}}\begin{bmatrix}\Delta v & \Delta\omega\end{bmatrix}\Big] = \mathbf{\Sigma}_{\mathbf{u}} \tag{2.7}$$

Otra fuente de ruido que entra en juego en este análisis es la debida a la medida del sensor, en este caso la cámara. Cuando el sensor detecta la pose del robot $\mathbf{y}_k$ la medida no está exenta de ruido $\mathbf{v}_k$. El subíndice k que acompaña a las variables denota que la medida se ha tomado en el instante $t_k = n_k T$.

$$\mathbf{y}_k = \mathbf{x}_k + \mathbf{v}_k \tag{2.8}$$

Se asume que el ruido es Gaussiano e incorrelado de forma que la matriz de covarianza de define:

$$E[\mathbf{v}_k \mathbf{v}_k^{\mathrm{T}}] = \mathbf{R}_k \tag{2.9}$$

2.3.1.1 Filtro de Kalman Unscented (UKF)

La predicción de la estimación está basada en un Unscented Kalman Filter (UKF) [20], una de las razones de que se utilice un filtro de Kalman UKF, es porque el filtro de Kalman sólo es óptimo para sistemas lineales los cuales se ven perturbados por ruido blanco Gaussiano, en cambio para sistemas no lineales existen variantes del filtro de Kalman, como el filtro de Kalman extendido (EKF) y el Unscented Kalman Filter (UKF). Aunque el coste computacional es mayor, este filtro proporciona una mejor aproximación de la matriz de corvarianza del error de estimación de estados $\mathbf{P}$ que el EKF.

Este estimador calcula el vector de estimación $\widehat{\mathbf{x}}$, así como la aproximación de la matriz $\mathbf{P}$, cada periodo $t = nT$.

$$\mathbf{P} \approx E[(\widehat{\mathbf{x}} - \mathbf{x})(\widehat{\mathbf{x}} - \mathbf{x})^{\mathrm{T}}] = \begin{bmatrix} \sigma_x^2 & \sigma_{x,y} & \sigma_{x,\theta} \\ \sigma_{y,x} & \sigma_y^2 & \sigma_{y,\theta} \\ \sigma_{\theta,x} & \sigma_{\theta,y} & \sigma_\theta^2 \end{bmatrix} \tag{2.10}$$

Se va a definir e_x y e_y como el error de estimación para las coordenadas x e y. Sea $P_{i,j}$ la i-ésima fila y la j-ésima columna de la matriz $\mathbf{P}$. Entonces

$$\begin{aligned} P_{1,1} &= E\big[(x - \widehat{x})^2\big] = E[e_x^2] \\ P_{2,2} &= E\big[(y - \widehat{y})^2\big] = E[e_y^2] \\ P_{1,2} = P_{2,1} &= E\big[(x - \widehat{x})(y - \widehat{y})\big] = E[e_x e_y] \end{aligned} \tag{2.11}$$

En el estimador basado en el filtro de Kalman hay dos etapas diferenciadas. Una etapa de predicción dónde se tiene en cuenta el modelo de la planta, así como las entradas para permitir conocer el avance del punto a lo largo del tiempo. La etapa de corrección es la encarga decorregir la estimación del estado a partir de la información obtenida por la medida del sensor. A continuación se explican con mayor detalle estas dos etapas.

2.3.1.1.1 Etapa de predicción

En esta etapa, el vector de estados estimado $\hat{\mathbf{x}}$ debe ser propagado a través de las ecuaciones que describen la cinemática del móvil 2.1. Esta etapa de predicción se ejecutará periódicamente cada T_s, ya que esta parte no requiere de ningún tipo de información que provenga de algún sensor.

Por cada periodo de tiempo T_s, se genera un conjunto de puntos alrededor del punto estimado actual $\hat{\mathbf{x}}$, distribuidos de acuerdo con $\mathbf{P}$, a dicho conjunto de puntos se le va a denominar sigma-points. Los sigma-points representan puntos del estado estimado actual distribuido con una función de densidad de probabilidad propia del estado.

$$\hat{\mathbf{x}}_a = \hat{\mathbf{x}}, \quad \mathbf{P}_a = \mathbf{P} \tag{2.12}$$

Se crearán un total de $2N$ sigma-points $\hat{\mathbf{x}}_a^{(i)}$ que serán calculadas con las fórmulas, donde $N = 3$, debido al número de estados del vector de estados:

$$\begin{aligned} \hat{\mathbf{x}}_a^{(i)}(t) &= \hat{\mathbf{x}}_a(t) + \left(\sqrt{N\mathbf{P}_a(t)}\right)_i^{\mathrm{T}} \quad i = 1, 2, \ldots, N \\ \hat{\mathbf{x}}_a^{(N+i)}(t) &= \hat{\mathbf{x}}_a(t) - \left(\sqrt{N\mathbf{P}_a(t)}\right)_i^{\mathrm{T}} \quad i = 1, 2, \ldots, N \end{aligned} \tag{2.13}$$

De la expresión anterior, $(\sqrt{N\mathbf{P}_a})_i$ es la i-ésima fila de la descomposición de Cholesky de la matriz $N\mathbf{P}_a$. Para la evolución temporal de los sigma points se aplica la función f_d del sistema discreto:

$$\begin{aligned} \hat{\mathbf{x}}^{(i)}(t+T) &= f_d\big(\hat{\mathbf{x}}_a^{(i)}(t), \mathbf{u}_c(t)\big) \\ \hat{\mathbf{x}}(t+T) &= \frac{1}{2N} \sum_{i=1}^{2N} \hat{\mathbf{x}}_a^{(i)}(t+T) \end{aligned} \tag{2.14}$$

Por último, la matriz de covarianza $\mathbf{P}$ se obtiene como la matriz cruzada de los sigma-points transformados:

$$\mathbf{P} = \frac{1}{2N}\sum_{i=1}^{2N}(\widehat{\mathbf{x}}^{(i)} - \mathbf{x})(\widehat{\mathbf{x}}^{(i)} - \mathbf{x})^{\mathrm{T}} \tag{2.15}$$

2.3.1.1.2 Etapa de corrección

En la etapa de corrección no es necesario aplicar la tranformación Unscented propia del UKF. Por tanto la actualización de la medida se va a calcular mediante un filtro de Kalman asíncrono (AKF), que se rige con las siguientes ecuaciones:

$$\widehat{\mathbf{x}}_k^+ = \widehat{\mathbf{x}}_k^- + \mathbf{L}_k(\mathbf{y}_k - \mathbf{H}\widehat{\mathbf{x}}_k^-) \tag{2.16}$$

donde $\widehat{\mathbf{x}}_k^-$ representa la estimación a priori en el instante t_k y $\widehat{\mathbf{x}}_k^+$ la estimación a posteriori en el instante t_k.

El algoritmo del filtro de Kalman está orientada para minimizar la covarianza del error de estimación a posteriori encontrando el valor óptimo de L_k para cada actualización de la medida. El valor de L_k se calcula como:

$$\mathbf{L}_k = \mathbf{P}_k^-(\mathbf{P}_k^- + \mathbf{R}_k)^{-1} \tag{2.17}$$

El resultado de la matriz de covarianza a posteriori es:

$$\mathbf{P}_k^+ = (\mathbf{I}_N - \mathbf{L}_k)\mathbf{P}_k^- \tag{2.18}$$

donde $\mathbf{I}_N$ es la matriz identidad N-ésima. $\mathbf{P}_k^-$ y $\mathbf{P}_k^+$ representan las matrices de covarianza del error a priori y a posteriori respectivamente.

Se puede deducir de las ecuaciones 2.17 y 2.18, que la etapa de corrección ayuda a reducir la traza de la matriz $\mathbf{P}$, y la cantidad de reducción de la traza depende de la precisión de la medida, la cual viene dada por la matriz de covarianza del ruido $\mathbf{R}_k$.

2.3.2 Estimador de estados basado en eventos

De las alternativa posibles [21], en este TFM se utiliza la técnica cuyo mecanismo de disparo (decisión de aportar información actualizada del sensor) se implementa en el centro remoto.

En los sistemas no lineales con valores de ruido variables, la covarianza del error de estimación $\mathbf{P}$ no converge a una solución periódica, como ocurre en sistemas lineales. El aplicar un estimador de estados basado en eventos tiene como finalidad de mantener el nivel de covarianza de ruido por delimitado y además, se puede aplicar para que sólo se tome una muestra de medida cuando se necesite, que en el caso de redes de sensores permite un uso del canal más eficiente.

Los intervalos de tiempo en los que se pida medida a los sensores, va a depender del comportamiento de la matriz de covarianza, sobre todo la forma de crecimiento, así como del valor inicial. El crecimiento de la matriz de covarianza $\mathbf{P}$ es independiente de las señales sensadas. En la etapa de prediccción, $\mathbf{P}$ crece hasta un determinado umbral, generándose un evento que actualiza la información de salida de la planta y se reduce el valor de $\mathbf{P}$.

El algoritmo de estimador de estados basado en eventos propuesto se indica en el diagrama de flujo mostrado en la figura 2.4 (Referencia: [22]).

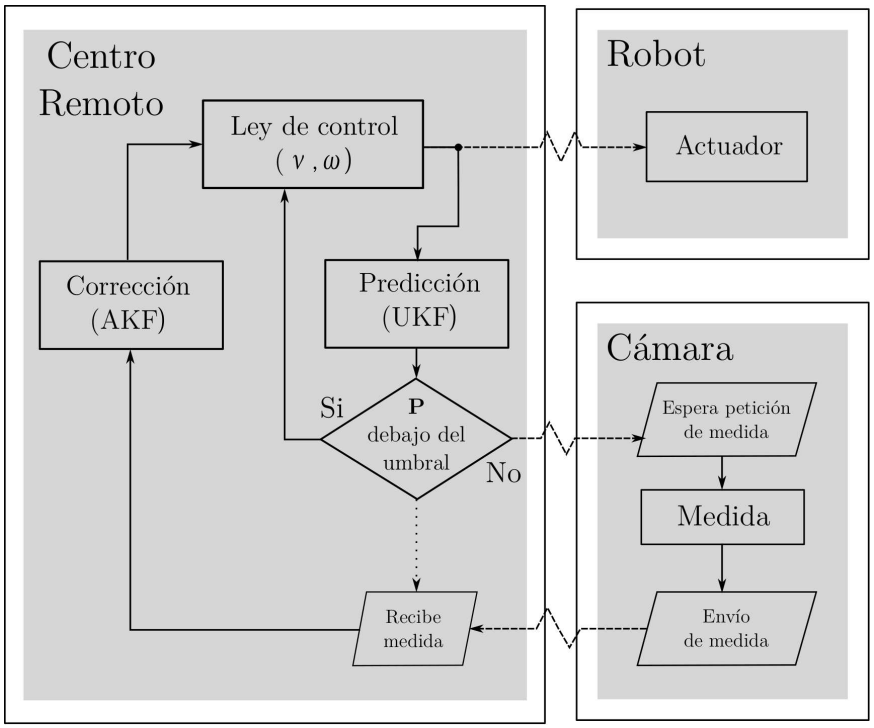

Figura 2.4: Diagrama de flujo del estimador basado en eventos.

Con la pose estimada y la trayectoria de referencia, el módulo de control calcula las consignas de velocidad que se van a enviar al robot. Con dichas consignas de velocidad lineal y angular el módulo estimador puede predecir la ubicación del vehículo pasado un periodo T_s, así como la covarianza de error. Si la covarianza del error de predicción se mantiene por debajo de un cierto umbral, es posible calcular las consignas de velocidad, así como la pose del robot con incertidumbre aceptable sin requerir medida actualizada del sensor.

A continuación de muestran dos formas de determinar el umbral, las cuales se puede complementar.

2.3.2.1 Umbral en base al error de distancia y ángulo

El umbral en base al error de distancia y ángulo, trata de conocer el error de localización que se tiene respecto a la ubicación real. Pero es complicado operar con los errores, ya que son no lineales, al tratarse de dos variables aleatorias correlacionadas. Por esta razón, trabajar con el error cuadrático medio simplifica el procesamiento.

Sea e_d^2 el error de distancia al cuadrado, definido como:

$$e_d^2 = e_x^2 + e_y^2 \tag{2.19}$$

El valor medio de e_d^2 se puede obtener como:

$$E[e_d^2] = E[e_x^2 + e_y^2] = E[e_x^2] + E[e_y^2] = P_{1,1} + P_{2,2} \tag{2.20}$$

La raíz cuadrada de dicho valor se conoce como la raíz del error cuadrático medido de la distancia, usado como indicador de la precisión de la localización.

$$\text{DRMS} = \sqrt{E[e_d^2]} = \sqrt{P_{1,1} + P_{2,2}} \tag{2.21}$$

Por supuesto, hay que tener en cuenta el error de estimación del ángulo θ. Una estimación precisa de θ es fundamental para el cálculo de la orientación en el algoritmo de control. Por tanto, por sí misma se trata de un umbral de disparo para segurar así que la incertidumbre de la orientación siempre sea lo suficientemente pequeña. Combinando tanto el error de distancia como el de orientación, el umbral de disparo para solicitar una medida al sensor es la siguiente:

$$(P_{1,1} + P_{2,2} > D_{thr}^2) \vee (P_{3,3} > \tilde{\theta}_{thr}^2) \tag{2.22}$$

Donde D_{thr} y $\tilde{\theta}_{thr}$ son los umbrales que deberán ser ajustados por el diseñador de acuerdo a las necesidades.

Para una mejor adaptación en circunstancias cambiantes, en la siguiente sección se propone un umbral adaptativo dónde se tiene en cuenta la distancia del móvil respecto al punto de referencia.

2.3.2.2 Umbral adaptativo en base al error de distancia

En el apartado anterior, se ha visto como el umbral estaba definido por el error en distancia que existía, junto con el error de ángulo. Pero si el móvil se encuentra a una distancia elevada del punto de referencia la condición de disparo se estaría cumpliendo siempre, pero esto no es útil desde el punto de vista de consumo de recursos, ya que hasta que el móvil no se encuentre próximo al punto de referencia no sería necesario realizar un seguimiento de la trayectoria.

Por tanto, hay dos etapas diferenciadas, por un lado la de aproximación al punto de referencia y por otro la etapa de seguimiento de trayectoria. En la primera etapa, basta con tener una referencia de dónde se sitúa el robot, por tanto el umbral de disparo para pedir una medida será mayor, mientras que la segunda se requiere que este umbral sea lo suficientemente pequeño para realizar el seguimiento.

Por tanto, el nuevo umbral va a estar definido por la distancia del móvil al punto de referencia (e_d) que se va a ajustar con una constante K_D, de esta forma se podrá mantener mejor una relación entre el umbral de disparo y la distancia al punto de referencia.

$$D_{thr} \approx e_d K_D = D_{appr} \tag{2.23}$$

Este nuevo umbral va a tender a cero, lo que hace que cuando el móvil esté cerca de la referencia se produzcan disparos para pedir medida continuamente. Para evitar esto, con el umbral obtenido en 2.3.2.1, se va a sumar al umbral obtenido en este punto. De forma que el límite inferior del umbral de disparo adaptativo, se verá fijado por el umbral de seguimiento de la trayectoria (D_{trk}), que será fijado dependiendo de los requerimientos de diseño.

$$D_{thr} = \sqrt{D_{trk}^2 + D_{appr}^2} \tag{2.24}$$

En la figura 2.5 se muestra la gráfica que ilustra lo que trata de obtener con el umbral de disparo adaptativo.

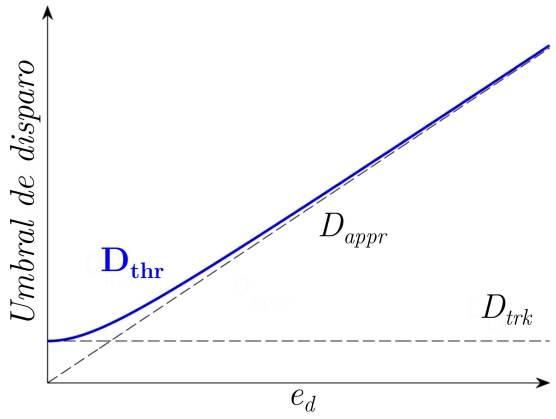

Figura 2.5: Umbral de disparo adaptativo.

2.4 Sistemas de control en red, redes de sensores y comunicaciones inalámbricas

Pasando a la parte de las redes de sensores distribuidos, que en este caso al tratarse de una red inalámbrica se van a denominar WSN, de sus siglas en inglés Wireless Sensor Network. En [23], se hace un repaso de todos los aspectos fundamentales que tiene una WSN, como son los protocolos de comunicación utilizado en la conexión inalámbrica, así como los tipos de sensores y su posible aplicación.

En este caso, la red inalámbrica de sensores se trata de cámaras que procesan la imagen con el fin de encontrar a la unidad móvil en la imagen y aportar su pose (coordenadas x e y junto con la orientación θ) al centro remoto. Al igual que ocurría anteriormente con las comunicaciones el envío periódico de medidas hace que se sobrecargue la red. Pero además, se tiene otro inconveniente debido al tiempo de procesamiento de la imagen siendo este no periódico.

Se recuerda que el objetivo de posicionar los robots en entornos interiores, los sistemas sensoriales son cruciales para determinar la posición de las distintas unidades móviles, así lo demuestran distintos autores [24], [25].

Otro aspecto fundamental es la precisión que posea la tecnología utilizada, así como la mejora que se pueda realizar tras un postprocesado de la medida, son objeto de estudio por los distintos grupos de investigadores, [26].

Pero la tendencia hace que la investigación se posicione en la interacción del entorno con las unidades móviles, especialmente en el ámbito de los entornos inteligentes [27].

Independientemente de cómo se ubiquen la red de sensores ya sea en la infraestructura [28], así como que utilicen otros sensores para la localización [29] o una combinación de ambos [30], el tratamiento de las medidas, así como de las tareas de control se asume de forma periódica.

Generalmente, para la determinación de la pose de una unidad móvil, utilizan algoritmos de estimación basado en el Filtro de Kalman los cuales tienen una fase de predicción, que se ejecuta de forma periódica y otra fase de corrección que utiliza la información proporcionada por los sensores cuando esté disponible. Esta aperiodicidad viene impuesta bien por la tecnología o bien por la estrategia de disponer información solo cuando esta sea relevante. Además, esto obliga en parte a decidir cuándo y cuánta información se aporta por parte del sensor, ya que el envío masivo de información por parte de la unidad sensorial reduce la autonomía energética del mismo y contribuye a que su consumo sea poco eficiente, un aspecto a tener en cuenta para aquellos casos que los sensores estén alimentados por baterías.

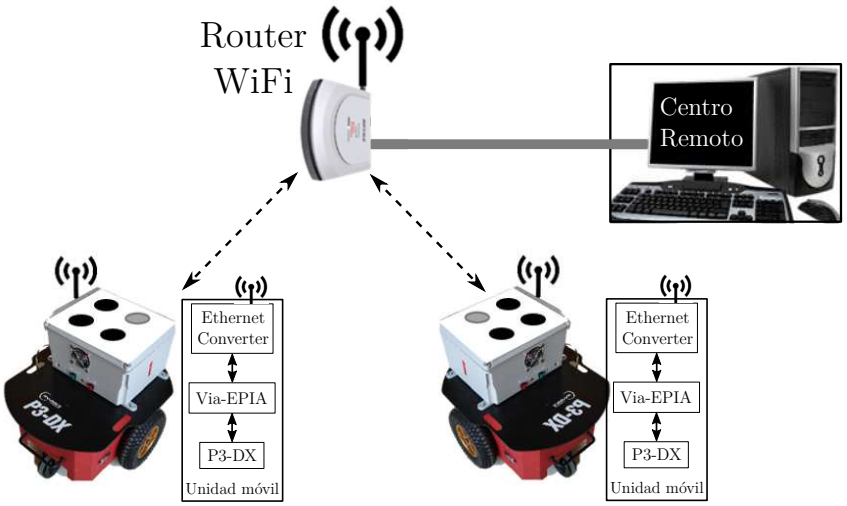

Figura 2.6: Esquema de la arquitectura hardware de comunicaciones.

Para llevar a cabo la comunicación inalámbrica entre el centro remoto y las unidades móviles, cada robot lleva incorporado un convertidor Wirellesss (Ethernet Convertes de 125Mbps [31] que se conecta a la Via-Epia [32] mediante un enlace Ethernet.

Cuando se inicia la aplicación, cada una de las unidades móviles arranca el sistema operativo que lleva embarcado cada robot, los cuales se conectan automáticamente al router inalámbrico donde se encuentra conectado el centro remoto, formando así una red privada. En la figura 2.6 se muestra un esquema gráfico de la arquitectura hardware descrita.

Entrando más a fondo dentro de esta arquitectura de comunicaciones, podemos decir que en trabajos previos realizados dentro del proyecto ALCOR, la arquitectura desarrollada se basaba en el uso de Player/Stage, a partir del estándar de comunicación 802.11 (Wi-Fi) y haciendo uso del protocolo de comunicación TCP.

Esta arquitectura presentaba algún inconveniente, por un lado la comunicación cliente-servidor era transparente para el usuario de Player, haciendo imposible implementar algún método de compensación de pérdida de paquetes. Por otro lado, el uso del protocolo de comunicación basado en TCP no es apropiado para este tipo de aplicaciones de control con comunicaciones inalámbricas, ya que las retransmisiones de información antigua no resulta útil, como se demuestra en [33].

Como solución a estos dos inconvenientes en [34] se realizó el diseño de una arquitectura de comunicaciones basada en el protocolo UDP, sobre el estándar 802.11 (WiFi). El uso del protocolo de la capa de transporte UDP que no está orientado a conexión, hace que no requiera establecer una conexión previamente, además, el problema de las retransmisiones no se da, ya que no tiene confirmación ni control de flujo, por lo que los paquetes pueden adelantarse unos a otro, o que no lleguen correctamente. Estos dos últimos inconveniente relacionados con el orden de los paquetes o la integridad de los mismos, no son de mucha importancia ya que el enlace se establece dentro de una red privada. Por tanto, UDP se basa en el intercambio de datagramas donde se incorpora la información necesaria de direccionamiento en la cabecera del propio datagrama.

La programación de la red de comunicaciones hace uso de sockets y aplicaciones cliente-servidor [35]. Las aplicaciones cliente-servidor consisten, en distribuir la aplicación de comunicación en dos partes. Una parte definida por el proceso llamado cliente, que emite las peticiones; y la otra, el servidor, que está escuchando las peticiones y las atiende. Este tipo de aplicaciones tiene una serie de ventajas que se enumeran a continuación.

- Transparencia entre tipo de equipo y sistema operativo. Los procesos cliente-servidor son capaces de comunicarse con independencia de máquina o tipo de sistema operativo.
- Fiabilidad, ya que puede haber redundancia de servidores y en caso de que falle uno, los demás podrían estar preparado para entrar en funcionamiento en cualquier momento.
- Flexibilidad para añadir tantos cliente y servidores como se desee.

Cada aplicación cliente-servidor trabaja con un canal de comunicación, al descriptor de ese canal se le denomina socket. A través del socket se indica el protocolo de comunicación a usar, en este caso UDP, la dirección de red IP local y remota y el número de puerto local y remoto. Así es posible lograr identificar a cada aplicación cliente-servidor de forma única.

En la figura 2.7 se muestra el esquema de funcionamiento de la aplicación cliente-servidor utilizada trabajando con el protocolo UDP. En el caso de estudio, el cliente enviará datos al servidor correspondiente, el cual los almacenará y procesará para esperar los siguientes datos.

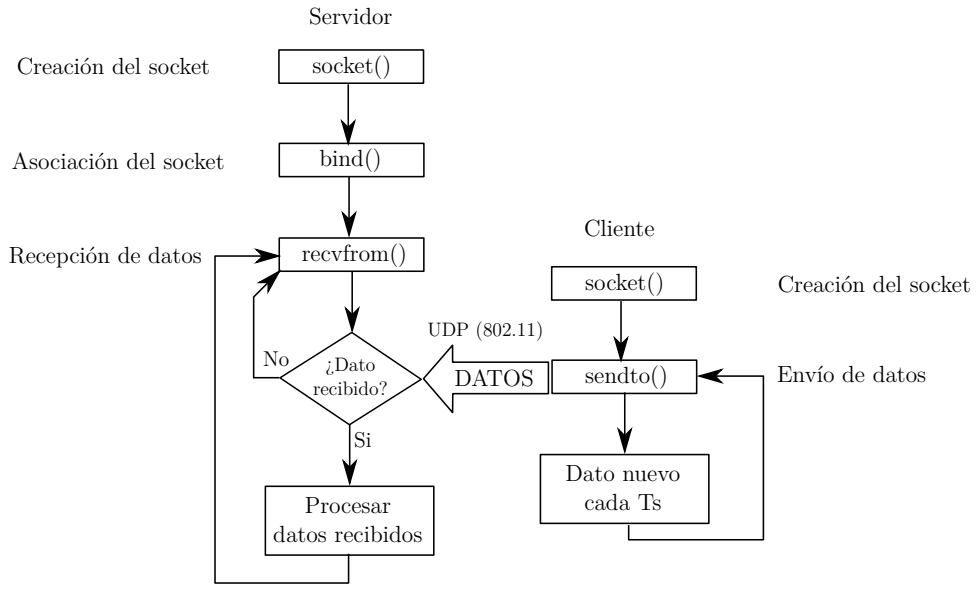

Figura 2.7: Diagrama de flujo de las comunicaciones entre cliente y servidor.

2.5 Conclusiones

Como se ha podido ver a lo largo del capítulo, además de proporcionar una visión actual de las investigaciones que tienen relación, se ha introducido la parte teórica que sustenta el trabajo actual. Llegado a este punto, se va a continuar explicando el desarrollo que se ha llevado a cabo a lo largo de este trabajo.

Capítulo 3

Desarrollo

3.1 Introducción

En este capítulo se quiere poner en contraste las ventajas o inconvenientes que presentan las técnicas de control o sensado aperiódico con las técnicas tradicionales periódicas.

1. Comparación entre soluciones aperiódica y periódica de control remoto.
2. Comparación entre soluciones de estimación basada en eventos y periódica, admitiendo un control periódico.
3. Propuesta de fusión de control y sensado aperiódicos, y comparación con la solución clásica (control y sensado periódicos).

El primero de ellos se va a centrar en explicar la implementación llevada a cabo de la propuesta de control aperiódico, haciendo una comparación con la propuesta clásica de control periódico. El segundo de los apartados, se centrará en obtener las mejoras que presenta una estimación basada en eventos. El último de los apartados comprenderá el conjunto de los dos anteriores, ya que se tratará de fusionar la parte de control aperiódico junto con la de estimación basada en eventos, de esta forma se verá el comportamiento de ambas técnicas en conjunto.

3.2 Comparación entre soluciones aperiódica y periódica de control remoto

En esta sección se va a realizar la formalización de un control *event-triggered*, basándose en la estructura que está implementada actualmente, que se ha mostrado en la sección A.1. Se va a determinar los mecanismos de disparo que van a permitir realizar el seguimiento de trayectorias, así como la aproximación a un punto, validando su funcionamiento con resultados obtenidos de la simulación.

La estrategia *event-triggered* determina cuando es necesario actualizar la ley de control para que el comportamiento de la unidad móvil se corresponda con el predeterminado. Para determinar cuando es necesario actualizar la ley de control, el controlador *event-triggered* utiliza el modelo del sistema, así como la última medida obtenida para evaluar si es necesario actualizar. De esta manera, el controlador basado en *event-triggered* genera la trayectoria que relaciona el estado actual con la última medida obtenida.

En el capítulo 5 de [2] se demuestra la estabilidad de este tipo de controladores basado en *event-triggered*. Como este controlador está basado en el controlador de Lyapunov presentado en la sección A.1.3, se va a centrar en las condiciones de disparo para la aproximación a un punto, así como el seguimiento de trayectoria.

3.2.1 Control *event-triggered* enfocado a la aproximación de un punto

Se va a considerar que el error de orientación es más crítico que el error de distancia, ya que el robot debe permanecer orientado en todo momento para reducir el error de distancia. Por tanto, se va a desglosar la función de Lyapunov, que recordamos a continuación.

$$V = \frac{1}{2}d^2 + \frac{1}{2}\alpha^2 \tag{3.1}$$

$$V = V_1 + V_2 \tag{3.2}$$

donde:

$$V_1 = \frac{1}{2}d^2; \quad V_2 = \frac{1}{2}\alpha^2 \tag{3.3}$$

V_1 y V_2 representan el error de distancia y el error de ángulo respectivamente a un determinado punto. De esta forma, es posible analizar cada error de forma independiente.

Se va a definir como el instante de actualización t_k, determinando entonces la condición de disparo presentada en la ecuación 3.4. Básicamente la condición de disparo trata que la función V_1 o V_2 sea decreciente hasta el próximo disparo, hasta alcanzar de forma asintótica el valor de cero.

$$t_{k+1} = min\big(t > t_k \mid \dot{V}_1(d(t)) \geq 0 \vee \dot{V}_2(\alpha(t)) \geq 0\big) \tag{3.4}$$

Esta condición de disparo permite que tanto el error de distancia (d) como el error de orientación (θ) que converjan de forma asintótica a una zona próxima al punto de equilibrio.

3.2.1.1 Resultado de la simulación

En este caso se trata de alcanzar un punto con una determinada orientación, partiendo de otro alejado con distinta orientación. En este punto se va a tratar de comparar la estrategia aperiódica, con la periódica de aproximación al punto.

Para la parte periódica, basta con implementar el controlador explicado en A.1.3.3, la cual se ejecutará con la periodicidad impuesta, que en este caso es de 0,01 segundos.

Eso mismo junto con la evaluación de la condición de disparo explicada anteriormente, obtenemos el controlador basado en la estrategia aperiódica.

En el ejemplo simulado, las constantes utilizadas son $K_{v1} = 0{,}2$ y $K_{\omega 1} = 0{,}2$, donde cada una de las constantes corresponde a la velocidad de respuesta ante los errores de distancia y orientación respectivamente.

La posición de inicio y orientación de la unidad móvil se encuentra en $(-5,\ 3,\ -\frac{\pi}{2}\ )$, mientras que el pose de objetivo se encuentra en (0,0,0). En la figura 3.1 se muestra la trayectoria que sigue el robot hasta alcanzar el objetivo, se muestra con un círculo los puntos donde se produce una actualización de las consignas de velocidad lineal y angular. Como se puede apreciar, la trayectoria realiza una curva en

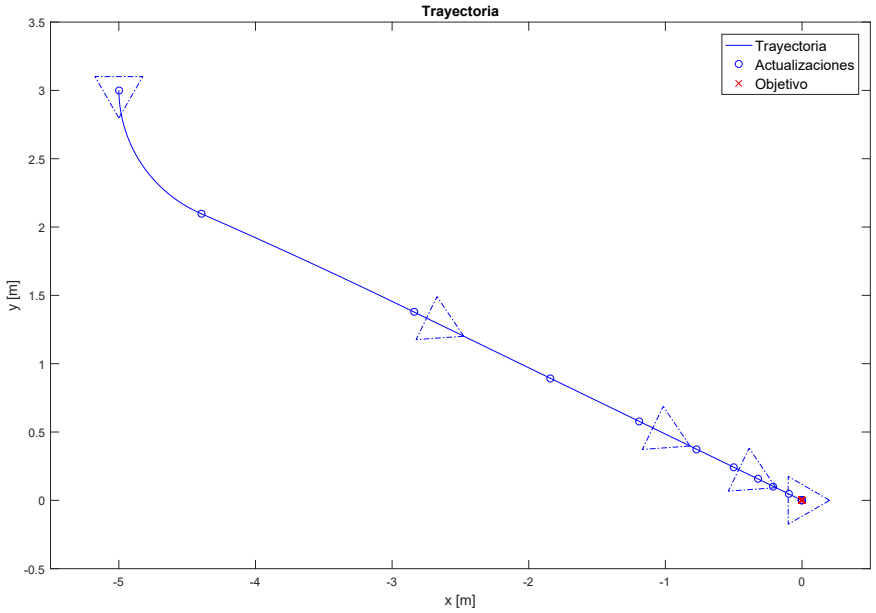

Figura 3.1: Ejemplo de trayectoria con la estrategia de aproximación al punto junto con las actualizaciones de consigna.

sentido hacia el origen de coordenadas (destino de la trayectoria) y luego continúa prácticamente recto hasta alcanzar el objetivo, que finalmente alcanza la orientación que se ha impuesto.

En la figura 3.2 se muestra las consignas de velocidad lineal y angular que lleva la unidad móvil a lo largo de la simulación. Se muestra el momento dónde se produce una actualización en la consigna por parte del controlador.

Para analizar la función de Lyapunov, en la figura 3.3 muestra cómo evoluciona a lo largo del tiempo. Se puede observar cómo dicha función tiende a cero, ya que como se muestra en la ecuación 3.1 está compuesto por el error de distancia y el error de ángulo. Además se muestran los momentos de actualización de la consigna, los cuales se generan a partir de la condición de disparo.

Si comparamos esta solución, con una solución periódica, en la figura 3.4 se puede comprobar cómo la función de Lyapunov (V) de la propuesta aperiódica converge a cero de forma más lenta que se encuentra la mayor parte del tiempo por debajo de la función de Lyapunov de la solución periódica.

3.2.2 Control *event-triggered* enfocado al seguimiento de una trayectoria

En este caso, la condición de disparo que se debe establecer tiene que cumplir las correspondientes condiciones descritas en la sección anterior A.1.3.3.2, dónde se tiene como objetivo el realizar el seguimiento de una trayectoria.

La condición de disparo no pretende asegurar la estabilidad asintótica, sino garantizar una zona acotada que incluya un punto de equilibrio, de forma que se mantenga en dicha zona hasta que se designe una nueva consigna. Por tanto, el siguiente instante de actualización t_k viene determinado por:

$$t_{k+1} = min\left(t > t_k \mid \left(\left(\dot{V}(t) \geq 0\right) \wedge \left(V(t_k) > V_0\right)\right)\right) \tag{3.5}$$

Al igual que en el apartado anterior (3.2.1) se mantenía las funciones V_1 y V_2 decreciente, con la condición 3.5 obliga a la función de Lyapunov a ser decreciente y además mantenerla por debajo de un umbral acotado denominado V_0. De esta forma, el controlador tiene una respuesta más rápida siempre y

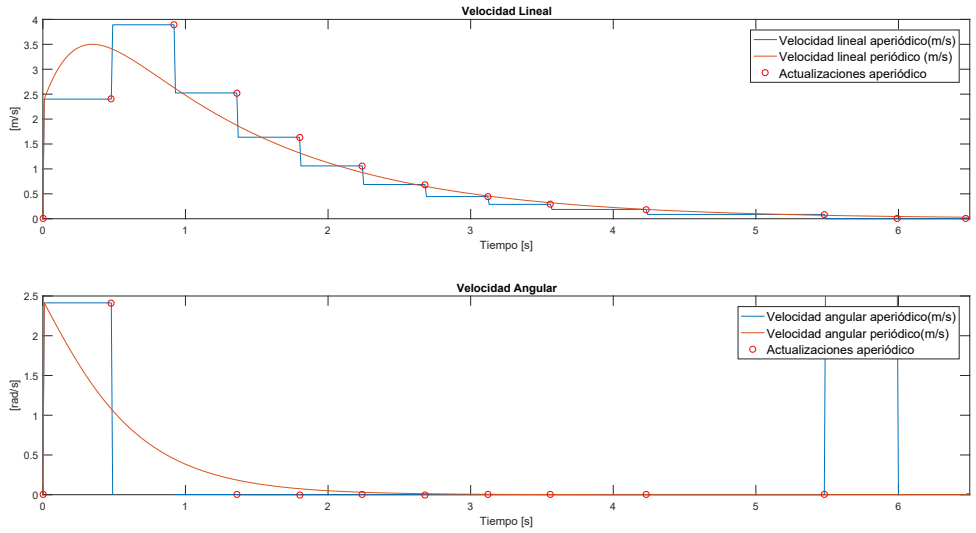

Figura 3.2: Consignas de velocidad lineal y angular obtenidas para el ejemplo.

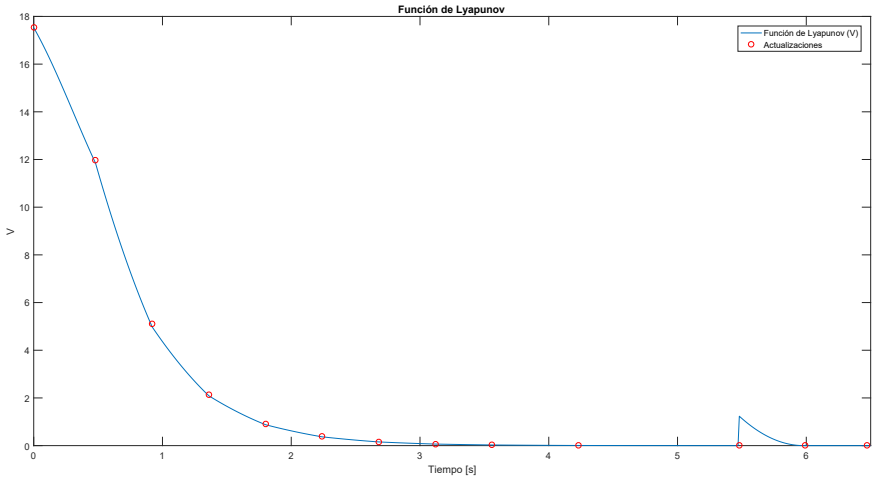

Figura 3.3: Función de Lyapunov V.

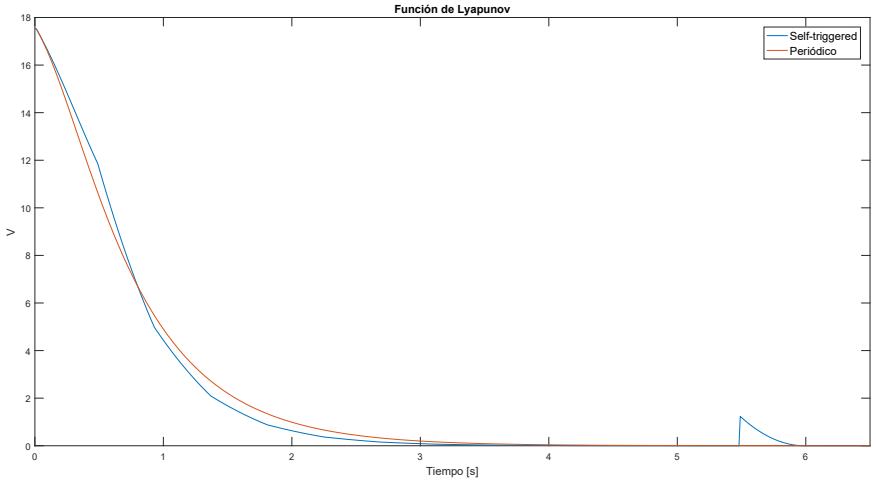

Figura 3.4: Comparación entre la función V de Lyapunov obtenida de forma periódica y de forma aperiódica.

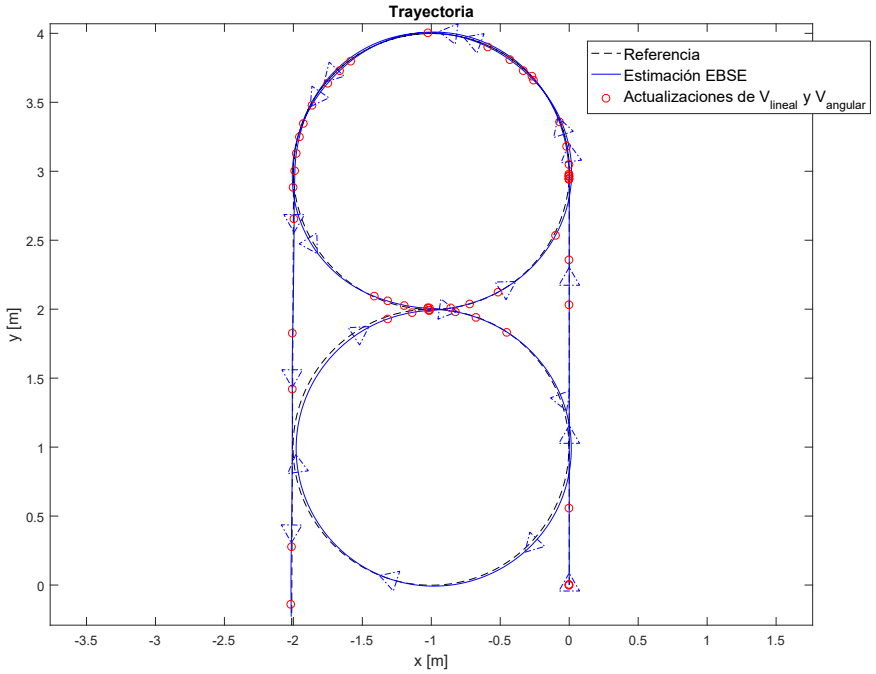

Figura 3.5: Ejemplo de trayectoria realizada con la estrategia de seguimiento de trayectoria junto con las actualizaciones.

cuando el robot se encuentre más lejos del punto deseado de la trayectoria, mientras que cuando el robot esté próximo a dicho punto, la función de disparo quedará acotada por un determinado umbral V_0.

Esto provoca que cuando se produzcan cambios en la velocidad lineal o angular de referencia (v_r, ω_r), se produzcan mayor número de disparos, favoreciendo que el robot se aproxime a la trayectoria de referencia.

3.2.2.1 Resultado de la simulación

En esta sección se va a comprobar la correcta implementación de la estrategia de seguimiento de trayectoria explicada en A.1.3.3.2. En este caso se ha generado una trayectoria en forma de ocho de referencia, la cual se compone de un tramo recto de unos tres metros y continúa realizando un trazado en forma de ocho y finaliza con un nuevo tramo recto. En la figura 3.5 se muestra en trazo rojo la trayectoria de referencia. El punto de inicio de la trayectoria de referencia, así como el punto de inicio de la unidad móvil se encuentran situados en la misma posición, que en este caso es el origen.

Las constantes utilizadas en la simulación que se exponen en el apartado A.1.3.3.2, se han fijado a $K_{v2} = 0{,}4$ y $K_{\omega 2} = 1$. El resultado de la simulación se muestra en la figura 3.5, dónde se muestra el punto inicial de la trayectoria en la parte inferior derecha, momento donde se produce una actualización en las consignas de velocidad lineal y angular. En las siguientes marcar azules se producen las siguientes actualizaciones.

Se puede comprobar el error de distancia y orientación en la figura 3.6 comparada con respecto de la pose de referencia.

Para evaluar el sistema de control se va a utilizar el Error Cuadrático Integral, ISE de sus siglas en inglés (Integrated Square Error) ecuación 3.6. En este caso la ISE obtenida es casi despreciable de 0.0017. En cuanto al número de actualizaciones que se producen a lo largo de la trayectoria es de 46, en una trayectoria que dura aproximadamente 90 segundos. Si se utilizara la tradicional solución periódica, dado

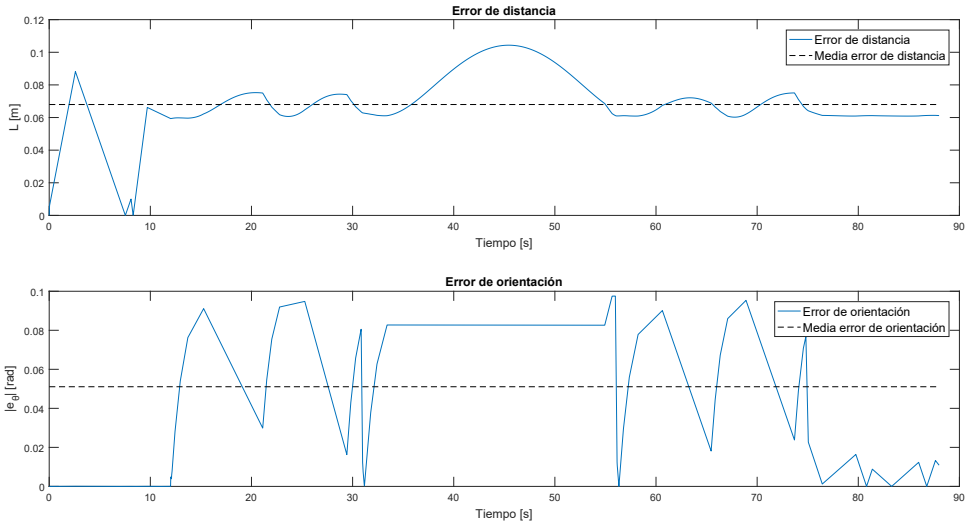

Figura 3.6: Error de distancia y orientación obtenidos con la estrategia de seguimiento de trayectoria junto con las actualizaciones.

que el periodo que está impuesto actualmente es de 10ms, el número de actualizaciones que se llevarían a cabo es 9000, lo que conlleva un ahorro considerable del número de accesos al canal de comunicaciones.

$$ISE = \sum_{k=0}^{\infty} |y(k\Delta) - y_{ref}(k\Delta)|^2 \Delta \tag{3.6}$$

Si se observa la velocidad lineal y angular que se han obtenido, tanto la de referencia como la del controlador, en la figura 3.7 se ve como la variación es mínima entre la de referencia y la obtenida por el controlador.

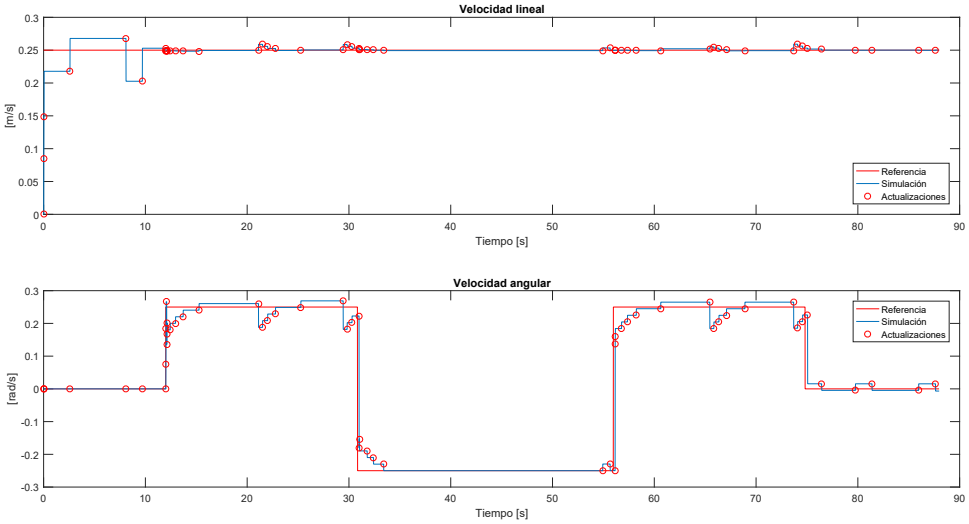

Figura 3.7: Consignas de velocidad lineal y angular obtenidas para el ejemplo.

En este caso, como se ha mencionado antes, tanto la trayectoria como la unidad móvil se encuentran en el mismo punto inicial, por tanto la función de Luapunov (V) está siempre acotada (ver figura 3.8). En este caso, lo que se obtiene es que la función de Lyapunov está acotada en una región.

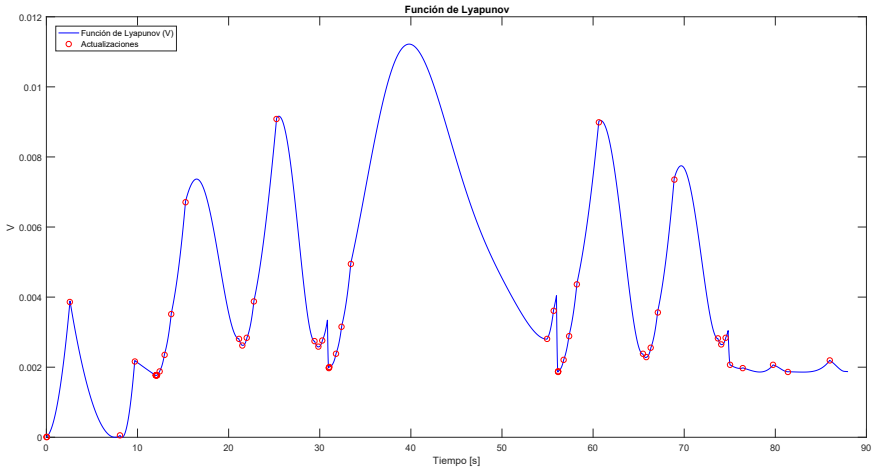

Figura 3.8: Función de Lyapunov V.

En la tabla 3.1 se recogen los resultados más relevantes de la simulación, los cuales van a permitir poder comparar con el resto de simulaciones realizadas en los siguientes apartados.

Parámetro	Control Apereriódico.
ISE	0.77086
Nº de actualizaciones periódico	9000
Nº de actualizaciones aperiódico	54
Med. Error Distancia [m]	0.06796
Med. Error Orientación [rad]	0.05107

Tabla 3.1: Tabla resumen de resultados de la simulación control *event-triggered*.

3.3 Comparación entre soluciones de estimación basada en eventos y periódica, admitiendo un control periódico

En esta sección se va a implementar la parte explicada en el estudio teórico desarrollado 2.3. En este caso el controlador que se va a utilizar es el LBC de pose ejecutado periódicamente, por lo que no se va a tener en cuenta las condiciones de disparo citadas en la sección anterior reservada al control aperiódico.

Se ha tratado de llevar a cabo una simulación que permitiera ofrecer resultados lo más fieles a la realidad, de forma que se ha generado un código que permite realizar la simulación del sensor que se va a utilizar, en este caso una cámara. La simulación de dicho sensor incluye el ruido de medida propio del sensor, así como los parámetros característicos (ver tabla 3.2) de la cámara, como la resolución, distancia focal, frames que es capaz de procesar, posición y orientación entre otros.

Para poder tener una visión más clara de la mejora que pueda suponer en realizar una estimación basada en eventos, se va a realizar la simulación periódica, para poder comparar posteriormente con el umbral adaptativo explicado en 2.3.2.2.

Parámetro	Valor
Resolución	1920 x 1080 [px]
Distancia focal	524 [px]
Imágenes por segundo (FPS)	3.5
Posición (x,y,z)	(0, 2000, 3450) [mm]
Orient. Horizontal (Pan)	$\pi/8$ [rad.]
Orient. Vertical (Tilt)	0 [rad.]
Orient. Longitudinal (Roll)	$\pi/2$ [rad.]

Tabla 3.2: Parámetros de la cámara utilizados.

3.3.1 Resultado de la simulación

En cada simulación se ha simulado bajo los mismo parámetros de configuración del sensor. Los parámetros más característicos que se han aplicado se muestran en la tabla 3.2. Cabe destacar que el número de imágenes por segundo (FPS) que puede dar la cámara es de 25 FPS, pero como esas imágenes es necesario procesarlas esta tasa se ve reducida hasta los 3.5 FPS.

Em cuanto al controlador, las constantes usadas en este caso se han fijado a $K_{v2} = 0{,}8$ y $K_{\omega 2} = 2$, que son los valores utilizados posteriormente en la implementación real que se realizará en el demostrador desarrollado.

3.3.1.1 Simulación periódica

Lo que se va a tratar en este apartado son los resultados obtenidos de la simulación para comprobar el correcto funcionamiento del sistema. La trayectoria elegida en este caso, es la misma utilizada en la sección 3.2.2.

El esquema que se va a seguir es el mostrado en la figura anterior, que se muestra en 3.9. De esta forma se va proceder a realizar una comprobación de la matriz de covarianza del error de estimación $\mathbf{P}$, pero se va a forzar que siempre esté por encima del umbral. Para ello se ha puesto un umbral que no es posible alcanzar nunca, como -1. De esta forma cada periodo está realizando una predicción, tomando una medida y aplicando la corrección, para actualizar las consignas de velocidad lineal y angular.

En la figura 3.10 se muestra el resultado de la trayectoria realizada. En la figura se muestra con un punto negro la posición de la cámara. En un tono verde se muestra el campo de visión (FOV) de la cámara, este va a depender de la orientación, y altura que esté situada la cámara.

En cuanto a los resultados obtenidos, se puede comprobar que siempre que se produce una medida de la cámara la estimación realiza la corrección pertinente y se actualizan las consignas de velocidad lineal y angular. Se puede apreciar cómo al comienzo de la trayectoria la estimación de la orientación parece no se correcta, debido al ruido de medida y provoca que el móvil se desvíe ligeramente de la trayectoria durante el tramo recto inicial, posteriormente realiza el seguimiento de la trayectoria correctamente. En la figura 3.12 se muestra el error de distancia y orientación obtenidas en la simulación con respecto a la pose de referencia, el valor de.

Por último, en la figura 3.11 se muestra la evolución del error en distancia (DRMS) y el error de orientación. Se puede apreciar en la DRMS como en este caso al no tener un umbral está periódicamente realizando medidas y corrigiendo lo que provoca que la evolución tenga forma de diente de sierra conforme

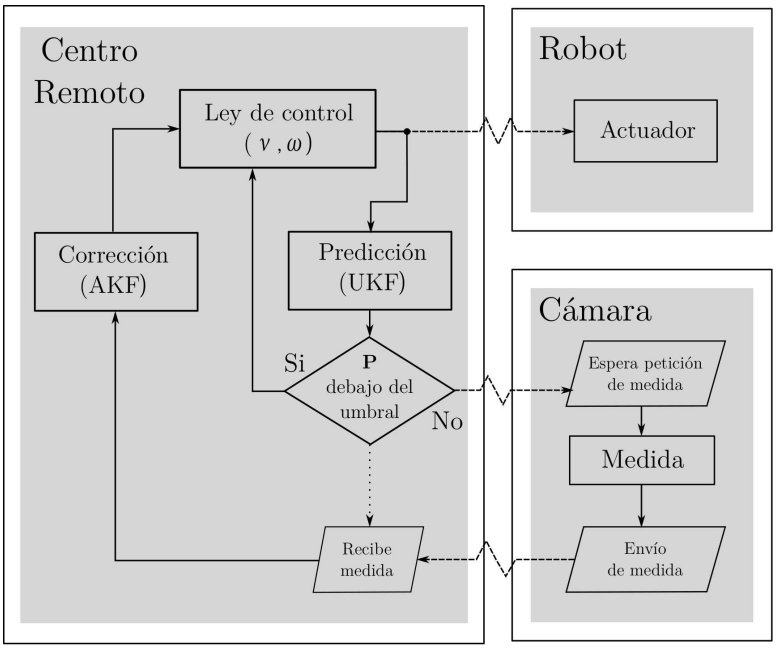

Figura 3.9: Diagrama de flujo del estimador basado en eventos.

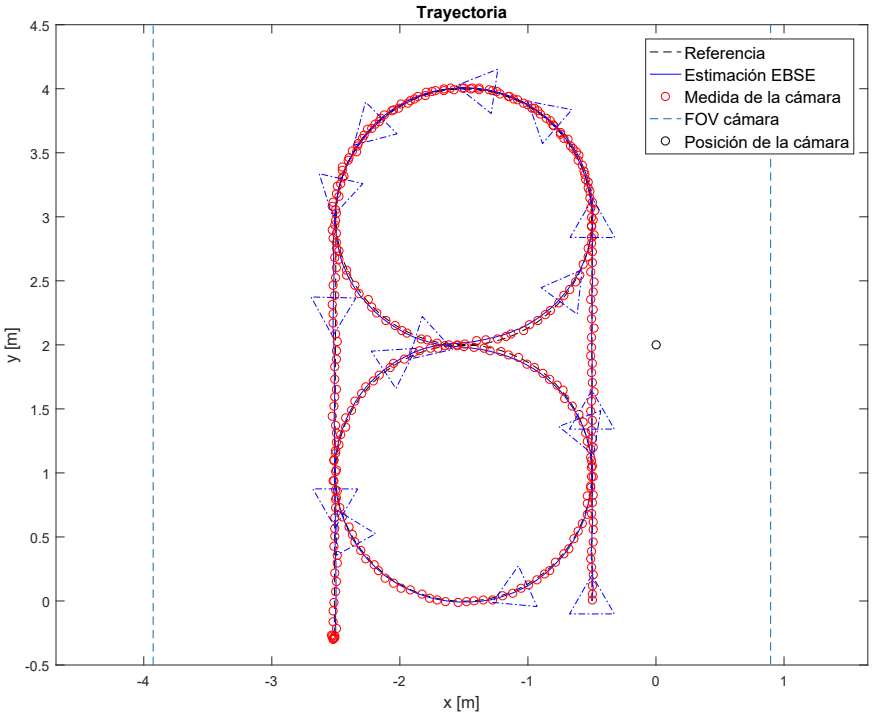

Figura 3.10: Resultado de la trayectoria realizada en la simulación de la estimación basada en eventos de forma periódica.

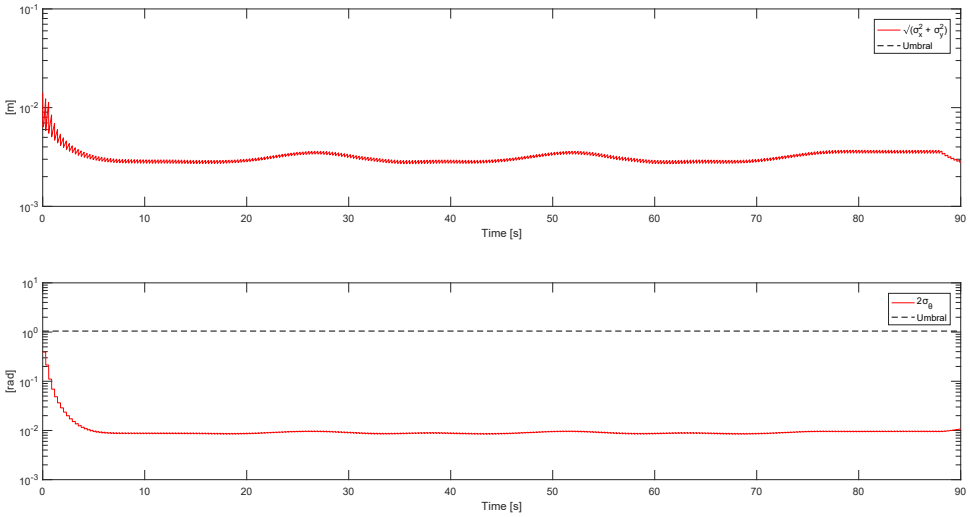

Figura 3.11: Evolución de la DRMS (superior) y el error de orientación (inferior) a lo largo del tiempo.

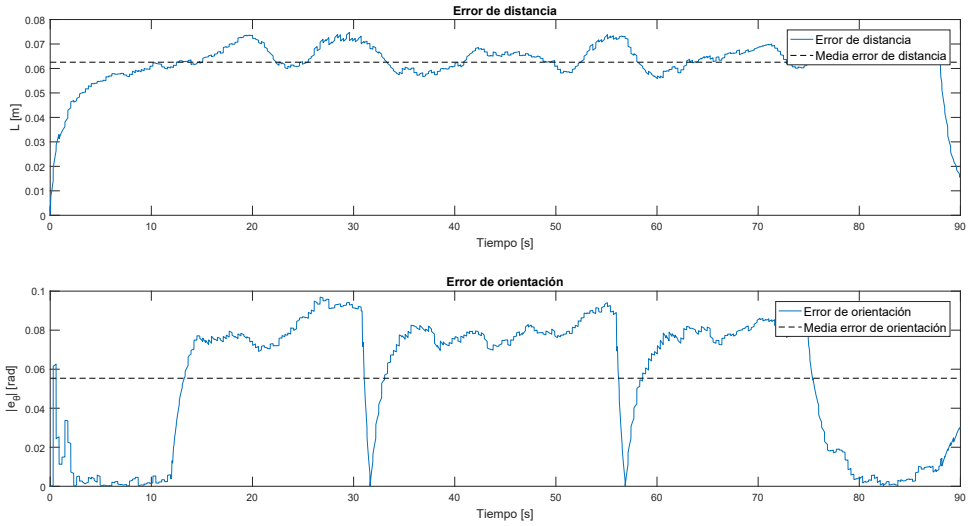

Figura 3.12: Error de distancia y orientación para la simulación de la estimación basada en tiempo (periódica).

avanza en el tiempo. En cuanto al error de orientación, si que está representado el umbral, pero como se puede ver está continuamente por debajo del mismo.

3.3.1.2 Simulación aperiódica umbral adaptativo

Una vez que se ha comprobado la simulación periódica, se va a proceder a realizar la solución propuesta en 2.3.2.2 con un umbral adaptativo en base al error de distancia.

En este caso se ha seleccionado como umbral de seguimiento de trayectoria de 5 mm ($D_{trk} = 5 \cdot 10^{-3}$) mientas que la constante que se ha seleccionado para el umbral de aproximación es de $1/6$ ($K_D = 1/6$). Con estos valores, en la figura 3.13 se muestra el recorrido llevado a cabo por la simulación.

Como se puede comprobar el número de disparos es inferior al caso anterior periódico, pese a ese inferior número de disparos en la figura se puede apreciar cómo se realiza correctamente la trayectoria de referencia.

En la figura 3.14 se muestra el error de distancia y orientación obtenidos de la simulación con respecto de la pose de referencia.

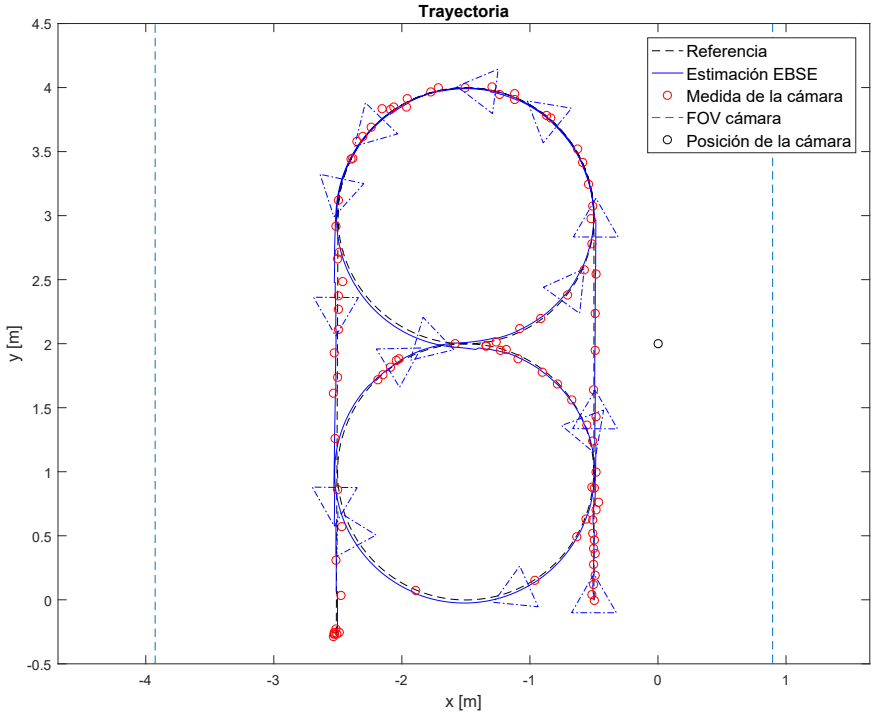

Figura 3.13: Resultado de la trayectoria realizada en la simulación de la estimación basada en eventos con umbral adaptativo.

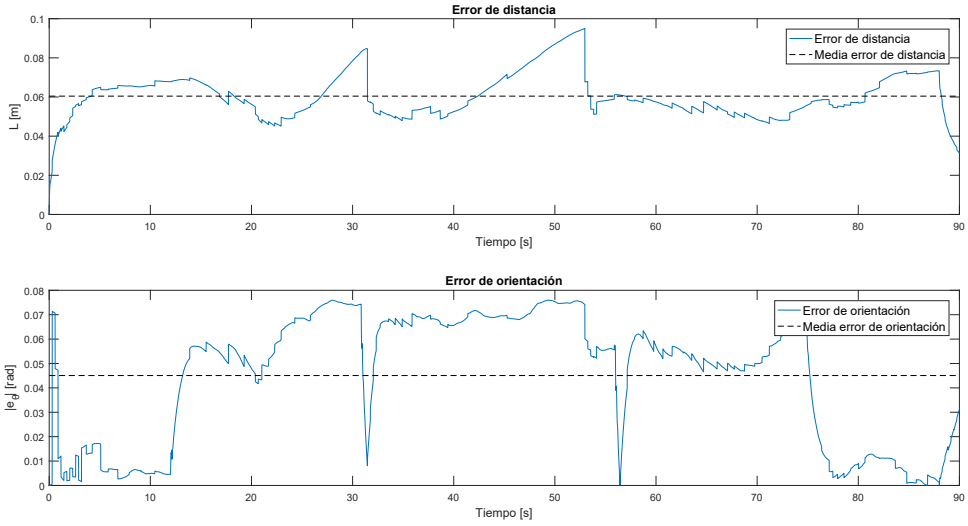

Figura 3.14: Error de distancia y orientación para la simulación de la estimación basada en eventos de forma adaptativa.

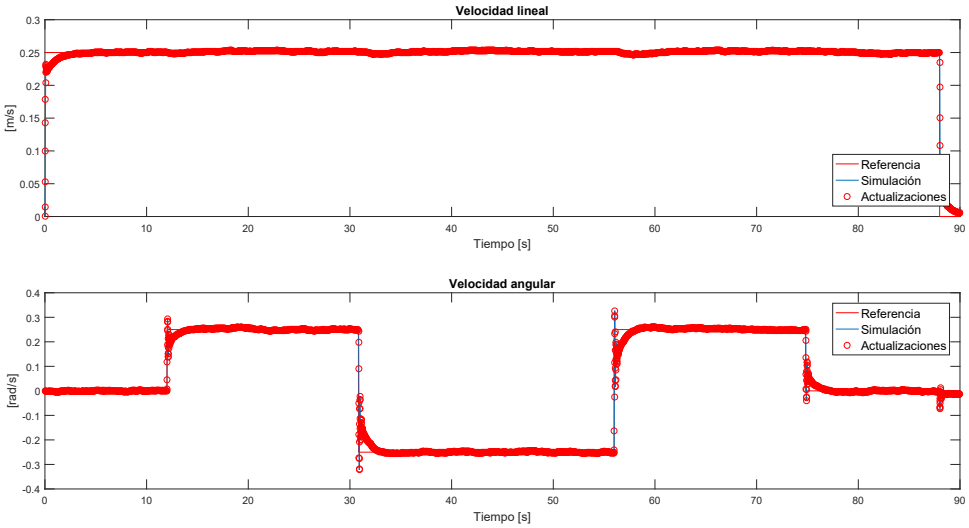

Figura 3.15: Consignas de velocidad lineal y angular para la simulación de estimación basada en eventos de forma adaptativa.

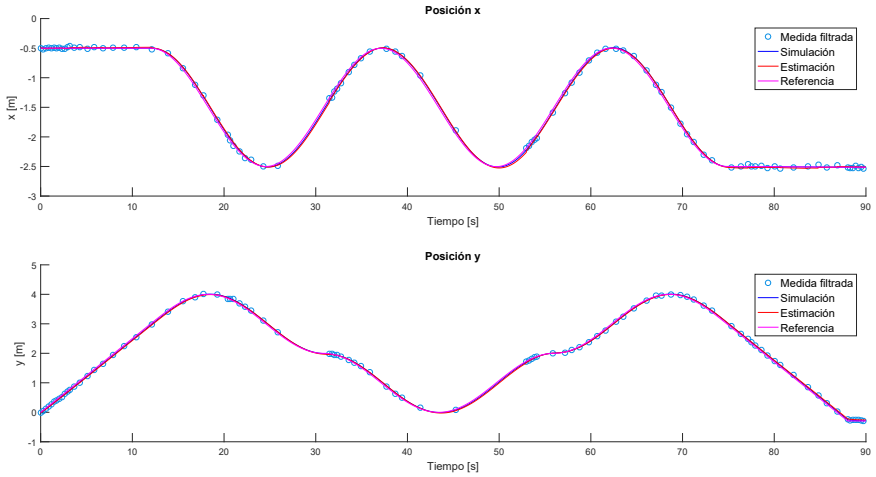

Figura 3.16: Posiciones en X e Y para la simulación de estimación basada en eventos de forma adaptativa.

En la figura 3.15 y 3.16 se puede observar las consignas de velocidad lineal y angular y las posiciones en X e Y respectivamente. Se observa como tanto las consignas de velocidad lineal y angular se siguen correctamente por parte del controlador. Pero además, en la posición en X e Y no existe apenas desviación respecto a la de referencia. El error medio obtenido en la velocidad lineal es de 0.002 m/s mientras que para la velocidad angular es de 0.02 rad/s. En cuanto al error de posición obtenido, el error medio para la estimación es de 6.8 mm, mientras que para el seguimiento se obtiene un error ligeramente superior de 9 mm.

Por último, en la figura 3.17 se muestra la evolución del error en distancia (DRMS) y el error de orientación. A diferencia de antes, en este caso se obtiene una respuesta más plana, debida a la condición del umbral adaptativo. Se puede apreciar ligeramente como se producen ciertas variaciones en el umbral debidas a las condiciones de umbral adaptativo expuestas anteriormente. Estas variaciones se producen sobretodo en zonas de cambio de velocidad angular, ya que la distancia con el punto de referencia cuando se produce una curva aumenta ligeramente y por la condición de umbral adaptativo también lo hace dicho umbral con la función de ajuste K_D. En este caso el número de medidas que se han tomado ha sido 40.5, en una media de diez ejecuciones.

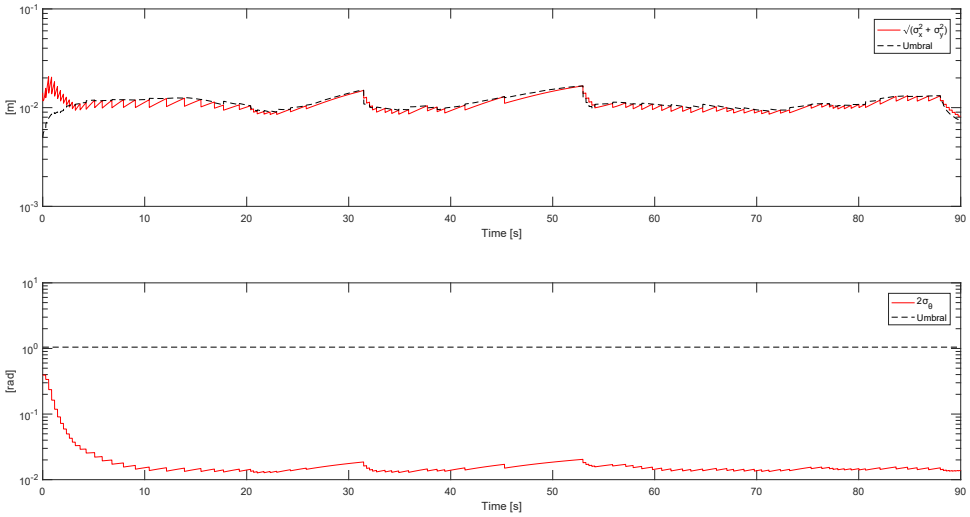

Figura 3.17: Evolución de la DRMS (superior) y el error de orientación (inferior) a lo largo del tiempo.

3.3.1.3 Conclusiones

Para comparar la evolución del error cuadrático medio de distancia (DRMS), en la figura 3.18 se muestra los resultados obtenidos para ambas soluciones tanto periódica, como basada en eventos y adaptativa. Se aprecia cómo el umbral adaptativo responde más rápido al inicio de la simulación frente al caso periódico, pero su evolución se mantiene sin variaciones a lo largo del tiempo en ambos casos.

Se puede decir que la solución adaptativa que se ha propuesto responde de forma similar a la solución periódica, pero en este caso la solución adaptativa genera muchas menos peticiones de medida, 54 frente a 9000, y por tanto menos accesos al canal de comunicaciones compartido. Obteniendo una ISE de 0,77.

3.4 Propuesta de fusión de control y sensado aperiódicos, y comparación con la solución clásica (control y sensado periódicos)

El objetivo principal de este TFM es el minimizar el uso del canal de comunicaciones tanto por la parte de control como la de sensado. Para ello se han presentado en las secciones 3.2 y 3.3 soluciones para reducir el uso del canal de comunicación en ambos casos. En 3.2 se ha presentado una solución para el control, mientras que en 3.3 en base a un control periódico, se ha propuesto una solución para la estimación basada en evento.

En ambos casos se ha comprobado el correcto funcionamiento en simulación de cada uno de ellos de forma independiente, pero sería clave poder obtener una simulación de la fusión de ambas técnicas, para comprobar el funcionamiento de forma correcta previo a la implementación bajo la unidad robótica P3-DX.

Para ello se va a seguir el esquema mostrado en la figura 3.9. Donde se puede observar que la medida obtenida del sensor se adquiere de una cámara que proporciona la pose del robot siempre que $\mathbf{P}$ se mantenga por debajo del umbral adaptativo presentado en el apartado anterior. Esa pose realimentará a la etapa de control, ya sea con la predicción del estado o con la corrección del mismo a partir de la medida de la cámara dependiendo de la evolución de $\mathbf{P}$. Así perimitirá a la unidad móvil realizar la trayectoria deseada y obtener un correcto funcionamiento del sistema.

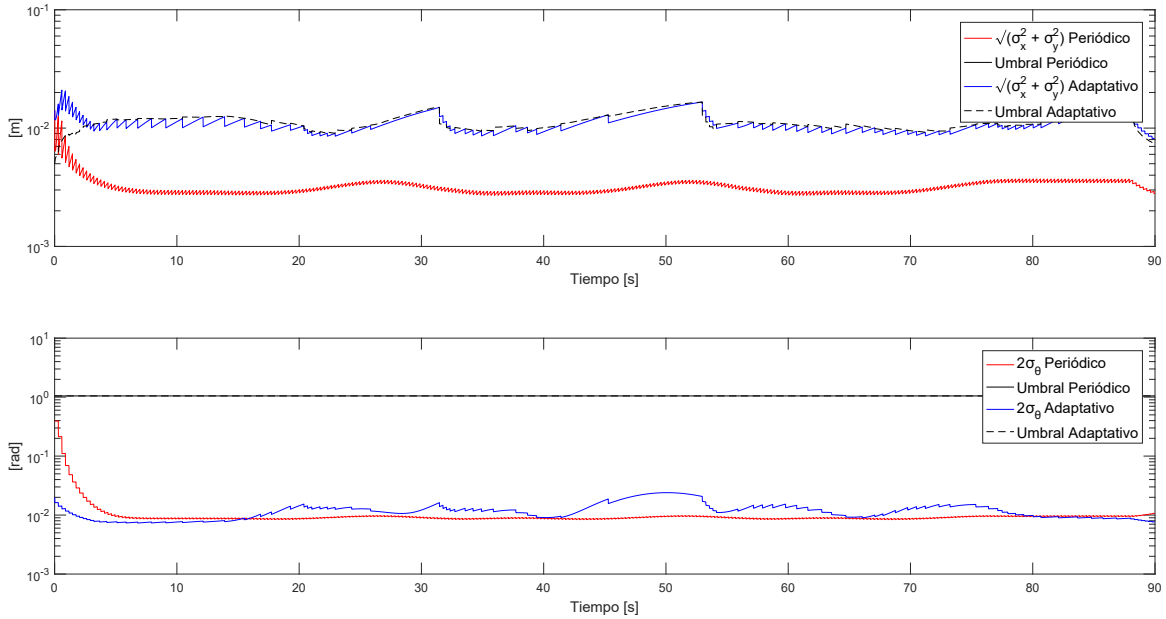

Figura 3.18: Comparación de la evolución de la DRMS (superior) y el error de orientación (inferior) a lo largo del tiempo para el caso periódico y el adaptativo.

La idea inicial era fusionar el control aperiódico *self-triggered* con la estimación basada en eventos (EBSE) gobernada por la matriz de covarianza del error de estimación.

El valor añadido del control *self-triggered* es actualizar la ley de control sin contar con las medidas del sensor, manteniendo el sistema de control en lazo abierto entre disparo y disparo. Sin embargo, en este caso, el EBSE proporciona periódica de la predicción del estado, corregida con medida cuando ésta es relevante en función de los cambios experimentados por la planta. Esto nos lleva a plantear, que la solución idónea de control para el problema planteado de control remoto aperiódico de seguimiento de trayectorias de un robot, sea *event-triggered* en lugar de un *self-triggered*. Por tanto, el último objetivo será comparar esta estrategia con un control y sensado periódico implementado en el centro remoto.

3.4.1 Resultado de la simulación

En la figura 3.19 y 3.20 se muestra la trayectoria con las actualizaciones de control y las medidas del sensor respectivamente. Se puede observar cómo el número de actualizaciones de las consignas de control es ligeramente superior al número de medidas obtenidas. En la figura 3.20 se muestran en rojo las medidas que se han aplicado directamente al control, el resto de actualizaciones de la etapa de control utiliza la estimación realizada en la etapa de sensado.

En cuanto a las consignas de velocidad lineal y angular en la figura se puede comprobar en la figura 3.21 como evolucionan a lo largo del tiempo. En cuanto a la velocidad lineal se mantiene prácticamente estable alrededor de la de referencia a lo largo de toda la simulación, en cambio la velocidad angular hay más variaciones, pero se mantiene estable en torno a la consigna de referencia. Las constantes utilizadas en el controlador, son las mismas que se han expuesto en 3.2.2.1, que si recordamos son $K_{v2} = 0{,}4$ y $K_{\omega 2} = 1$.

En la figura 3.22 se puede ver la evolución de la DRMS y el error de orientación obtenida de la parte del sensor. En este caso los valores para determinar el umbral adaptativo, son los mismo que en el apartado anterior, que si recordamos son $D_{trk} = 5 \cdot 10^{-3}$ y $K_D = 1/6$. Como ocurría con la simulación del apartado 3.3.1.2 el umbral adaptativo presenta unas pequeñas variaciones cuando aparecen cambios en la

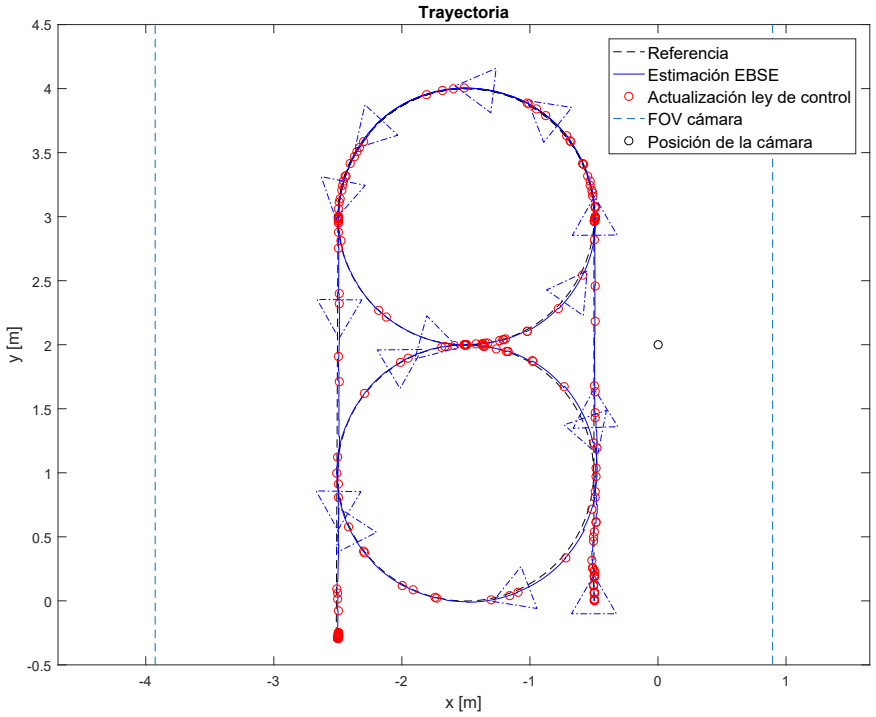

Figura 3.19: Resultado de la simulación de la trayectoria con la propuesta de fusión de control aperiódico y estimación basada en eventos. Trayectoria simulada junto con las actualizaciones de control.

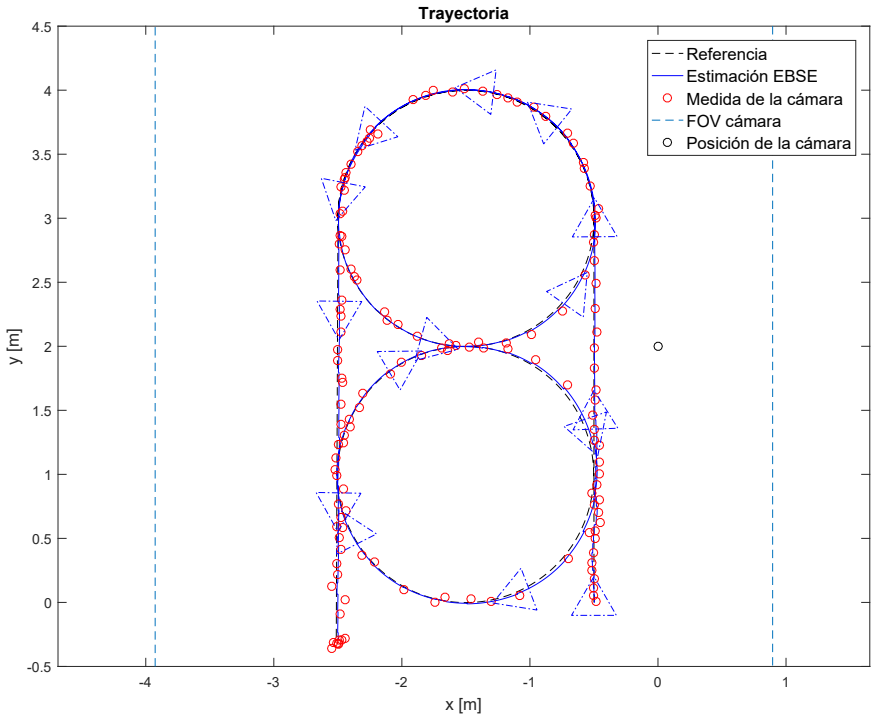

Figura 3.20: Resultado de la simulación de la trayectoria con la propuesta de fusión de control aperiódico y estimación basada en eventos. Trayectoria simulada junto con las medidas tomadas.

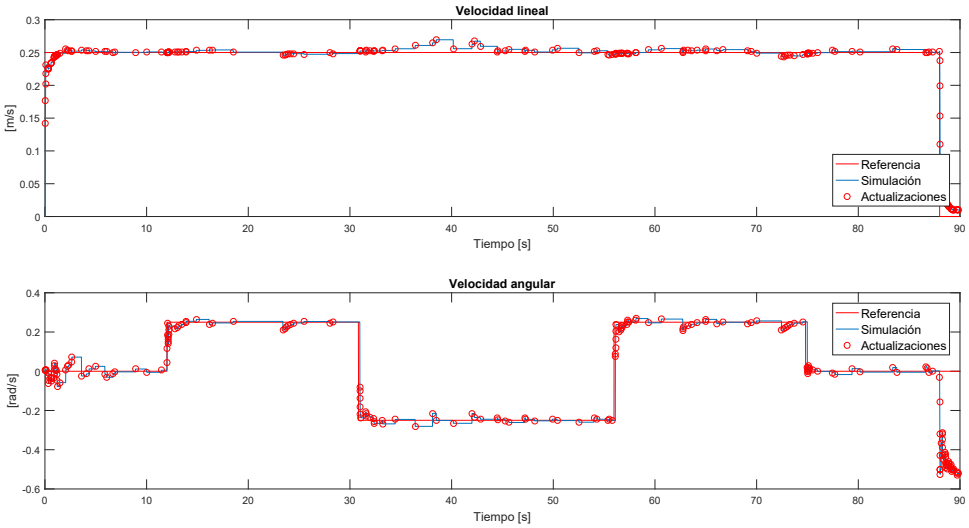

Figura 3.21: Consignas de velocidad lineal y angular para la propuesta de fusión de control aperiódico y estimación basada en eventos.

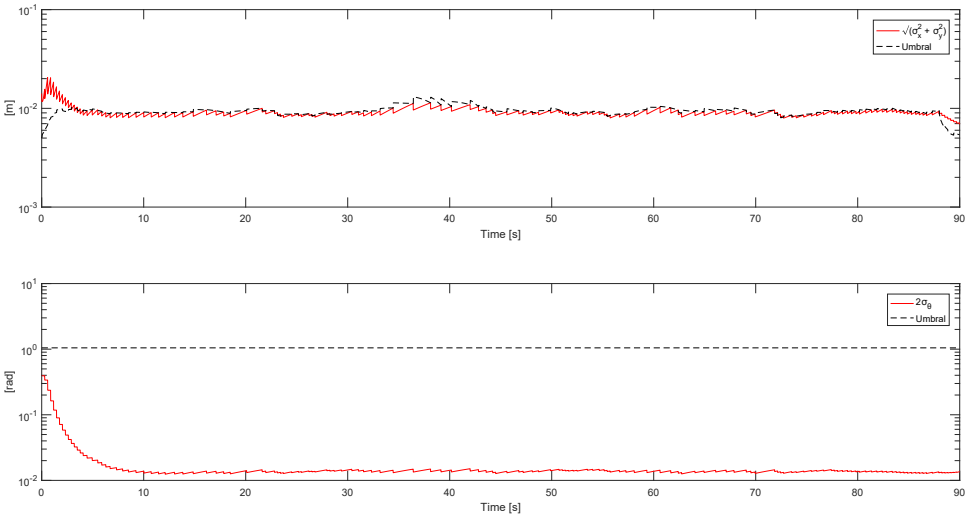

Figura 3.22: Comparación de la evolución de la DRMS (superior) y el error de orientación (inferior) a lo largo del tiempo para la propuesta de fusión de control aperiódico y estimación basada en eventos.

velocidad lineal o angular, pero se mantiene debajo del umbral durante toda la ejecución. Si se deseasen que se produjeran más medidas porque la aplicación así lo requiera, bastaría con cambiar el valor de D_{trk} a un valor inferior, por el contrario si se desean obtener menos medidas habría que aumentar dicho valor.

Por último destacar que el número medio de actualizaciones de las consignas de control en diez ejecuciones es de 175.2 actualizaciones, mientras que en esas mismas diez ejecuciones las medidas obtenidas por el sensor aplicando en umbral adaptativo es de 45.9 medidas.

3.5 Conclusiones

A lo largo del capítulo se ha comprobado el funcionamiento mediante simulación de los principales objetivos del trabajo fin de máster, que son la propuesta de control aperiódico, la propuesta de sensado basado en covarianza, simular cada una de ellas y proponer una solución compuesta.

Se han propuesto condiciones de disparo para hacer que el controlador basado en Lyapunov pudiera

tener un menor número de actualizaciones de las consignas, de esta forma se ahorran accesos al canal de comunicación por parte del controlador garantizando la estabilidad del sistema en lazo cerrado.

Estas condiciones de disparo varían dependiendo de la estrategia que se esté llevando a cabo en cada momento y proporciona un peso diferente a los errores de distancia u orientación, ya que es posible crear una función de pesos que proporcione al algoritmo una mayor importancia a la orientación o la distancia. Esto permite adaptarse a las distintas aplicaciones que se puedan dar, ya que puede interesar al diseñador que el esfuerzo de control para corregir el error de orientación sea más importante que el esfuerzo de control para corregir el error de distancia.

En cuanto a la parte de estimación basada en eventos, se han impuesto las directrices para llevar a cabo la posterior implementación en una cámara real, tratando que las condiciones de simulación sean lo más fieles a la realidad.

En este caso, se ha comprobado el funcionamiento en base al modelo de la planta, pero en ningún caso teniendo en cuenta el controlador basado en Lyapunov. Así se ha podido comprobar como la condiciones de disparo basadas en la matriz de covarianza $\mathbf{P}$ permiten llevar a cabo la correcta estimación y cuando se supera el umbral decovarianza del error de estimación de estados y corregir la posición.

Para concluir esta parte del desarrollo se ha presentado una propuesta de fusión de control y estimación basada en eventos. En ella se ha mostrado que los resultados que se han obtenido mejorar claramente el número de accesos al canal de comunicación, pero para tener clara la mejora que esto supondría en las tablas 3.3 se recogen el número de actualizaciones y el número de medidas que se producen en cada una de las situaciones planteadas, así como el valor de ISE obtenido y el error de distancia y orientación. Aunque son actualizaciones o medidas, se traducen a accesos al canal de comunicación, ya que tanto la cámara como la unidad móvil utilizan dicho medio para intercambiar la información.

Por tanto si planteamos una solución puramente periódica el número de accesos al canal que se llevarían acabo sería 9311 accesos (9000 debidos a las actualizaciones de control y 311 a los envíos de medida al centro centro remoto por parte del sensor). En cambio, si planteamos la solución desarrollada en la sección anterior, utilizando un controlador aperiódico y un estimador basado en covarianza, el número de accesos que se producen son de unos 221 (175.2 de media debido a las actualizaciones de control y 45.9 de media a las medidas enviadas al centro remoto) de media que equivale aproximadamente a un 2.5 % del uso del canal respecto a la solución periódica.

Para un sistema dónde sólo se dispone de una unidad móvil y un sensor puede que esta solución sea menos necesaria, pero esto además de reducir el numero de accesos al canal, permite que se puedan utilizar muchas más unidades móviles y sensores utilizando el mismo canal.

Parámetro	Control Per. y EBSE Per.	Control Per. y EBSE Aper.	Control Aper. y EBSE Aper.
ISE	0.73495	0.82352	0.89099
Nº de medidas	311	93	144
Nº de actualizaciones	9000	9000	209
Med. Error Distancia [m]	0.0625	0.0747	0.0859
Med. Error Orientación [rad]	0.05537	0.06541	0.07142

Tabla 3.3: Tabla resumen de resultados de la simulación.

Capítulo 4

Resultados experimentales

4.1 Introducción

A lo largo de este capítulo se van a mostrar los resultados obtenidos de las experimentaciones realizadas en el demostrador.

Se han llevado a cabo diferentes experimentaciones relacionadas con las simulaciones realizadas en el capítulo anterior con el fin de constatar los resultados obtenidos en la simulación.

El capítulo está estructurado en cuatro apartados:

4.1. Se trata de la introducción al capítulo.

4.2. Se describen los aspectos más relevantes del demostrador donde se van a desarrollar cada uno de los experimentos, así como la estrategia de experimentación que se ha llevado a cabo.

4.3. Se muestran los resultados obtenidos en los distintos experimentos realizados, en este caso se han desarrollado tres experimentaciones:

   (a) La primera (4.3.3) se centra en evaluar las mejoras que suponen el control aperiódico en presencia de un sensor externo que permita corregir la posición del robot.

   (b) El segundo de los experimentos (4.3.2) dejando a un lado la parte de control aperiódico y usando un control periódico, se va a recoger los resultados de sensado basado en covarianza.

   (c) El último de los experimentos (4.3.4) se realiza la ejecución la propuesta de control aperódico, junto con el sensado basado en covarianza.

4.4. El último de los apartados del capítulo de resultados está reservado a las conclusiones que se puedan adquirir a lo largo del mismo.

4.2 Entorno experimental

Los experimentos se van a desarrollar en un demostrador realizado para poder implementar distintas soluciones de control. Este demostrador se sitúa en uno de los pasillos de laboratorios del Edificio de la Escuela Politécnica de la Universidad de Alcalá, situado en la zona oeste del edificio. En la figura 4.1 se muestra una situación del pasillo donde se encuentra el demostrador.

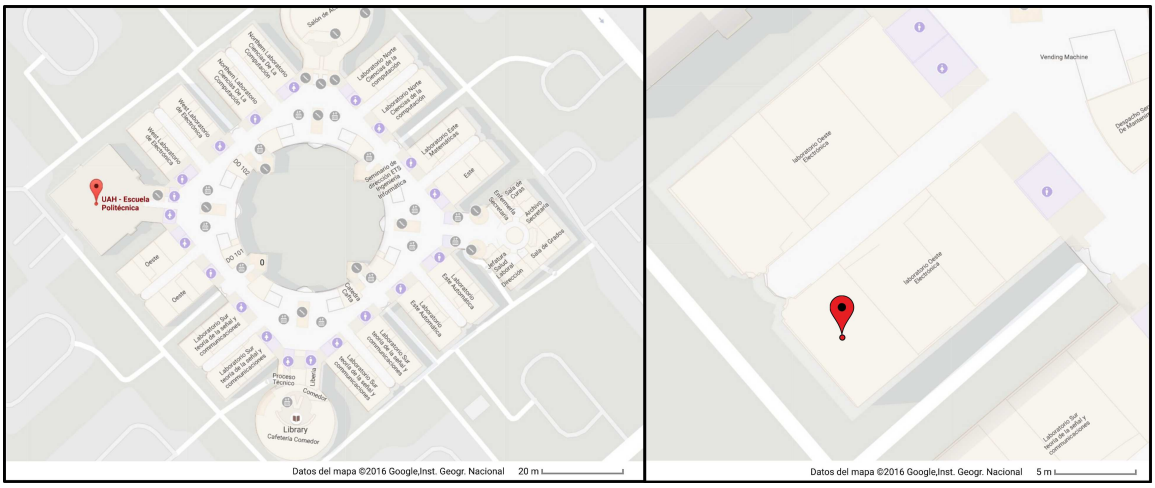

Figura 4.1: Mapa de situación del demostrador. Primera planta del Edificio de la Escuela Politécnica de la Universidad de Alcalá (izquierda). Pasillo de localización del demostrador (derecha).

El pasillo tiene al fondo del mismo un ventanal, por lo que los experimentos están expuesto a posibles variaciones en la iluminación o deslumbramientos por la iluminación excesiva del sol. Otros factores que pueden intervenir en el entorno experimental es la propia iluminación interior del edificio, que puede variar dependiendo de la franja horaria en la que se esté desarrollado el experimento.

Otros factores que pueden afectar al demostrador son las interferencias en la banda de 2.4GHz utilizada para la conexión inalámbrica entre el centro remoto y las unidades móviles o unidades sensoriales. La banda de 2.4GHz es utilizada por WiFi, Bluetooth o ZigBee que son las tecnologías que han desarrollado estándares para su uso. Aunque en este caso la posibilidad de saturar el canal, o que aparezcan factores puedan degradarlo es poco probable, ya que se utiliza una red privada en la que los usuarios del canal WiFi son pocos elementos (el número de unidades móviles y unidades sensoriales es reducido, por lo que no se llega a generar una carga excesiva a nivel de datos sobre el canal de comunicaciones).

4.2.1 Características del demostrador

El demostrador tiene a su disposición un conjunto de unidades robóticas Pioneer 3-DX, que se han utilizado en otro tipo de ensayos para realizar ensayos de unidades móviles en convoy a lo largo del pasillo, estos trabajos se recogen en [34].

En trabajos más recientes [36], el demostrador se ha utilizado para realizar el guiado de unidades móviles utilizando un estimador basado en eventos y una cámara.

Como se ha comentado en la introducción de este trabajo, las unidades móviles a controlar son robots P3-DX, los cuales incorporan una placa base Via-Epia (mini-ITX), conectada a través de un convertidor Ethernet-Wireless que permite la comunicación entre la unidad móvil y el centro remoto. Todo ello ejecuta un sistema operativo de tiempo real Ubuntu kernel 2.6.23 con RTAI.

En cuanto a los sensores que se van a utilizar son cámaras, concretamente se va a usar la cámara RGB que incorpora la Kinect 2 (ver figura 4.2), esta tiene una resolución de 1920 x 1080 píxeles. El tiempo de procesamiento de la imagen que se maneja actualmente es aproximadamente de 300ms. El intercambio de datos entre la cámara y el PC del sensor se realiza a través del USB 3.0, la cual tiene una tasa de transmisión superior a la del USB 2.0 usado en la versión previa de la kinect.

Las cámaras están conectadas a unos mini PCs dispuesto a lo largo del pasillo. Algunas características relevantes de estos ordenadores, se trata de los Intel NUC con un procesador Intel Core i3 con 4GB de

Figura 4.2: Detalle del sensor utilizado (Kinect 2).

RAM, ampliable hasta 16GB, los cuales disponen de un disco duro SSD de 120GB y cuatro puertos con conectividad USB 3.0, además de LAN y WiFi integrado. Todo en un reducido tamaño de 11.5 x 11.1 x 4.9 cm, una de las razones por las que se seleccionó este tipo de ordenador fue por su reducido tamaño ya que facilita la integración de los componentes en el pasillo.

En la figura 4.3 muestra un esquema gráfico de la disposición de los distintos elementos a lo largo del pasillo. El demostrados dispone de cinco cámaras que cubren la visualización del pasillo completo, pero en este caso sólo se va a realizar uso de una de ellas para llevar a cabo los experimentos, ya que para cubrir una trayectoria que permita recorrer todo el pasillo, es necesario establecer un criterio que permita seleccionar la información de cada sensor que menor incertidumbre de medida tenga u otros criterios. Dado que dicho aspecto está fuera de los objetivos de este proyecto, se deja como trabajo futuro el desarrollar dicho mecanismo que permita hacer uso de todas las cámaras que dispone el demostrados.

Actualmente la unidad móvil tiene conexión inalámbrica con el centro remoto, en cambio los ordenadores a los que van conectados las cámaras están conectados a través de cable Ethernet a un switch que se conecta al router directamente. Debido a que la instalación de las cámara, así como de los ordenadores se ha realizado recientemente, se ha preferido mantener la conexión a través de cable Ethernet en vez de utilizar la conexión inalámbrica de los módulos sensoriales, de esta forma se reducen posibles factores de error en las pruebas realizadas. Pero para utilizar los sensores de forma inalámbrica, bastaría con habilitar la conexión WiFi de los ordenadores de las unidades sensoriales y establecer en el programa de control la dirección IP de correspondiente a la conexión WiFi de los módulos sensoriales.

En la figura 4.4 se muestra una imagen del pasillo donde se sitúa el demostrador. En ella aparece la unidad móvil con el patrón que será detectado por la cámara situada en el techo encima suyo.

4.2.2 Estrategia y metodología de experimentación

Para abordar la experimentación se ha determinado una estrategia progresiva que permita validar el correcto funcionamiento de cada una de las partes, pero sobre todo de aquellas que dependen de forma directa con el ensayo posterior. Por tanto, el orden de ensayos de experimentación que se va a llevar a cabo es el siguiente:

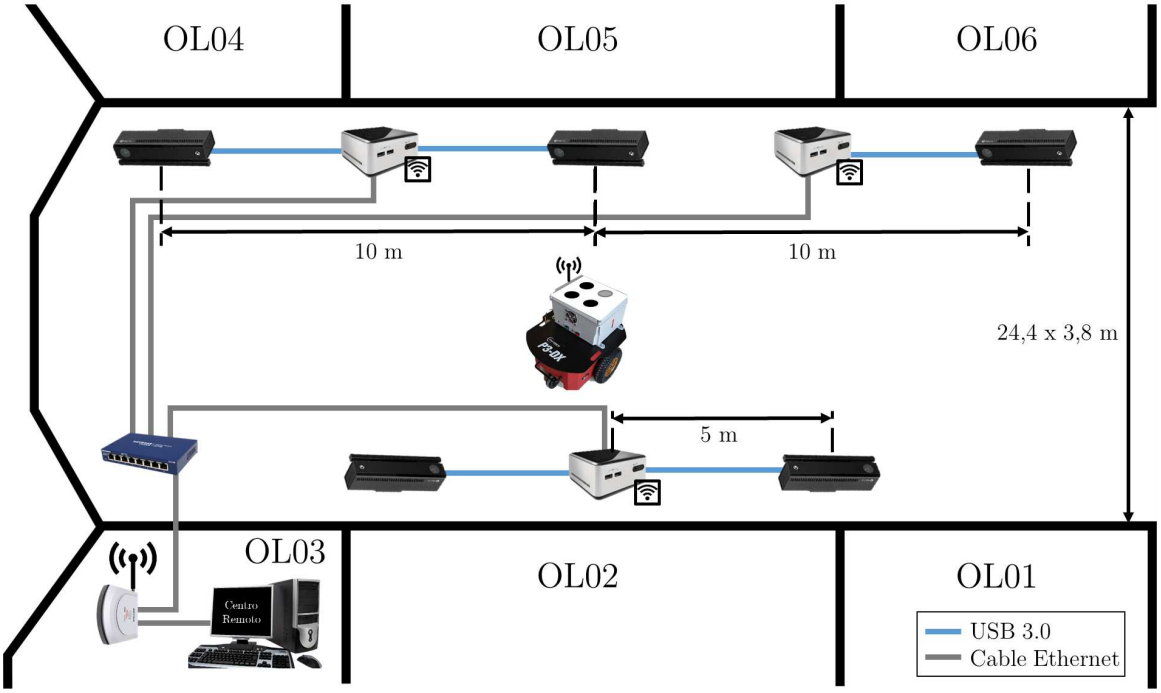

Figura 4.3: Esquema con elementos del demostrador.

Figura 4.4: Fotografía del demostrados utilizado.

1. Implementación de servosistema con realimentación de estados.
2. Implementación de Lyapunov Based Controller (LBC).
3. Incorporar la parte de estimador de estados basado en eventos (EBSE) se ejecute también en el centro remoto.
   (a) Explorar código en C++ realizado en trabajos previos.
   (b) Incorporación del EBSE a Simulink.
   (c) Validación de comunicación entre centro remoto (EBSE) y sensor.
4. Implementación de LBC periódico junto con UKF periódico.
5. Implementación de LBC periódico junto a EBSE con umbral adaptativo.
6. Implementación de LBC aperiódico junto con UKF periódico.
7. Implementación de LBC aperiódico junto a EBSE con umbral adaptativo.

Aunque esa ha sido la estrategia llevada a cabo, en este capítulo sólo se van a mostrar los resultados obtenidos para los últimos (4, 5, 6 y 7) puntos que son los que se centran en los objetivos de este trabajo. Los puntos previos se han desarrollado para comprobar el correcto funcionamiento tanto de la migración a una versión más actual de Ubuntu, como el funcionamiento del demostrador.

Con el fin de tener en mente un esquema general del flujo de cada uno de los programas realizados, en la figura 4.5 se muestra el flujograma de ejecución de los distintos programas. Básicamente al inicio de cada una de las ejecuciones se establece conexión con los distintos sensores y unidades móviles que intervengan en el ensayo, en este caso cómo máximo en cada ensayo habrá una unidad móvil y un sensor. Seguidamente se pide una primera lectura de los sensores del robot, así como de los módulos sensoriales, continuando con la ejecución del algoritmo de control así como la escritura de comandos en caso de que sea necesario, repitiendo este proceso hasta que el programa concluya. Una vez que se ha terminado la ejecución del programa, se termina también la conexión a los distintos servidores.

En cuanto a las comunicaciones que se van a utilizar son cuatro sockets, ya que se utilizan dos para establecer la comunicación entre el centro remoto y la unidad móvil, pero además se va a disponer de otras dos para establecer comunicación con el ordenador que controla los sensores. En la figura 4.6, se muestra en detalle la configuración utilizada en las comunicaciones incluyendo las direcciones IP, así como los puertos utilizados por cada socket. La figura tiene representado la conexión al mini PC del sensor distintos sensores, como una cámara, un sensor láser tipo lidar o cualquier otro sensor, en los ensayos sólo se dispone de una cámara para cada experimento, pero el esquema trata de representar que con independencia del sensor que se tenga, se podría procesar y mandar esa medida usando el mismo esquema de funcionamiento, así el demostrados permite una gran versatilidad, dando la posibilidad de que se usen para otros usos o en distintas situaciones.

4.2.2.1 Esquema de controlador periódico con estimador de estados basado en covarianza

En este apartado se va a proporcionar de forma general la estructura del controlador periódico, independientemente si se está utilizando un EBSE periódico o bien con un umbral adaptativo. En cualquiera de los dos casos el esquema seguiría siendo el mismo y cambiaría la condición de disparo implementada dentro del EBSE.

En la figura 4.7 se muestra el esquema general de un controlador periódico al que se le ha añadido el EBSE para proporcionar la estimación de la pose del robot siempre y cuando no se tenga medida. Los

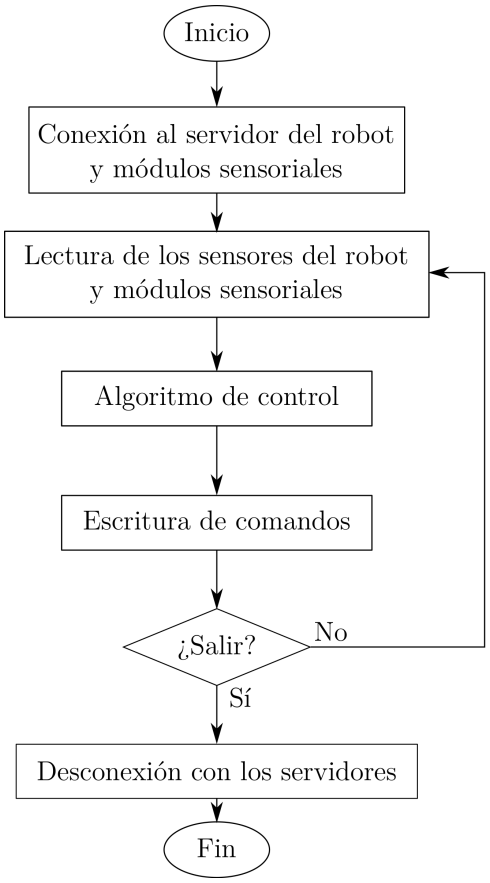

Figura 4.5: Diagrama de flujo del programa realizado.

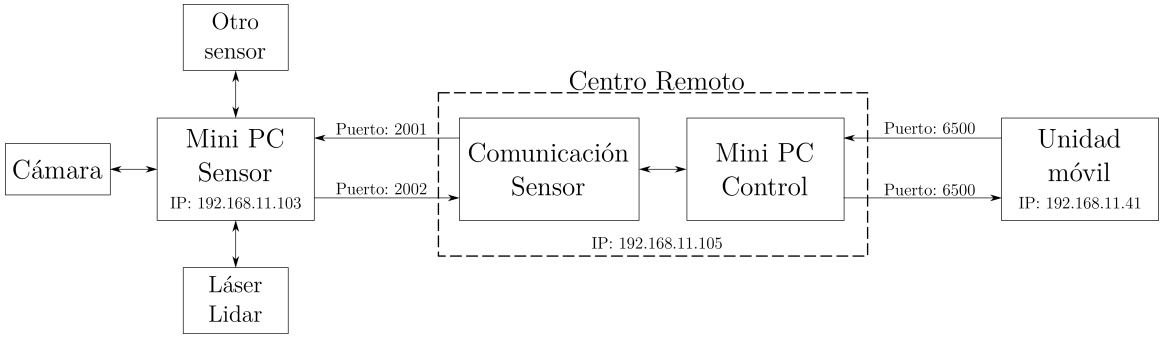

Figura 4.6: Esquema de comunicaciones del conjunto de elementos que compone el demostrador.

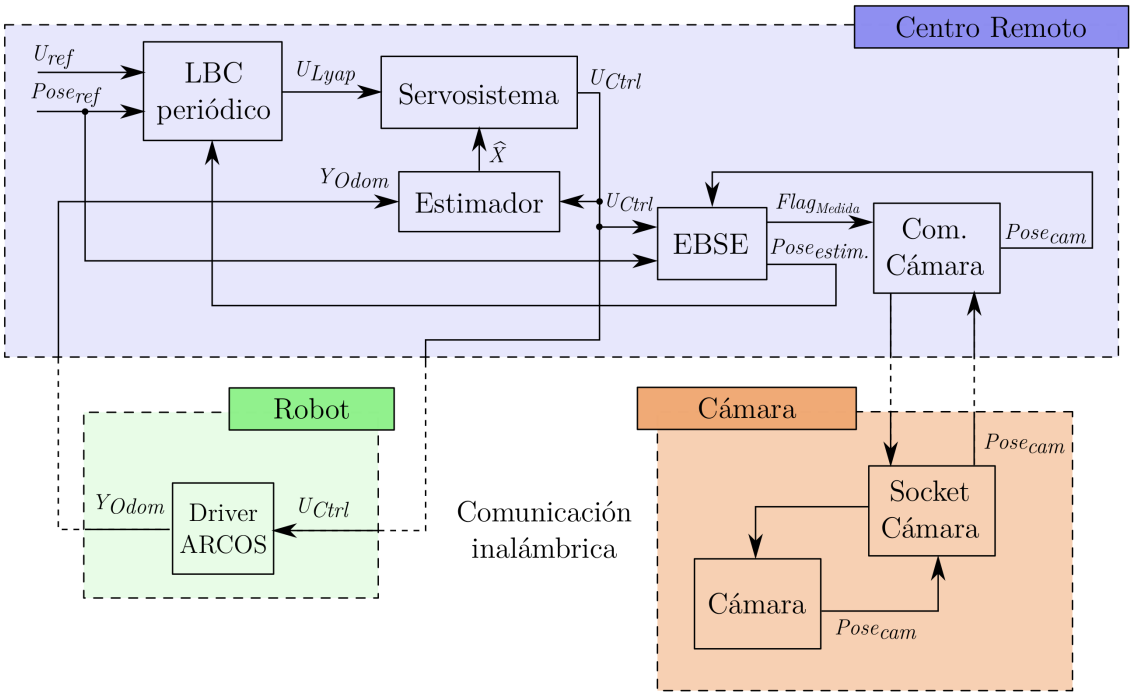

Figura 4.7: Esquema del controlador periódico junto con el Estimador de Estados Basado Covarianza (EBSE).

distintos elementos que intervienen se muestran en la figura, cabe destacar que a diferencia del esquema mostrado en la figura A.7 en el capítulo A.1.3.3 donde se utilizaba el modelo cinemático para obtener la diferencia entre la pose de referencia y la obtenida a partir del modelo cinemático, en este caso se va a usar la pose obtenida por cámara siempre que haya medida y sino la estimación realizada por el EBSE.

Destacar que la pose obtenida de la cámara se introduce al EBSE en vez de llevarla al LBC, ya que en el EBSE está implementado el algoritmo de corrección como se mostró en 3.3.

Tanto el driver del robot, como las comunicaciones con la cámara mostrados en la figura 4.7, son los bloques encargados de la parte de las comunicaciones, siempre que sea necesario serán los encargados de escribir la información en los paquetes que serán enviados o bien a la unidad móvil o al sensor.

4.2.2.2 Esquema de controlador aperiódico con estimador de estados basado en covarianza

A diferencia del control periódico, el control aperiódico no se actualizan las consignas de velocidad lineal y angular cada cierto periodo sino que, como se ha mencionado en el capítulo 3.2, se evalúa la función de Lyapunov para mantenerla por debajo de un umbral que en este caso se trata de que no se vuelva creciente, en el momento que la función de Lyapunov se vuelve creciente, se actualizan la consigna de velocidad lineal y angular.

En una primera implementación, se ha desarrollado el ensayo utilizando un controlador *self-triggered* el cual realiza una predicción del siguiente instante en el que se deben actualizar la ley de control, mientras tanto no se actualiza dicho valor. Esto provocaba que se dispone de información actualizada del estado de la planta que no se aprovecha con la solución *self-triggered*.

De modo, que se ha optado por no realizar un controlador *self-triggered* sino controlador *event-triggered*, el cual evalua la función de Lyapunov de forma periódica de manera que si la función V se vuelve creciente, la ley de control se aplica a la unidad móvil, que mantiene ese valor hasta que el centro remoto no vuelva actualizar las consignas de velocidad lineal y angular.

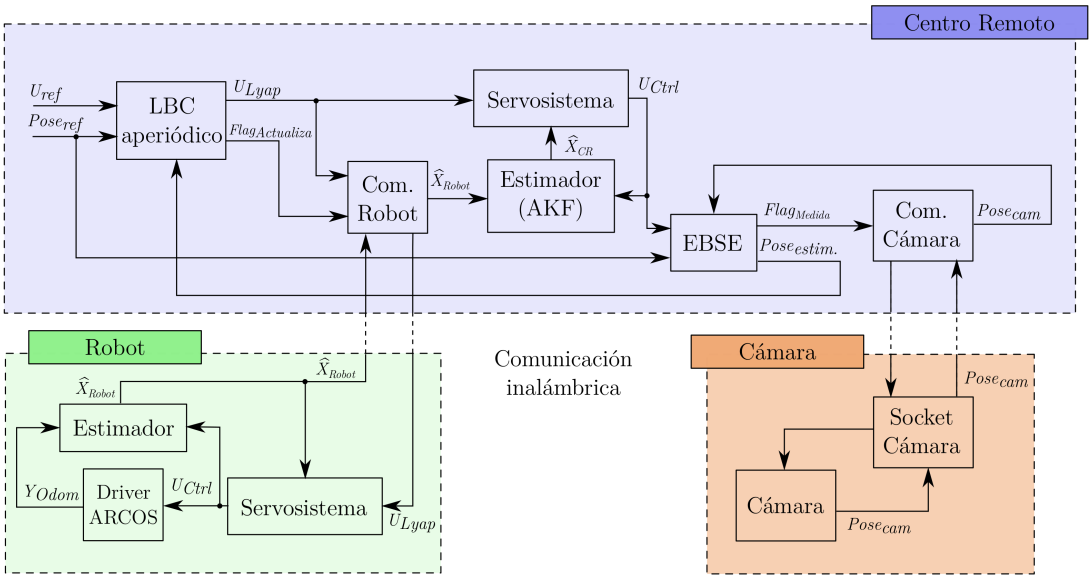

Figura 4.8: Esquema del controlador aperiódico junto con el Estimador de Estados Basado Covarianza (EBSE).

En la figura 4.8 se muestra el diagrama de bloques completo, con los tres elementos conectados en red: centro remoto, robot y cámara de visión artificial.

El centro remoto realiza dos tareas fundamentales: control LBC aperiódico disparado por eventos y estimación de estados basada en eventos (EBSE). Cada vez que el controlador genera un disparo, se envía la señal de actuación U_{Lyap} al servosistema implementado en el robot y este devuelve el estado del servosistema $\hat{X}_{Robot}$. Mientras no recibe consignas actualizadas del LBC, el servo del robot cierra periódicamente su lazo de control con la estimación del estado a partir de la última consigna recibida y las lecturas del encóder.

El EBSE predice periódicamente la pose del robot, hasta que se requiere una nueva medida de la cámara y se realiza la correspondiente corrección. Para la fase de predicción requiere de la salida periódica del servo implementado en el CR. Este servo necesita el estado estimado del robot, que le proporciona periódicamente el AKF y es corregido cuando el robot real aporta información en los instantes de disparo del controlador.

4.3 Resultados experimentales

En este apartado se muestran los resultados de los ensayos realizados en el demostrador. Como se ha comentado anteriormente, los resultados se van a centrar únicamente en la experimentación con el control periódico y aperiódico junto con el sensado periódico y con el sensado basado en covarianza con umbral adaptativo, en todos sus pares de combinaciones.

4.3.1 Experimentación de control periódico junto con sensado periódico

Un controlador periódico realimentado con los valores leídos de los sensores de odometría, hace que únicamente la unidad móvil mantenga las consignas de velocidad lineal y angular, de forma que si las ruedas patinan porque el suelo es deslizante, el control mantiene los valores de las consignas de velocidad

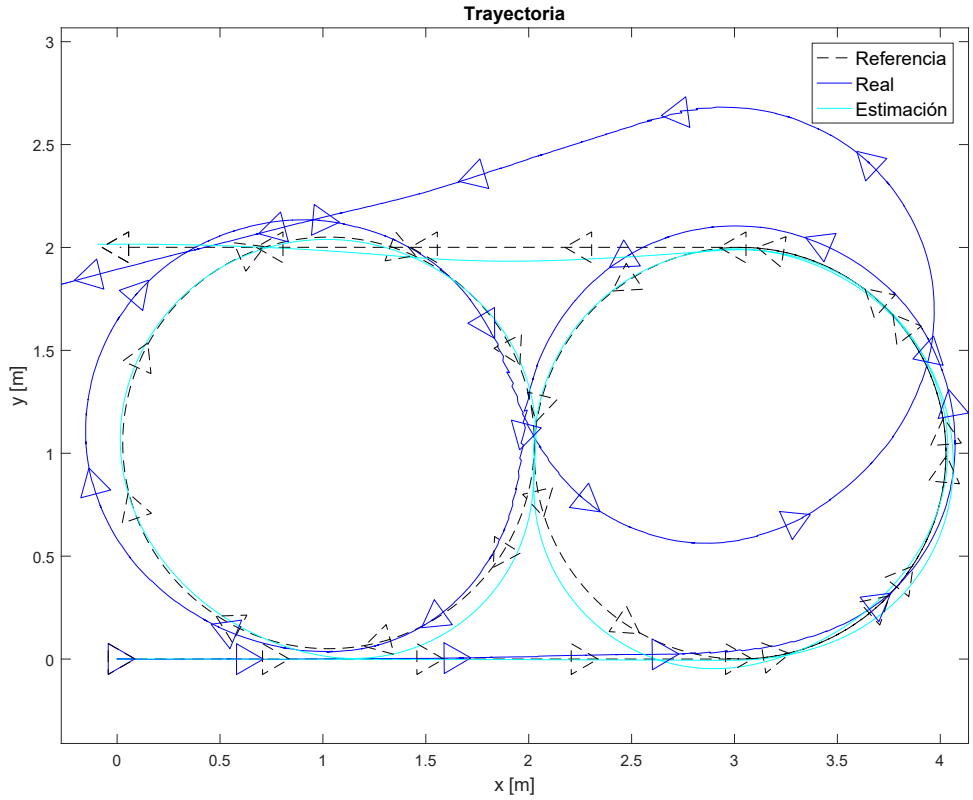

Figura 4.9: Ejemplo de trayectoria realizada con presencia de errores de odometría.

lineal y angular, pero no se mueve lo que debería. Esto se conocen como errores de odometría y están presentes en la mayoría de los robots, para solucionarlos existen varias propuestas que involucran más sensores embarcados en la unidad móvil y el problema se traslada a un problema de localización conocido como SLAM (Simultaneous Localization And Mapping). La figura 4.9 muestra un ejemplo donde se producen estos errores de odometría, se ve que la trayectoria seguida por el robot difiere mucho de la trayectoria de referencia debido a que se han producido perturbaciones externas que no han permitido al robot seguir la trayectoria correctamente, pero la estimación calculada a partir de los valores de velocidad lineal y angular leídos de los sensores de odometría se aproxima mucho más a la trayectoria y es lo que el controlador piensa que está realizando la unidad móvil.

Para corregir estos errores de odometría y que la unidad móvil sea capaz de realizar una trayectoria predefinida, es necesario que se utilice un sensor externo, en vez de un sensor embarcado, ya que va a proporcionar al robot la posición real, aunque el robot patine o similar, de esta manera se puede garantizar la realización correcta de la trayectoria de la unidad móvil. Además, el usar un sensor externo permite que al controlador basado en Lyapunov realiamentarlo con el valor de la pose de dicho sensor.

En este caso, el controlador se va a ejecutar de forma periódica y el sensor realizará la detección del robot de forma continua, de esta manera que cuando se obtenga una medida nueva se corregirá el valor de la estimación.

En la figura 4.10, se muestra la trayectoria del ensayo realizado para este caso, en la misma se representa con un círculo el valor de la medida obtenida por el sensor. Se puede comprobar que la trayectoria estimada por parte del EBSE realiza de forma correctamente la trayectoria de referencia. Es evidente que el robot ha realizado correctamente la trayectoria en este caso, pese a que se produzcan errores de odometría, ya que el sensor proporciona la posición del robot.

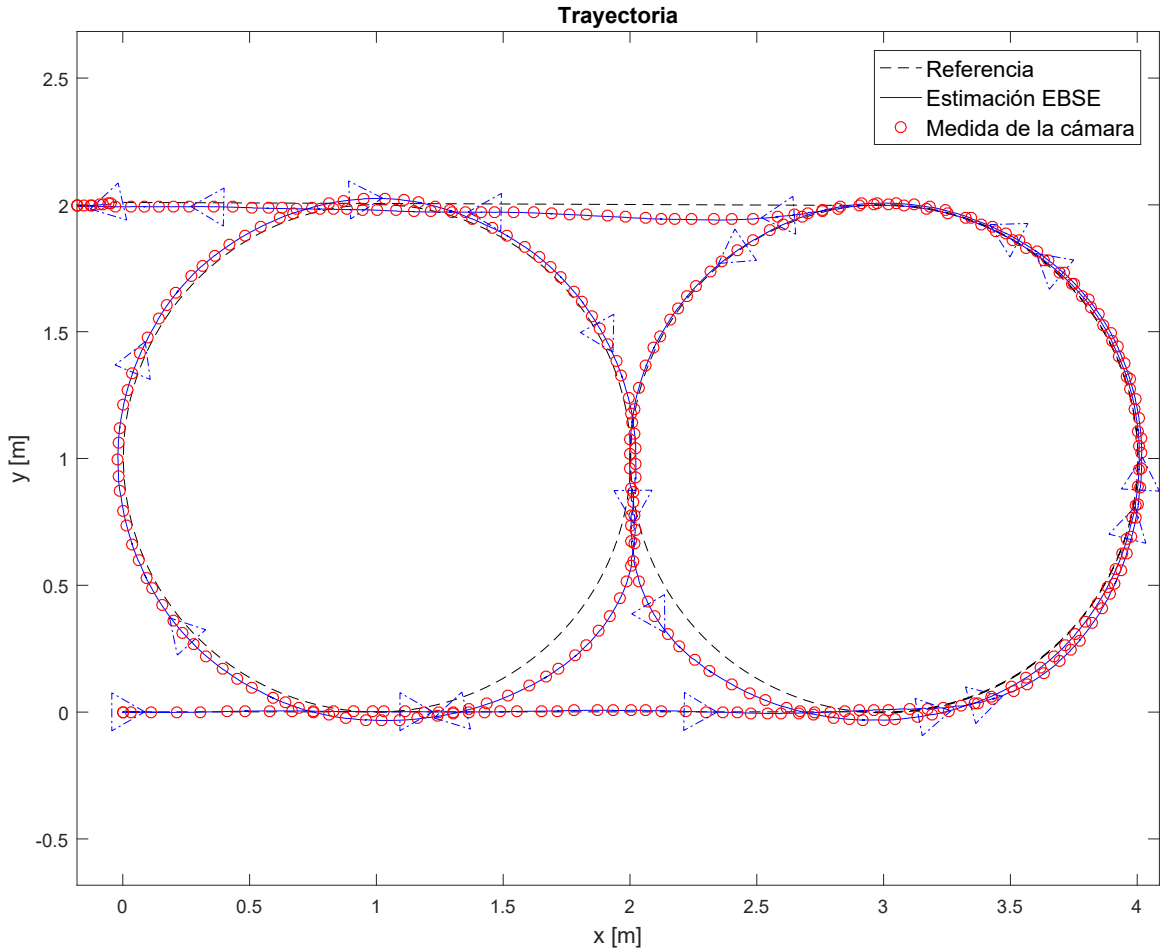

Figura 4.10: Trayectoria obtenida del ensayo con controlador periódico y estimación basada en tiempo (periódica).

De esta forma se ha conseguido que mediante un sensor externo, el robot realice correctamente la trayectoria, pese a que se producen ciertos errores, queda determinado por tanto, que es fundamental la utilización de un sensor externo que realimente al sistema de control, permitiendo así realizar el seguimiento del robot en todo momento.

En la figura 4.11, se muestra una evolución de la posición en X e Y del robot a lo largo del tiempo. Está representada la evolución de la posición de referencia, la obtenida por el estimador de estados (EBSE) y de la medida.

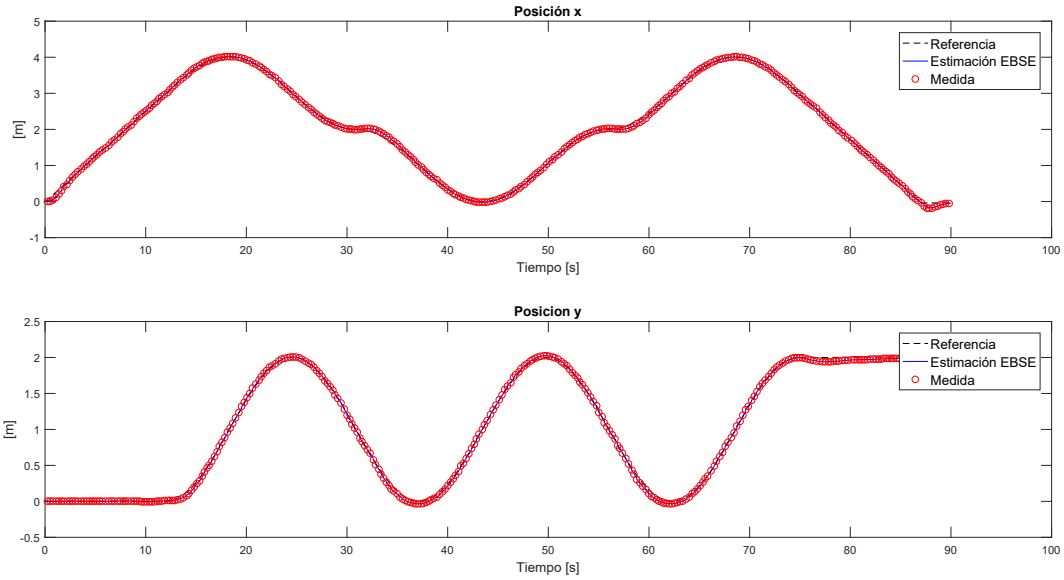

Figura 4.11: Evolución de la posición x e y obtenidas del ensayo con controlador periódico y estimación basada en tiempo (periódica).

Si se analizan los valores que se han obtenido de velocidad lineal y angular, en la figura 4.12 se muestra la evolución de las consignas de velocidad lineal y angular de referencia, así como la obtenida del controlador de Lyapunov y la obtenida por la odometría.

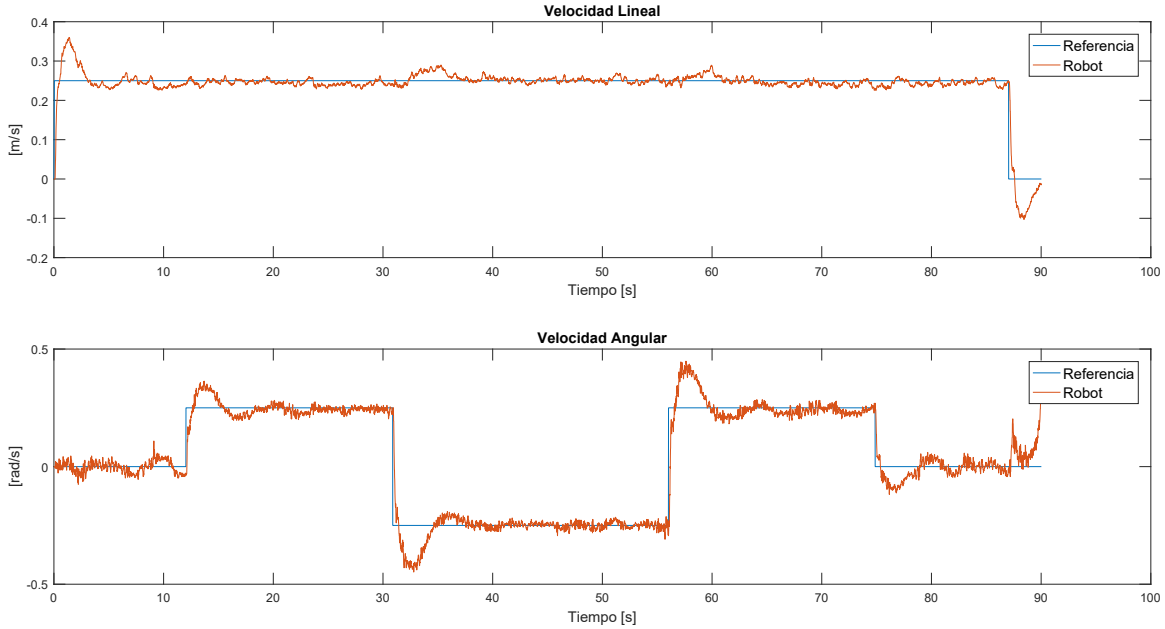

Figura 4.12: Consignas de velocidad lineal y angular obtenidas del ensayo con controlador periódico y estimación basada en tiempo (periódica).

En la figura 4.13 muestra la evolución del error de distancia, así como el error de orientación. Se puede observar que donde en la trayectoria el robot se ha desviado ligeramente de la trayectoria (ver figura 4.9), se obtienen valores mayores de error de distancia y orientación, esto es debido a los cambios producidos en las velocidades de referencia para generar la trayectoria. De media el error de distancia se sitúa en 3,3 cm, mientras que el error de orientación son aproximadamente 0,03 radianes, que equivale a 1,72^o.

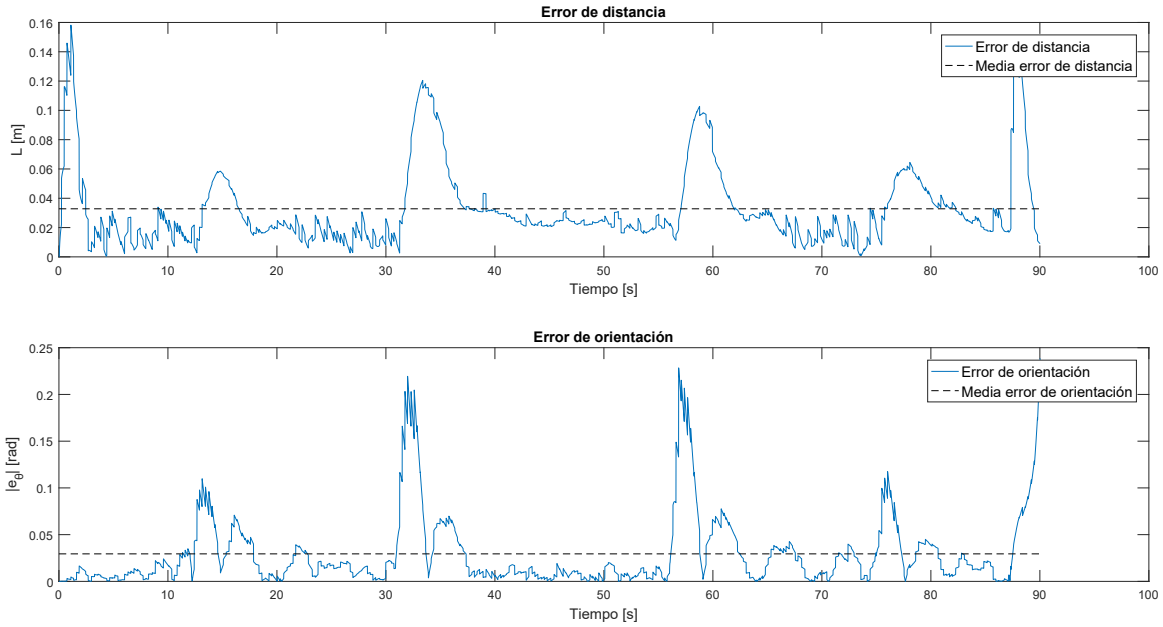

Figura 4.13: Error de distancia y orientación obtenidos del ensayo con controlador periódico y estimación basada en tiempo (periódica).

Para determinar el error que se ha tenido en la medida como la estimación, en la figura 4.14 y 4.15 se representa el error de medida así como de estimación con respecto a la posición de referencia. Se ha representado también la media de dicho error, en el caso del error de medida, la media se establece en 0.02461m para X y en 0.01747m para Y. En el caso de la estimación, se encuentra reflejado este error de medida ya que la estimación se realiza a partir de la medida, situando la media en 0.02321m para X y en 0.01721m para Y.

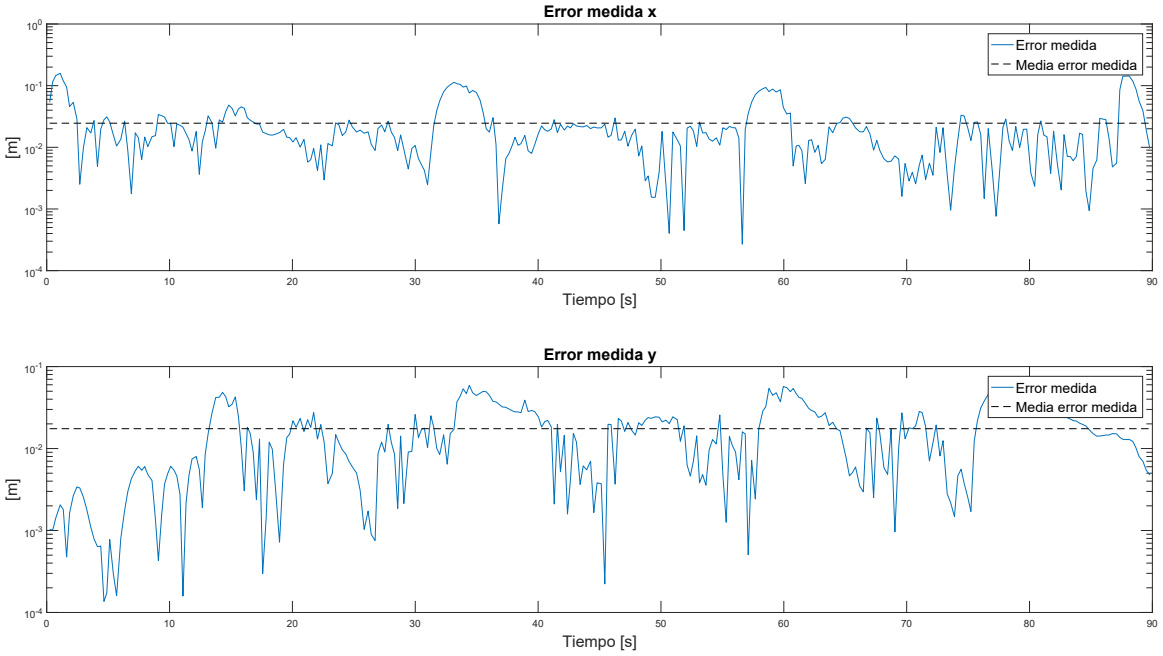

Figura 4.14: Error de medida obtenido del ensayo con controlador periódico y estimación basada en tiempo (periódica).

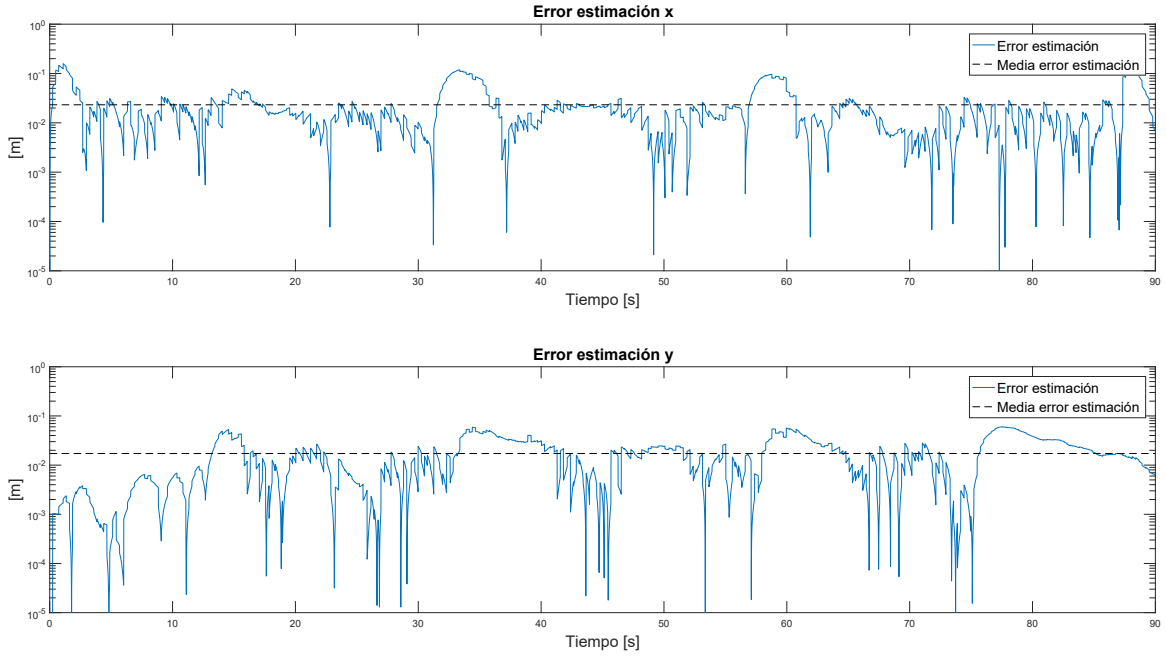

Figura 4.15: Error de estimación obtenido del ensayo con controlador periódico y estimación basada en tiempo (periódica).

Si observamos la evolución de la DRMS y el error de orientación, mostrados en la figura 4.16, se puede comprobar como cada vez que se obtiene una nueva medida el valor de la DRMS se reduce y crece hasta que se obtiene el valor de una nueva medida.

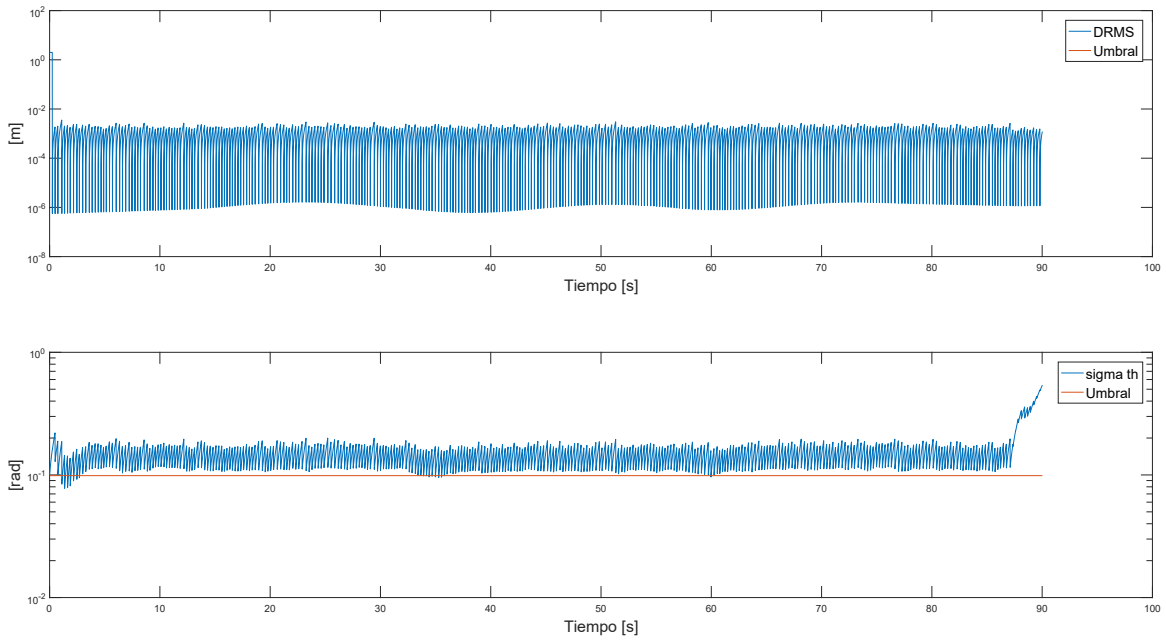

Figura 4.16: Evolución de la DRMS (superior) y el error de orientación (inferior) obtenida del ensayo con controlador periódico y estimación basada en tiempo (periódica).

4.3.2 Experimentación de control periódico junto con sensado basado en covarianza

Para reducir el número de comunicaciones que se puedan producir entre el sensor y el estimador implementado en el centro remoto, en el capítulo anterior se ha introducido a una solución que permite sólo enviar la medida sólo cuando la covarianza de la medida supere un determinado umbral.

En este ensayo se ha implementado un controlador periódico como en el caso anterior, pero en este caso el EBSE se ha implementado um umbral adaptativo que va a permitir reducir el número de comunicaciones entre el sensor y el centro remoto.

En la figura 4.17 se muestra la trayectoria del ensayo realizado, al igual que antes, se representa con un círculo el valor de la medida obtenida por el sensor. Se puede comprobar que la trayectoria estimada por parte del EBSE realiza de forma correctamente la trayectoria de referencia, pero a diferencia del caso anterior el número de medidas es inferior al caso anterior donde no se aplica ningún tipo de umbral adaptativo.

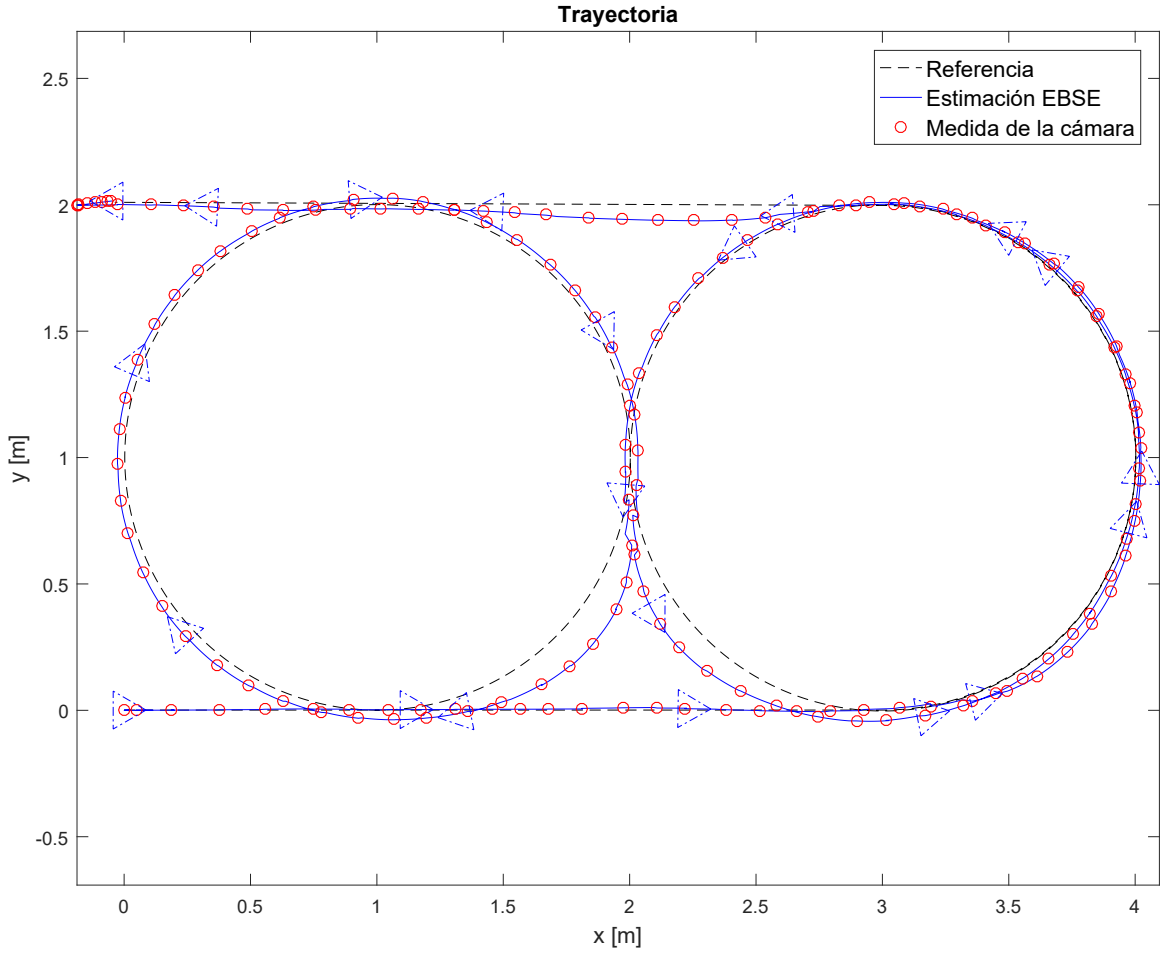

Figura 4.17: Trayectoria obtenida del ensayo con controlador periódico y estimación basada en eventos (aperiódica).

En cuanto al seguimiento se podría decir que lo realiza correctamente, como en el caso anterior. En la figura 4.18 se muestra la evolución de la posición x e y, como se puede comprobar que la trayectoria la realiza correctamente.

En cuanto a las consignas de velocidad lineal y angular, en la figura 4.19 se muestra la evolución de

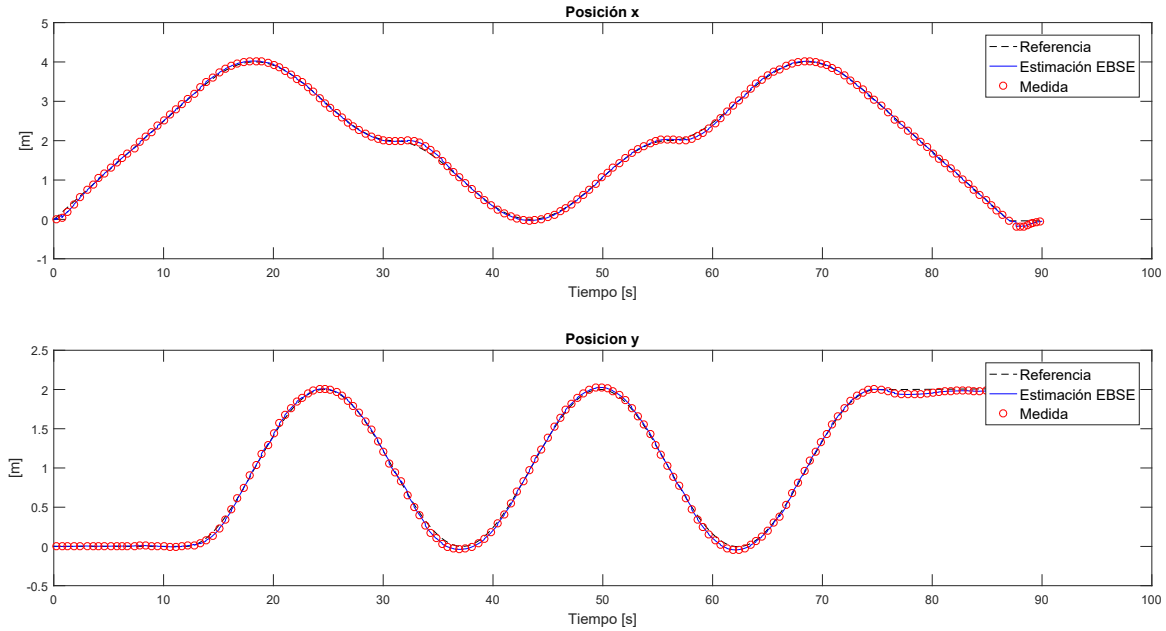

Figura 4.18: Posición x e y obtenidas del ensayo con controlador periódico y estimación basada en eventos (aperiódica).

las consignas de velocidad lineal y angular a lo largo del ensayo. En este caso, no hay gran diferencia con las obtenidas en el caso anterior, ya que el controlador se comporta de la misma forma y los errores de medida y estimación son relativamente parecidos.

En la figura 4.20 muestra la evolución del error de distancia, así como el error de orientación. Se puede observar que donde en la trayectoria el robot se ha desviado ligeramente de la trayectoria, se obtienen valores mayores de error de distancia y orientación. De media el error de distancia se sitúa en 3,6cm, mientras que el error de orientación son aproximadamente 0,03 radianes de media, que equivale a $1{,}72^o$.

Si evaluamos la diferencia que existe en la posición respecto con la referencia, en la figura 4.21 se muestra la evolución del error de medida. Al igual que antes, se ha establecido la media del error de medida, obteniendo para x una media del error de medida de 0.0288m, mientra que para y este valor es de 0.02044m. Se puede comprobar que es ligeramente superior al caso anterior que no estaba establecido el umbral adaptativo, habiendo una diferencia del rango de milímetros.

En cuanto al error de estimación, ocurre como con el error de medida, es ligeramente superior al caso donde no se implementaba el umbral adaptativo, situando la media del error de estimación para x en 0.0249m y para y la media se sitúa en 0.02024m.

Por último, la figura 4.23 representa la evolución de la DRMS y el error de orientación, en este caso se puede comprobar cómo cuando el valor de la DRMS superar el umbral adaptativo, en ese instante se produce una petición de medida que cuando es reducida se reduce el valor de DRMS. Hay que aclarar que la DRMS debería mantenerse por debajo del umbral en todo momento, pero debido a que el tiempo de procesamiento es variable y puede depender de la iluminación del espacio, provoca que desde el instante que se supera el umbral y se realiza la petición de medida al sensor, hasta que el sensor manda al centro remoto la medida, la DRMS sigue creciendo.

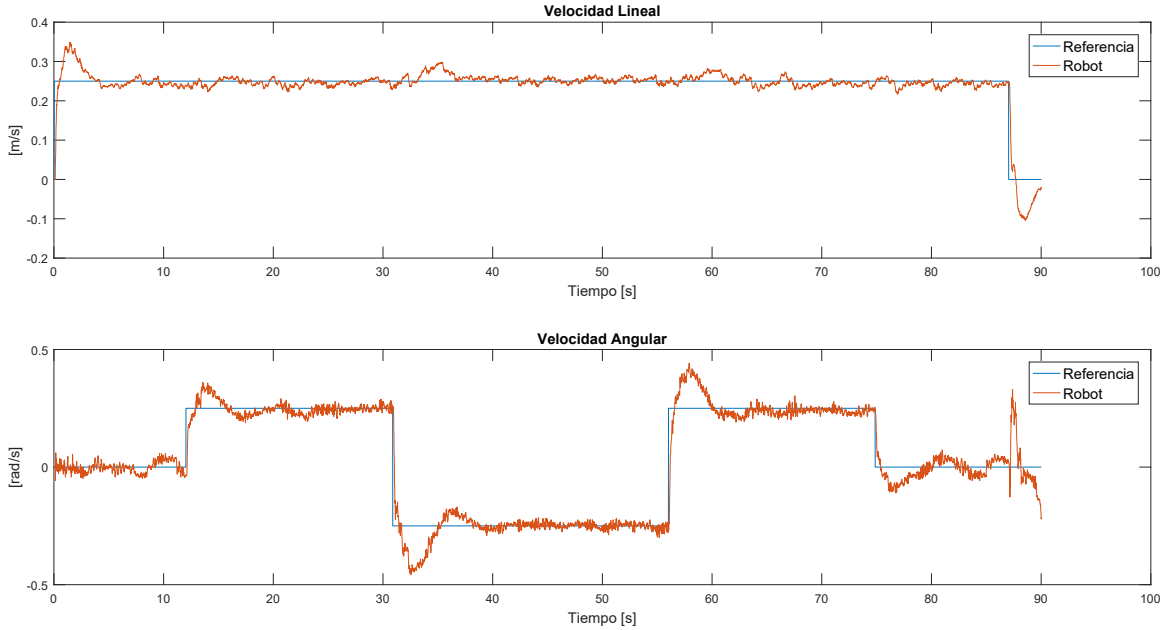

Figura 4.19: Velocidad lineal y angular obtenidas del ensayo con controlador periódico y estimación basada en eventos (aperiódica).

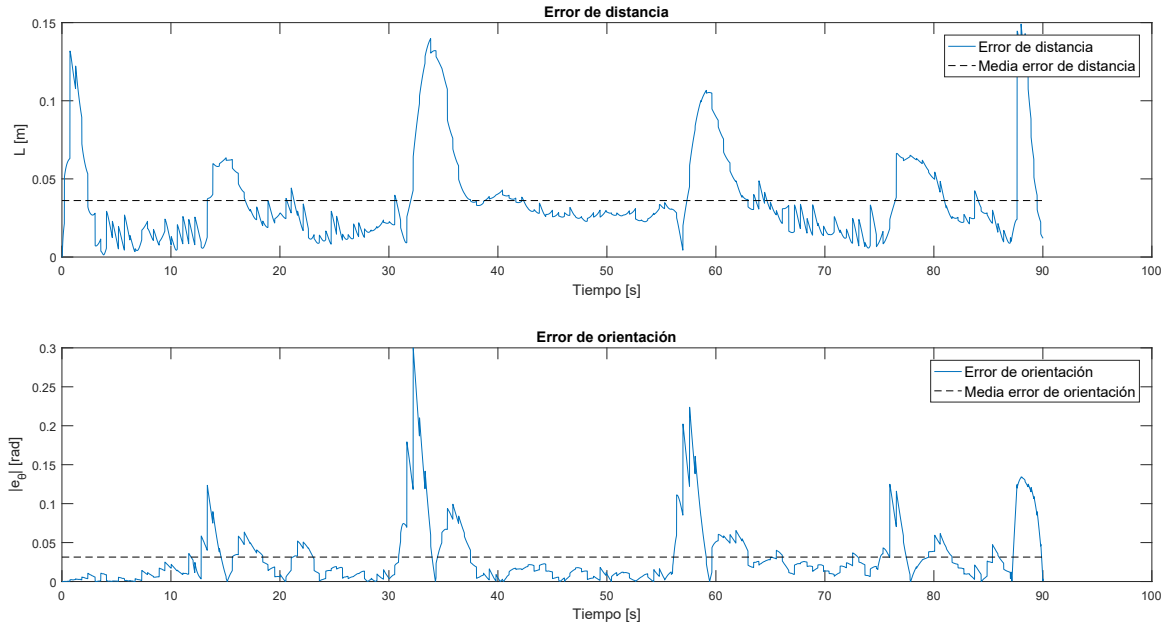

Figura 4.20: Error de distancia y orientiación obtenidos del ensayo con controlador periódico y estimación basada en eventos (aperiódica).

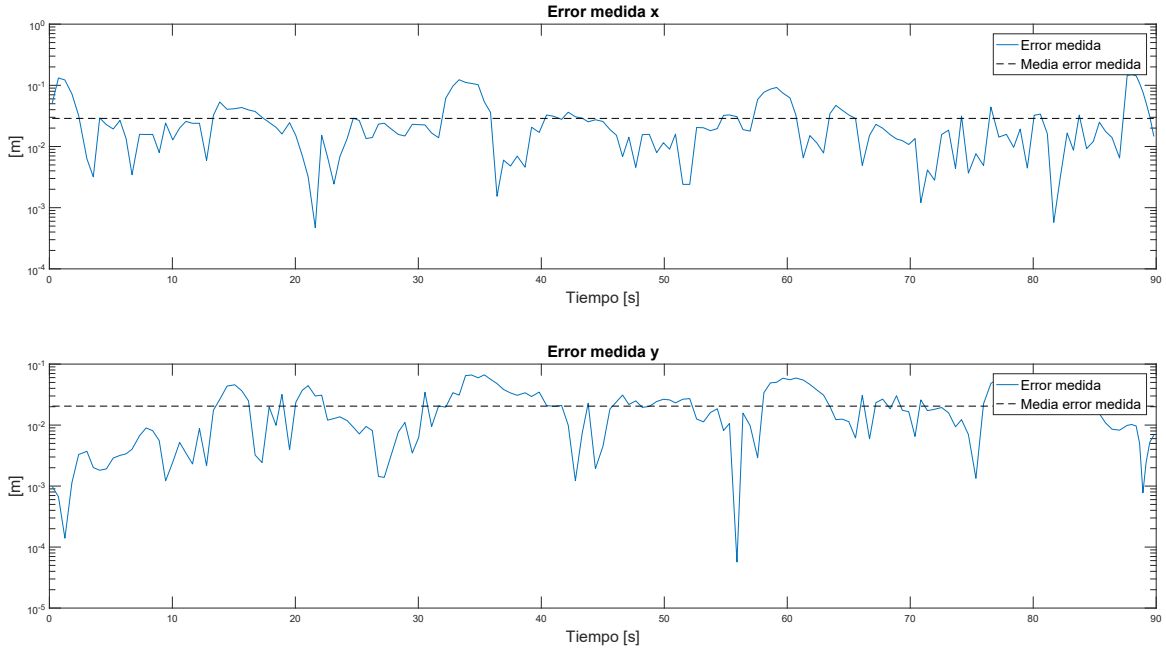

Figura 4.21: Error de medida obtenido del ensayo con controlador periódico y estimación basada en eventos (aperiódica).

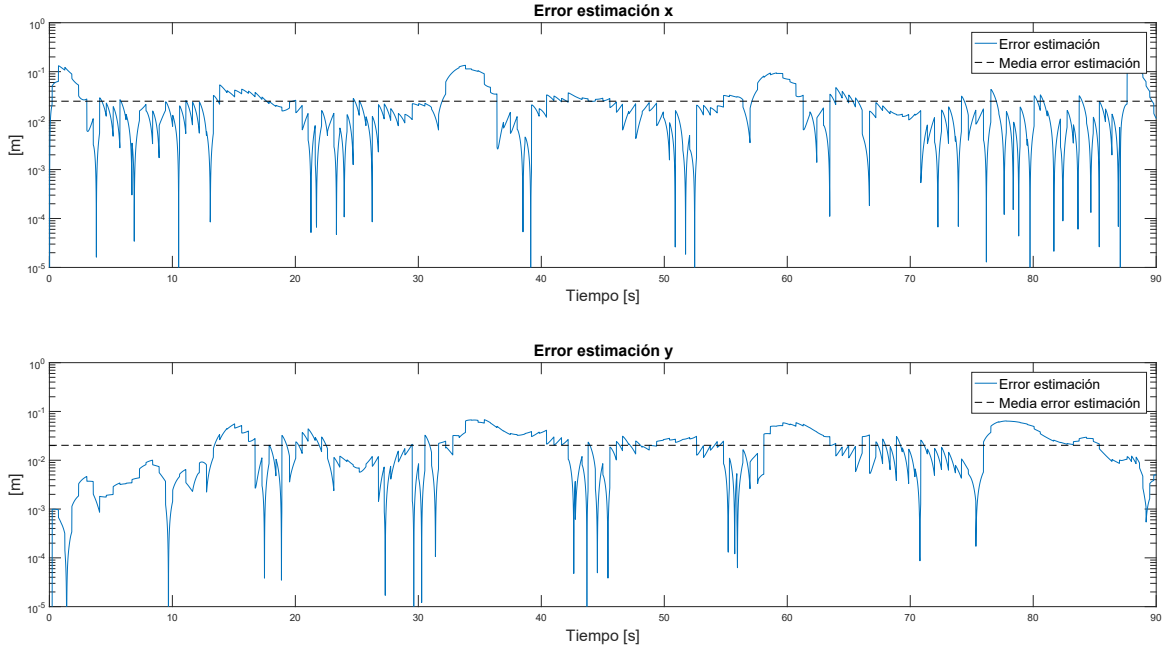

Figura 4.22: Error de estimación obtenido del ensayo con controlador periódico y estimación basada en eventos (aperiódica).

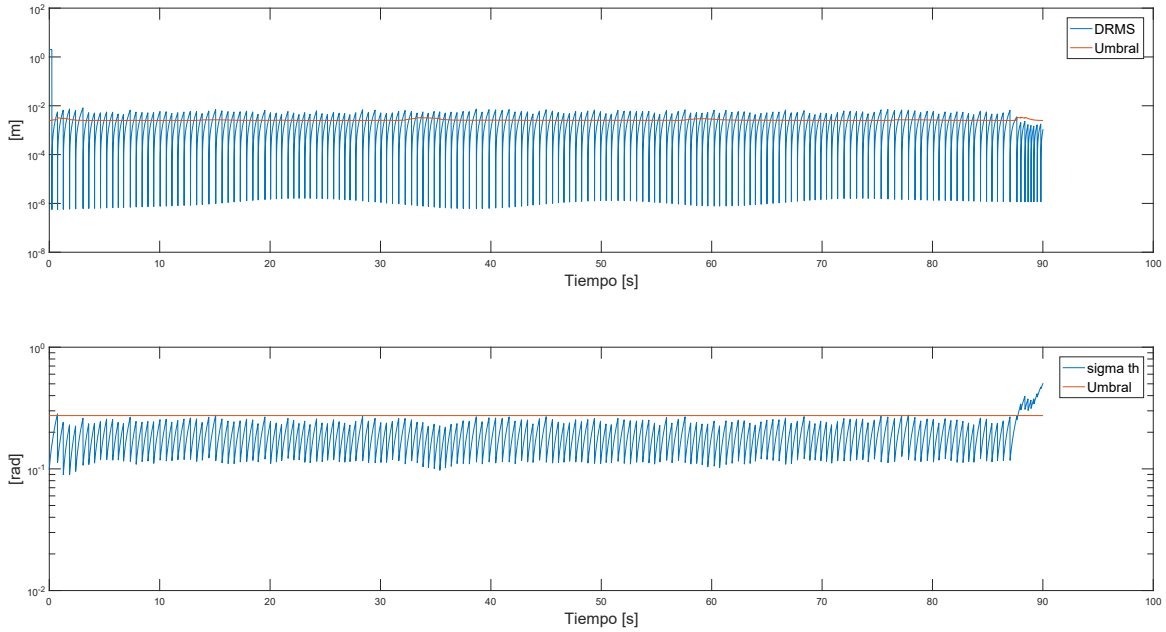

Figura 4.23: Evolución de la DRMS (superior) y el error de orientación (inferior) obtenidos del ensayo con controlador periódico y estimación basada en eventos (aperiódica).

A lo largo de esta sección se ha comprobado como el umbral adaptativo permite reducir el número de comunicaciones entre los sensores y el centro remoto haciendo que la trayectoria no se vea afectada.

4.3.3 Experimentación de control *event-triggered* junto con sensado periódico

En los dos apartados anteriores, se ha visto como es posible reducir el número de comunicaciones entre el centro remoto y el sensor. En este apartado se va a tratar de reducir el número de comunicaciones entre el centro remoto y el robot, comparando los resultados con los recogidos en la sección anterior 4.3.1, donde se ha implementado el controlador LBC periódico con realimentación de pose obtenido de la estimación del EBSE.

En la figura 4.24 se muestra la trayectoria obtenida realizada en el ensayo, donde se puede comprobar que la trayectoria se realiza de forma correcta, pese a que hay zonas donde se aleja ligeramente de la trayectoria referencia. Si comparamos con la trayectoria obtenida en 4.3.1 (figura 4.10) se puede comprobar como los puntos donde se aleja de la trayectoria de referencia son los mismo lugares que en ensayo actual.

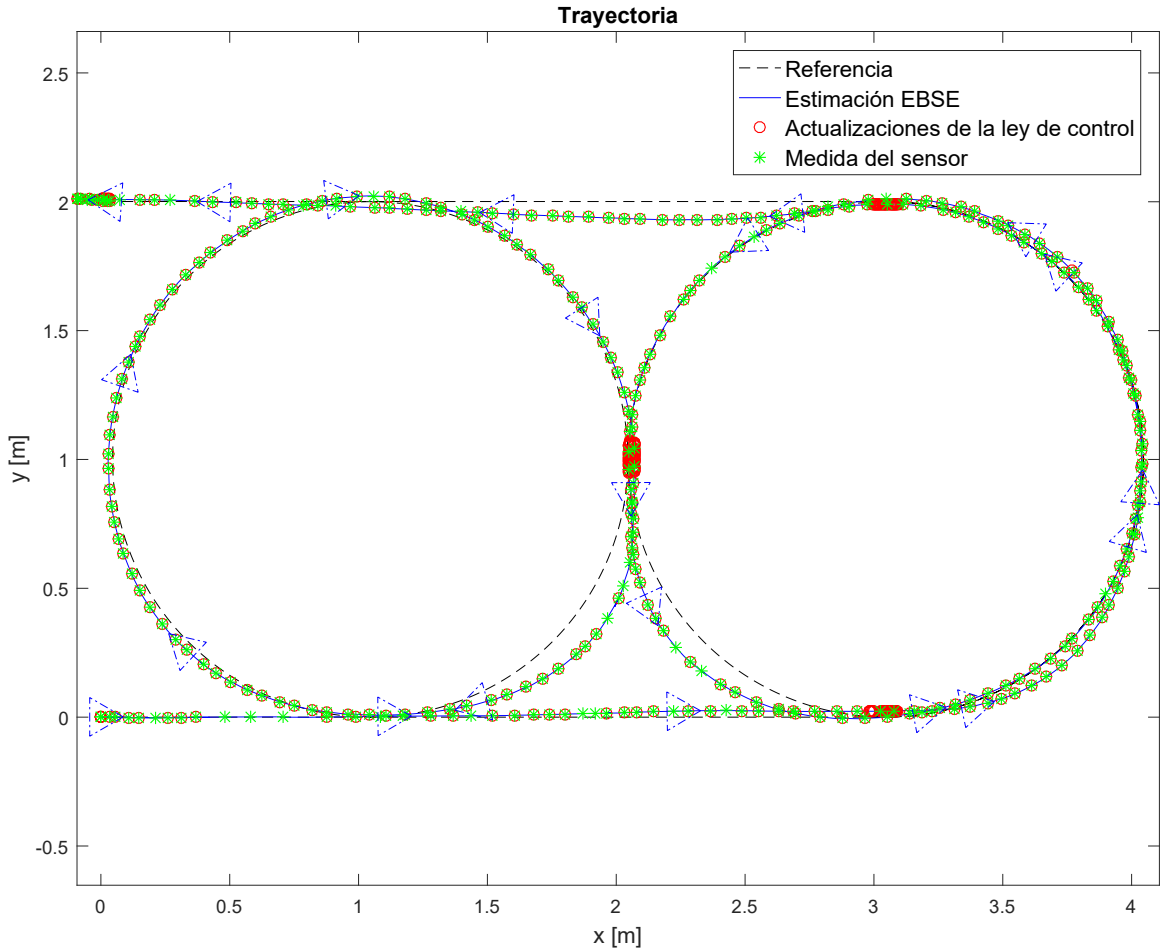

Figura 4.24: Trayectoria obtenida del ensayo con controlador aperiódico y estimación basada en tiempo (periódica).

En la figura 4.25 se muestra la evolución de la posición x e y a lo largo del ensayo. Comparando con la posición de referencia, no hay una variación que se deba destacar, de forma que es mejor analizar en detalle el error de distancia y orientación.

En cuanto a las consignas de velocidad lineal y angular que se envían a la unidad móvil, en la figura 4.26 se muestran las recogidas en el ensayo. En ellas se puede apreciar cómo se mantiene el valor de la consigna de velocidad hasta que se produce una nueva actualización por parte del centro remoto.

Analizando el los errores de distancia y orientación es posible alcanzar una idea más clara de que ambos resultados se asemejan bastante. En la figura 4.27, se muestra la evolución del error de distancia y orientación, obteniendo resultados de media ligeramente superiores que en el caso anterior (figura fig:ErrorDistYOrientPEPE). Los errores que son superiores al valor medio obtenido, coinciden con las zonas donde el robot se alejaba ligeramente de la trayectoria de referencia.

En este caso el EBSE se ha ejecutado con una estimación basa en tiempo, de forma que no se ha aplicado ningún umbral para obtener una nueva medida. La figura 4.28 se muestra el error de medida obtenido comparado con la trayectoria de referencia.

En cuanto al error que se haya producido en la estimación, la figura 4.29 muestra dicho error comparado con la trayectoria de referencia.

En cuanto a la DRMS y el error de orientación obtenidos por el sensor (cámara) en la figura 4.30 se muestra la evolución de las mismas a lo largo del tiempo. En este caso, al no existir un umbral el valor

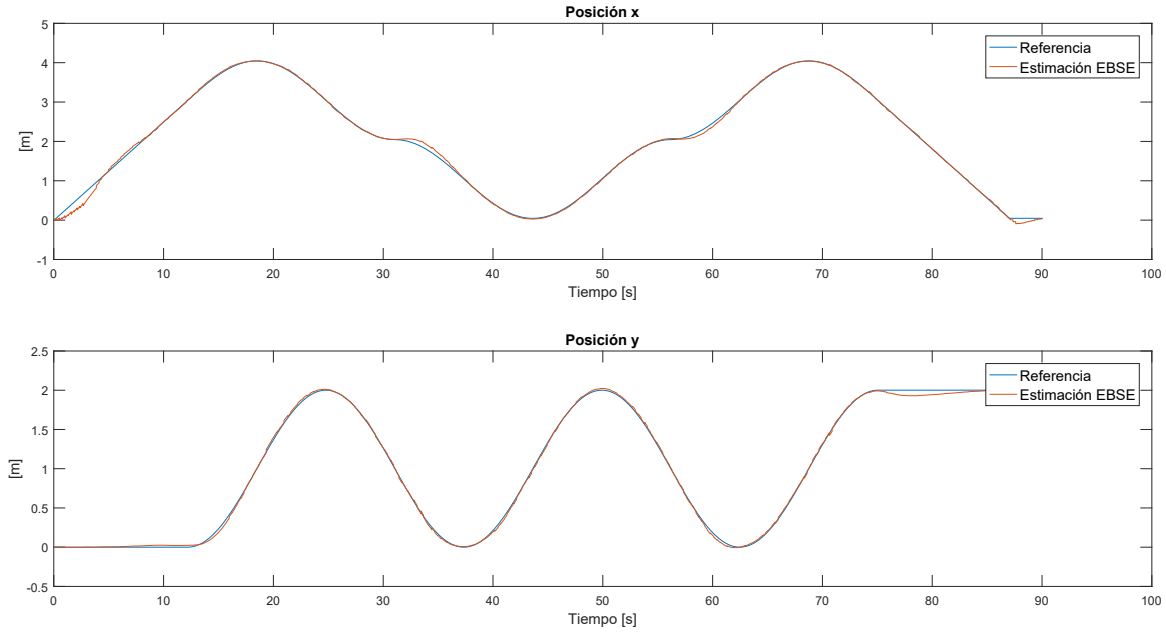

Figura 4.25: Posición x e y obtenidas del ensayo con controlador aperiódico y estimación basada en tiempo (periódica).

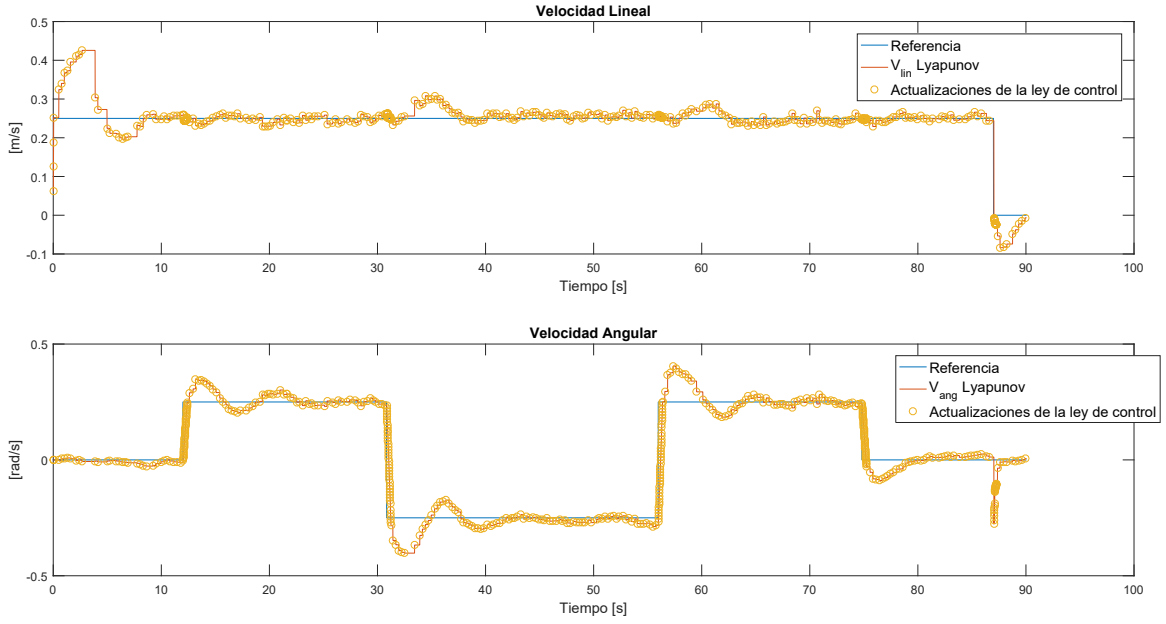

Figura 4.26: Consignas de Velocidad lineal y angular obtenidas del ensayo con controlador aperiódico y estimación basada en tiempo (periódica).

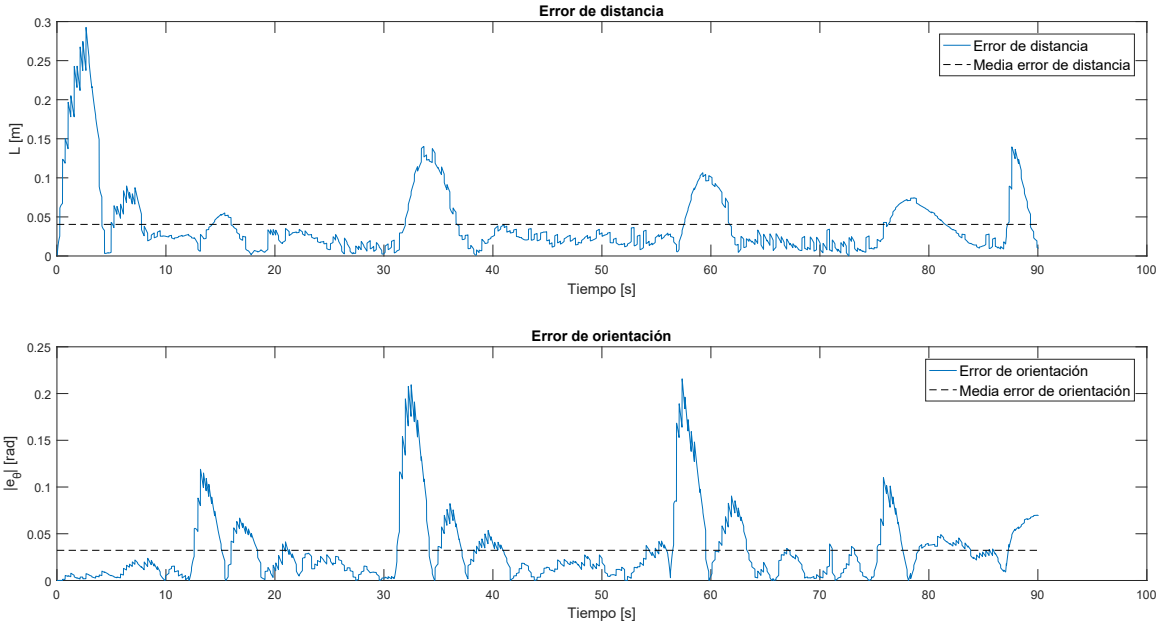

Figura 4.27: Error de distancia y orientiación obtenidos del ensayo con controlador aperiódico y estimación basada en tiempo (periódica).

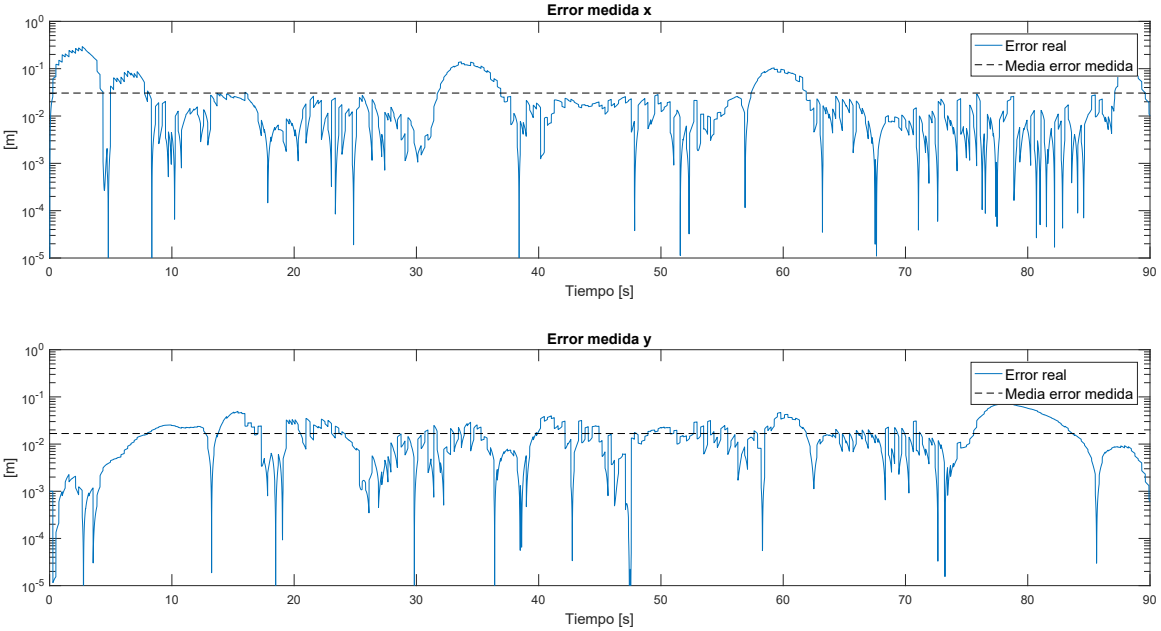

Figura 4.28: Error de medida obtenido del ensayo con controlador aperiódico y estimación basada en tiempo (periódica).

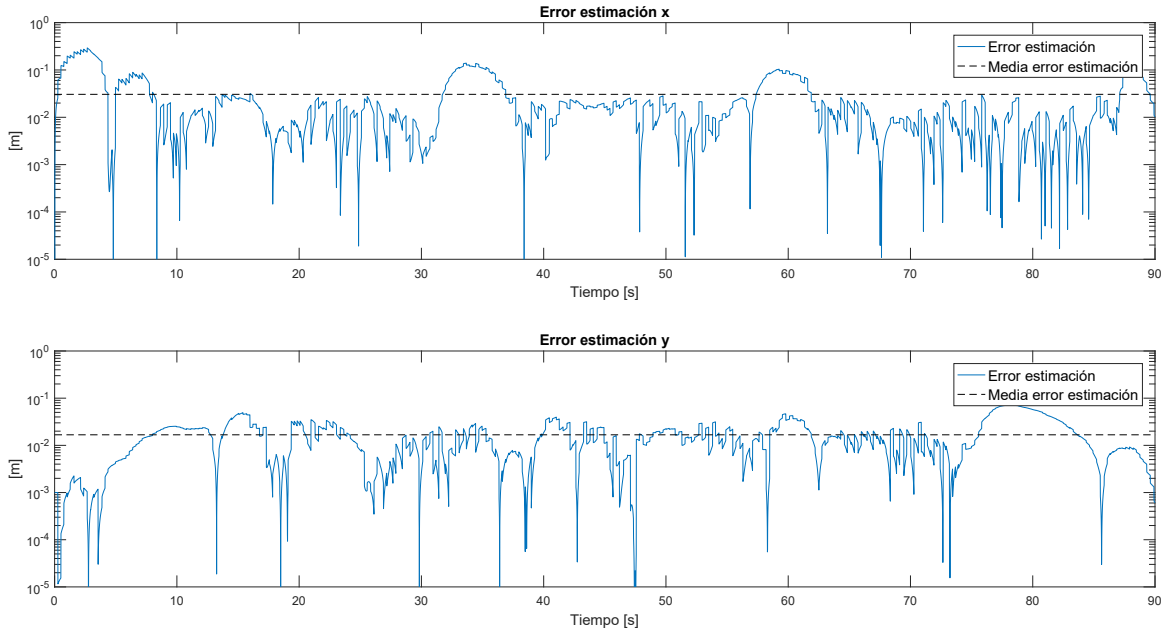

Figura 4.29: Error de estimación obtenido del ensayo con controlador aperiódico y estimación basada en tiempo (periódica).

de la DRMS me mantiene estable, ya que cada vez que se obtiene una medida el valor de DRMS decrece.

En este apartado, se ha comparado el funcionamiento del controlador *event-triggered* con el controlador LBC periódico con la realimentación de pose obtenida del estimador de estados basado en tiempo. Se ha podido ver que la trayectoria la realizaba sin grandes diferencias con respecto al caso periódico, además los errores de distancia y orientación así lo validan.

4.3.4 Experimentación de control *event-triggered* junto con sensado basado en covarianza

En este último apartado se va a tratar de analizar los resultados obtenidos de la propuesta de control aperiódico juntos con el sensado aperiódico basado en covarianza. Los resultados anteriores van a servir de referencia para poder comparar esta propuesta, analizando el comportamiento a lo largo de la trayectoria, las consignas de velocidad lineal y angular y los errores en la parte de estimación, distancia y orientación.

Con respecto a la trayectoria obtenida del apartado 4.3.2 (figura 4.17), en la figura 4.31, que representa la trayectoria obtenida para este ensayo donde el controlados está basado en un *event-triggered* y la realimentación de la pose se obtiene de un estimador de estados basado en covarianza (EBSE), ocurre como en la sección anterior, existen determinados momentos que se desvía de la trayectoria de referencia, pero comparándolo con 4.17 donde el controlador es periódico, no se pueden destacar grandes diferencias.

Si se ve en detalle la evolución de la posición x e y que ha llevado la unidad móvil junto con la referencia (ver figura 4.32), se puede comprobar que se producen tales desvíos con respecto a la posición de referencia, que se corresponden con las zonas donde la unidad móvil se alejaba de la trayectoria de referencia.

En cuanto a las consginas de velocidad lineal y angular, la figura 4.33 muestra las consignas de velocidad lineal y angular obtenidas en el ensayo.

Para cuantificar dicho error de distancia y orientación, en la figura 4.34 permite evaluar dichos errores. Cabe destacar que el error de distancia es ligeramente superior al obtenido en para el controlador periódico

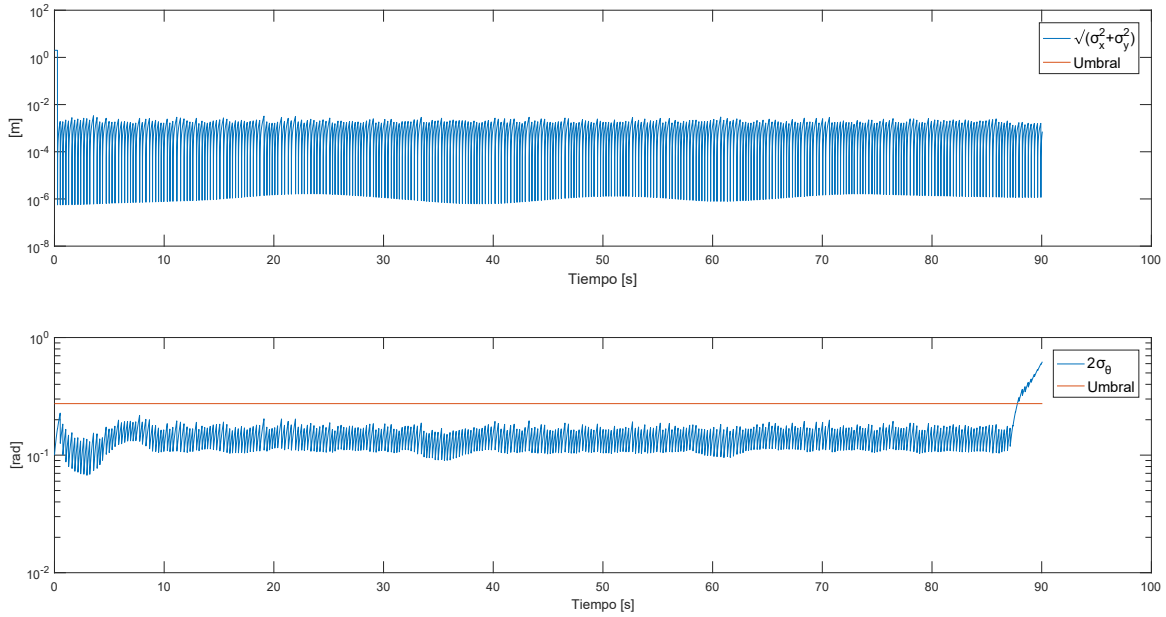

Figura 4.30: Evolución de la DRMS (superior) y el error de orientación (inferior) obtenidas del ensayo con controlador aperiódico y estimación basada en tiempo (periódica).

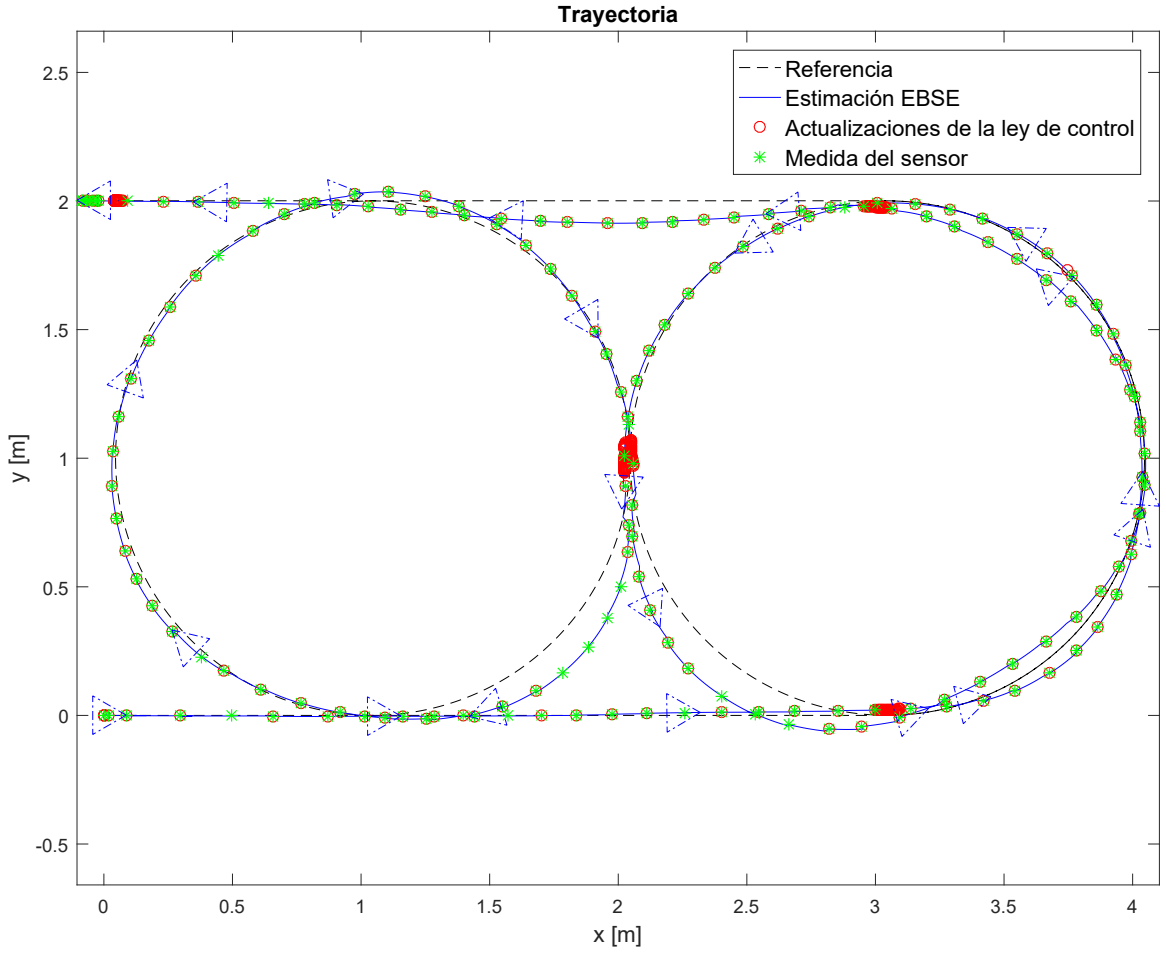

Figura 4.31: Trayectoria obtenida del ensayo con controlador aperiódico y estimación basada en eventos (aperiódica).

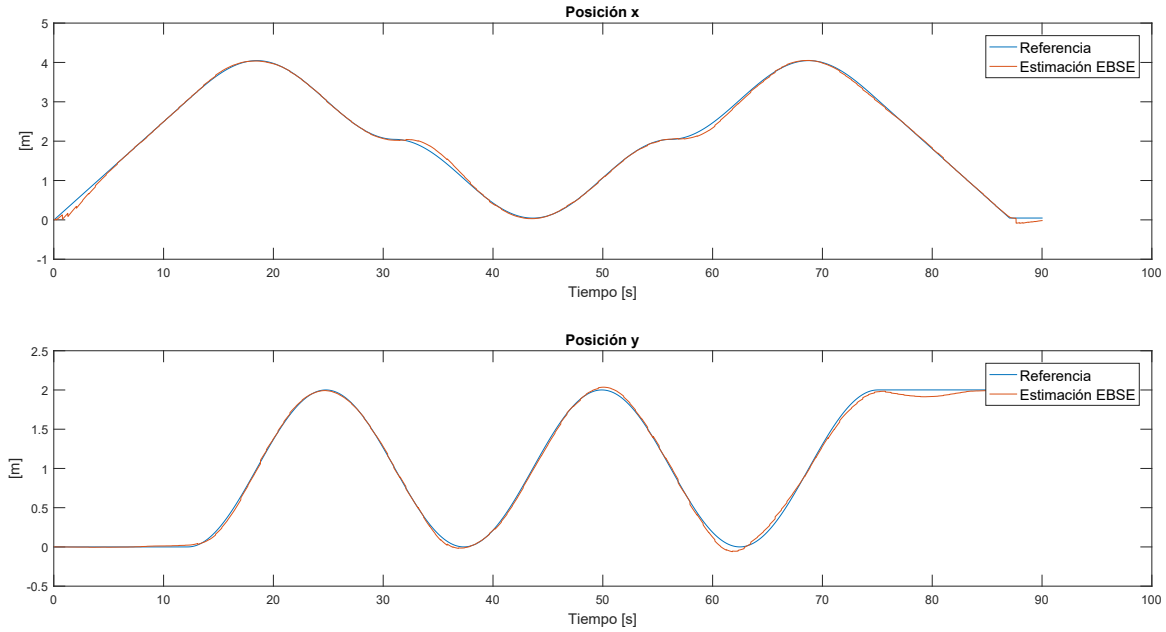

Figura 4.32: Posición x e y obtenidas del ensayo con controlador aperiódico y estimación basada en eventos (aperiódica).

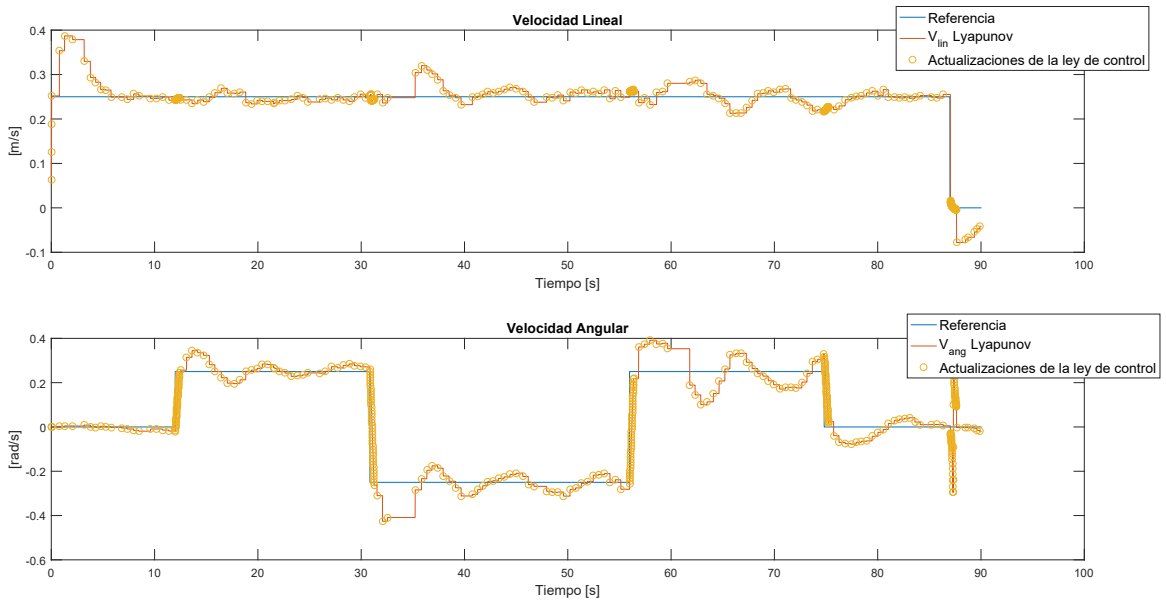

Figura 4.33: Consignas de Velocidad lineal y angular obtenidas del ensayo con controlador aperiódico y estimación basada en eventos (aperiódica).

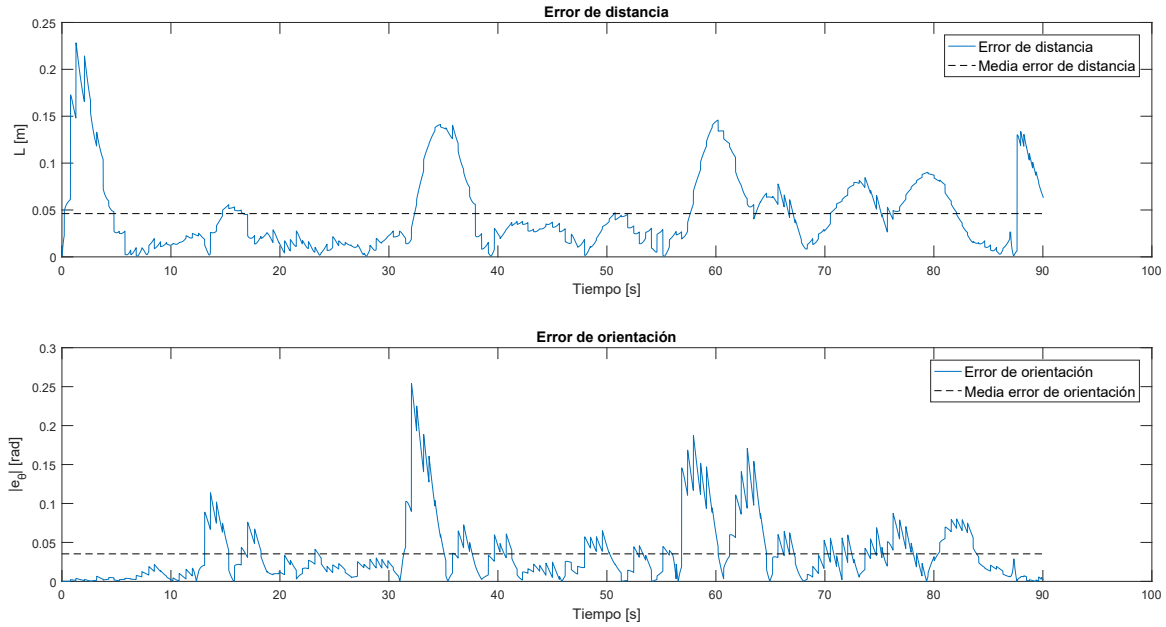

Figura 4.34: Error de distancia y orientiación obtenidos del ensayo con controlador aperiódico y estimación basada en eventos (aperiódica).

(ver figura 4.20), al igual que ocurre con el error de orientación.

En la figura 4.35 se muestra el error de medida obtenido por la cámara respecto a la referencia. En la figura 4.36 se muestra el error de estimación que se a producido a lo largo del ensayo. Se puede comparar con la figura 4.22 los errores están en el mismo rango de magnitud.

Por último, en la figura 4.37 se muestra la evolución de la DRMS y error de orientación obtenidas por el EBSE, se puede comprobar que el error de orientación está en todo momento por debajo del umbral pero, al igual que ocurría en la figura 4.23 anterior, debido a que la medida no se obtiene de forma inmediata y pasa un tiempo de procesamiento, se mantiene ligeramente por encima del umbral.

4.4 Conclusiones

A lo largo del capítulo se han mostrados los resultados realizados en el demostrador, para cada uno de los casos:

- LBC periódico junto con el EBSE periódico.
- LBC periódico junto con el EBSE umbral adaptativo.
- LBC aperiódico junto con el EBSE periódico.
- LBC aperiódico junto con el EBSE umbral adaptativo.

De los ensayos realizados en el demostrador, pese a que se cambia bien el controlador o el estimador, se obtienen resultados similares en cuanto al seguimiento de la trayectoria y por ende en los errores. En la 4.1 se muestra un resumen de los resultados obtenidos en cada uno de los ensayos, se puede comprobar que el valor de ISE es aproximadamente un 50 % superior para la propuesta de controlador aperiódico y EBSE aperiódico respecto al controlador periódico y estimador periódico. En cuanto al número de medidas realizadas por el sensor se reducen aproximadamente a la mitad para la propuesta realizada.

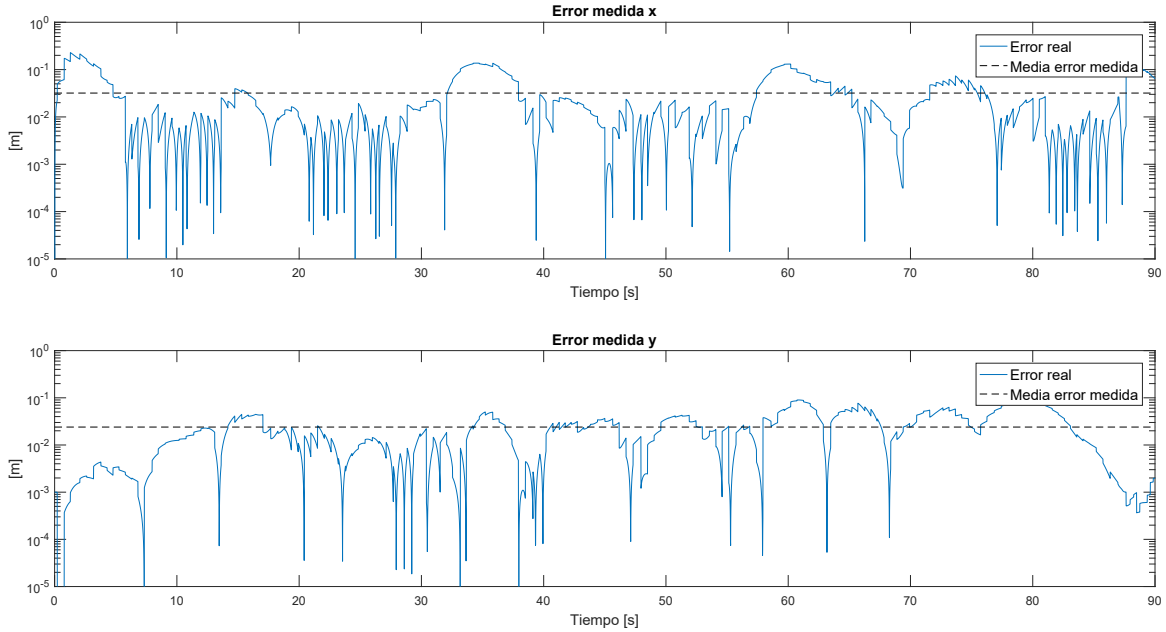

Figura 4.35: Error de medida obtenido del ensayo con controlador aperiódico y estimación basada en eventos (aperiódica).

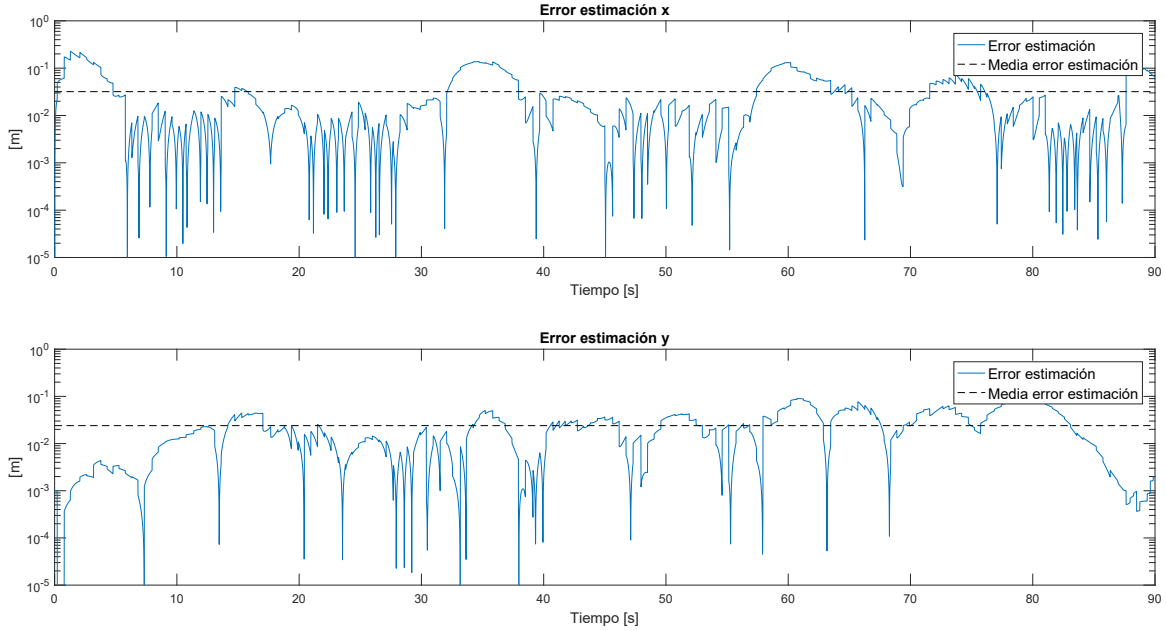

Figura 4.36: Error de estimación obtenido del ensayo con controlador aperiódico y estimación basada en eventos (aperiódica).

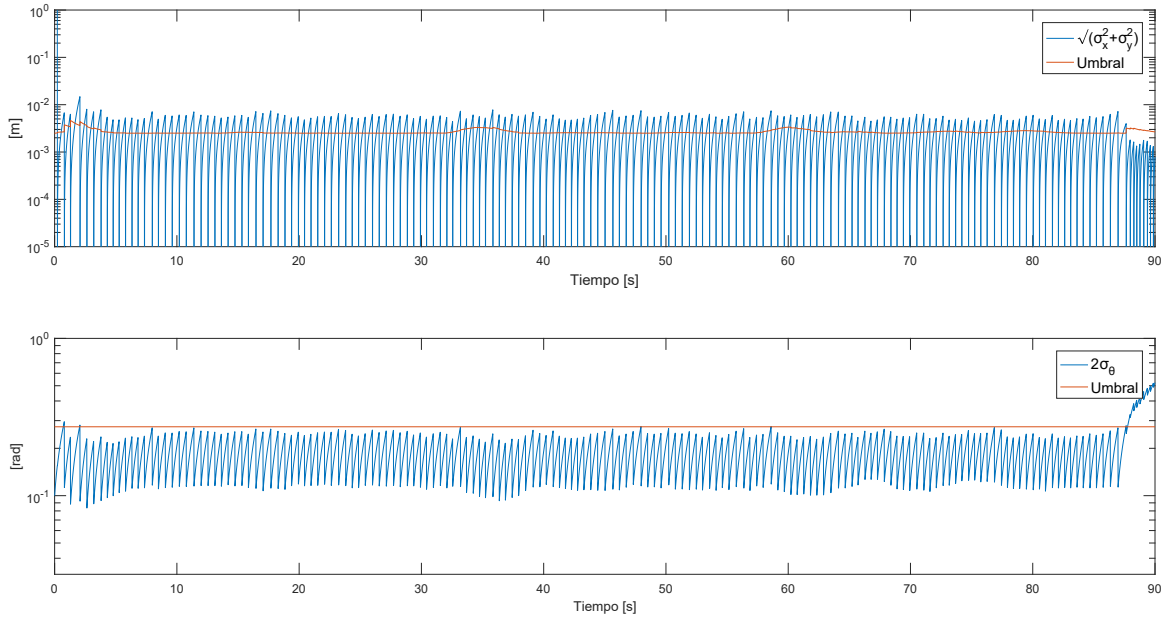

Figura 4.37: Evolución de la DRMS (superior) y el error de orientación (inferior) obtenidas del ensayo con controlador aperiódico y estimación basada en eventos (aperiódica).

Pero donde más relevancia tiene la propuesta que se ha realizado es en el número de actualizaciones de las consignas de velocidad lineal y angular que se reduce considerablemente obteniendo 373 actualizaciones que se trata de un 4.2 % de las actualizaciones obtenidas para el caso periódico, si comparamos el número total de accesos al canal de comunicaciones la reducción en el número de accesos es mayor, ya que se pasan de 18000 accesos en el caso periódico a los 746 para el caso aperiódico. Por último, comparando el error de distancia y orientación respecto a la trayectoria de referencia en la propuesta tan sólo es levemente superior que a la propuesta periódica, diferenciándose en un rango de magnitud de centímetros.

Parámetro	Control Per. y EBSE Per.	Control Per. y EBSE Aper.	Control Aper. y EBSE Per.	Control Aper. y EBSE Aper.
ISE	0.37746	0.41696	0.53762	0.59537
Nº de medidas	329 (*)	168 (*)	327 (*)	168 (*)
Nº de actualizaciones	9000 (*)	9000 (*)	483 (*)	373 (*)
Med. Error Distancia [m]	0.0328	0.0361	0.04043	0.0461
Med. Error Orientación [rad]	0.0294	0.0314	0.0323	0.0353

Tabla 4.1: Tabla resumen de resultados experimentales.
(*) El enlace de comunicación es bidireccional, por tanto el número de accesos al canal es el doble.

Si se comparan los resultados recogidos en la tabla 4.1 con los obtenidos en la simulación (ver tabla 3.3), se puede observar que en la implementación se obtienen mejores valores de ISE y errores que los obtenidos en la simulación. Esto se debe fundamentalmente a que en la simulación, con el modelo de planta aplicado, resultan menos accesos al canal (disparos) y, por tanto, valores de comportamiento (ISE, errores) ligeramente peores. Además, en la simulación la matriz **P** evoluciona más lentamente que lo hace en la implementación, , debido a errores de modelado del ruido de medida y de estado. Además, los retardos que se obtienen en la implementación desde que se realiza la petición de medida, hasta que

el sensor la proporciona no se reflejan en la simulación, todo esto hace que el número de medidas sea inferior en la simulación que en los resultados de la implementación.

Capítulo 5

Conclusiones y líneas futuras

En este apartado se resumen las conclusiones obtenidas y se proponen futuras líneas de investigación que se deriven del trabajo.

5.1 Conclusiones

A lo largo de todo el documento se ha trabajado en alcanzar los principales objetivos marcados en la introducción de este trabajo. A continuación se van a repasar cada uno de ellos haciendo referencia a los resultados obtenidos. A continuación se enumeran cada uno de los objetivos propuestos.

1. Propuesta de mecanismo de disparo de los eventos producidos en el lazo de control.
2. Propuesta de mecanismo de disparo para la estimación (corrección de la pose) a partir de las medidas recogidas por un módulo sensorial externo al robot, que en este caso es una cámara.
3. Validación mediante simulación de la estrategia comparándola con una solución periódica.
4. Validación de la propuesta mediante un demostrador de laboratorio utilizando un robot P3-DX, una cámara Kinect 2.0 y un PC como controlador, todos ellos conectados a través de una red WiFi.

A lo largo del capítulo 3.4 se ha realizado una simulación para obtener un resultado válido previo a la implementación que permitiera proporcionar de un mecanismo de disparo en el lazo de control. Esta solución, pasa por evaluar la función de Lyapunov evitando que se vuelva creciente para mantener así la estabilidad sel sistema.

Esta propuesta permitía reducir el número de comunicaciones que se producen entre el centro remoto y la unidad móvil, respecto a la solución periódica donde el número de accesos al canal de comunicaciones ascendía 9000 para la trayectoria de referencia dada, con el mecanismo de disparo se reducía al 5 % el número de accesos al canal de comunicaciones con respecto a la solución periódica.

Se ha comprobado mediante la simulación que la propuesta era válida, por tanto era posible realizar la implementación en el demostrador para comprobar que los resultados obtenidos en el simulador son válidos en un sistema real. En el capítulo 4.3.3 se ha comprobado que los resultados obtenidos validan la propuesta de mecanismo de disparo y además, se ha comprobado que los resultados obtenidos son muy parecidos a los obtenidos en el capítulo 4.3.1 donde se implementaba un controlador periódico.

En los ensayos mencionados anteriormente, se tenía un módulo sensorial externo al robot que realiza la medida de la posición de la unidad móvil y se la hace llegar al centro remoto haciendo uso del canal

de comunicaciones inalámbrico de forma periódica, es decir cada vez que se obtiene una medida se envía al centro remoto.

Para tratar de reducir el número de accesos al canal de comunicaciones entre el centro remoto y unidad sensora, en el capítulo 3.4 se ha propuesto el mecanismo de disparo que utiliza las medidas obtenidas por la cámara para generar el mismo. En el capítulo 3.3.1.2 se ha implementado mediante la simulación, el mecanismo de disparo, comprobando que los resultados son correctos y con respecto al caso periódico, los resultados son semejantes y claramente el número de comunicaciones entre el centro remoto y el sensor se reduce aproximadamente a la mitad. Esto lo corrobora el ensayo llevado a cabo en el demostrador y cuyos resultados se recogen en el capítulo 4.3 independientemente del controlador que se esté utilizando en cada caso.

Por último, se ha realizado una propuesta que une el controlador aperiódico junto con la estimación aperiódica. Los resultados de la simulación recogidos en el capítulo 3.4 donde se ha comprobado el correcto funcionamiento de ambas propuesta unidas. Los resultados obtenidos en el demostrador, han permitido verificar la solución propuesta probada en la simulación, los resultados del ensayo llevado a cabo se han recogido en el capítulo 4.3.4.

En este caso con la propuesta del conjunto periódico el número de accesos al canal de comunicaciones se reduce de forma considerable. En la tabla 4.1 se recogen el número de medidas y actualizaciones de las consignas de la ley de control, que son equivalente al número de accesos al canal de comunicaciones. Para el caso periódico de obtienen 9329 accesos entre los obtenidos por el controlador como por la unidad sensora, mientras que para el caso de fusión de control y estimación aperiódico propuesto el número de accesos es de 541 teniendo en cuenta los del controlador y del sensor. Se puede comprobar como el número de accesos se ha reducido considerablemente, dado que la propuesta de fusión solo alcanza el 5.8 % de accesos con respecto a la solución puramente periódica.

5.2 Líneas futuras

A continuación se van a enumerar una serie de líneas futuras que se pueden realizar a partir de este trabajo.

- **Utilización de canal inalámbrico WiFi para la comunicación con todos los sensores del demostrador**.

  Dado que en los ensayos llevados a cabo en este trabajo han hecho uso de la conexión cableada entre el centro remoto y el mini PC de la unidad sensora, se propone como trabajo futuro la utilización del canal WiFi que permita la comunicación inalámbrica entre el centro remoto y la unidad sensora. De esta forma se podrían obtener los resultados para los ensayos desarrollados en este trabajo y poder así analizar las ventajas o desventajas que puedan tener el hacer uso del un enlace inalámbrico entre el centro remoto y los sensores.

- **Extender la propuesta de control aperiódico y sensado aperiódico haciendo uso de los sensores disponibles en el demostrador**.

  El demostrador consta de cinco cámaras que permiten la visualización completa del pasillo, de modo que se propone realizar esta misma propuesta de control aperiódico haciendo uso de un mayor número de sensores.

  Para ello es necesario establecer ciertos mecanismos que permitan decidir cual de los sensores proporciona la información más fiable, en el caso que la unidad móvil sea detectada por más de un sensor.

- **Extender la propuesta a varias unidades móviles**.

  Esta propuesta se puede enfocar desde dos puntos de vista. Un enfoque se basa en controlar un convoy de unidades móviles que se desplazan por el demostrador, para ello bastaría con detectar a la primera de las unidades y el resto realicen el seguimiento de la primera de las unidades. El otro enfoque se dirige dentro de la línea de trabajo colaborativo, donde cada unidad móvil tiene una tarea determinada y se deben desplazar de forma independiente, para ello es necesario que las cámaras identifiquen a cada unidad móvil y proporcionen información al centro remoto de la unidad móvil que están detectando.

  Para poder distinguir entre varias unidades, se pueden hacer uso de patrones que aporten información adicional, como son los códigos QR o bien realizar la identificación de cada unidad por la detección del patrón de un determinado color, es decir, cada patrón a detectar por las cámaras es el mismo, pero el color de cada uno está asociado a cada una de las unidades móviles. Cada solución tiene sus ventajas e inconvenientes que se proponen estudiar como trabajo futuro.

Capítulo 6

Pliego de condiciones

Para garantizar el correcto funcionamiento del proyecto se asumirá que se cumplen con las siguientes especificaciones detalladas a continuación.

6.1 Condiciones materiales y equipos

6.1.1 Equipos informáticos

Son necesarios los siguientes recursos hardware y software.

6.1.1.1 Recursos Hardware

Se debe disponer de los siguiente equipos debidamente conectados y funcionando correctamente.

- Ordenador Intel NUC con microprocesador Intel i3, 2.1 GHz con 4GB de memoria RAM o similar.
- Unidad robótica P3-DX, con el sistema empotrado embarcado desarrollado en [37].
- Router Ethernet-Wireless.
- MICROSOFT Sensor Kinect Xbox One (versión 2)

6.1.1.2 Recursos Software

Se debe disponer de las licencias de cada uno de los siguiente programas software, así como su correcta instalación.

- Sistema operativo Linux Ubuntu 12.04 LTS con RTAI instalado para el PC centro remoto.
- Sistema operativo Linux Kubuntu 14.04 LTS con librerías de OpenCV instalado para el PC de la unidad sensorial.
- MATLAB r2012a.
- Librerías libfreenect2 para el correcto funcionamiento del sensor.

6.1.2 Condiciones de Seguridad

Con el fin de evitar posibles accidentes con alguna de las unidades móviles, se recomienda:

- Realizar los ensayos supervisando en todo momento la unidad móvil o delimitando la zona de trabajo.
- Garantizar de una luminosidad adecuada en la zona de trabajo donde se esté moviendo la unidad móvil.
- Comprobar previamente los resultados obtenidos del sensor, ya que cualquier manipulación que haya podido sufrir puede alterar el funcionamiento. Se recomienda calibrar el sistema si los sensores han podido sufrir algún tipo de manipulación.

6.1.3 Condiciones de ejecución

para garantizar el correcto funcionamiento y manejo del sistema, así como para el desarrollo de nuevas aplicaciones, es necesaria la lectura y entendimiento del presente trabajo.

Capítulo 7

Presupuesto

A continuación se presenta el presupuesto necesario para el desarrollo de la propuesta de estimación y control, remotos y aperiódicos, de unidad robótica P3-DX.

7.1 Introducción

El actual presupuesto se ha realizado bajo la premisa de que se trata de un proyecto de Investigación, por tanto se excluye del visado del colegio de ingenieros y de los honorarios por la realización del mismo.

El presupuesto incluye el desarrollo y puesta en marcha del demostrador con los elementos que intervienen en el proceso de control del movimiento del robot.

Los costes asociados al personal y mano de obra se desglosan a continuación:

7.2 Materiales

El coste de los equipos necesarios para la realización del proyecto se desglosan en la tabla 7.1.

Equipo	**Precio**	**Duración (años)**	**Uso (meses)**	**Subtotal**
Ordenador Intel NUC (NUC5i3RYH)	363.91 €	3	6	60.65 €
Router inalámbrico WHR-HP-54	120.00 €	3	6	20.00 €
Unidad móvil Robot P3-DX ampliada	3,884.80 €	3	6	647.47 €
Microsoft Sensor Kinect v2	200.00 €	3	6	33.33€
			TOTAL	**761.45 €**

Tabla 7.1: Resumen de costes materiales.

7.3 Software

El coste del software necesarios para la realización del proyecto se desglosan en la tabla 7.2.

Elemento software	Precio	Duración (años)	Uso (meses)	Subtotal
Matlab r2012a	6,000.00 €	4	6	750.00 €
Sistema opertativo Linux (Ubuntu 12.04 LTS)	- €	4	6	-
Sistema opertativo Linux (Ubuntu 14.04 LTS)	- €	4	6	-
Librerías de visión artificial	- €	4	6	-
Procesador de textos LaTEX	- €	4	6	-
			TOTAL	**750.00 €**

Tabla 7.2: Resumen de coste software.

7.4 Mano de obra

El coste de la mano de obra necesarios para la realización del proyecto se desglosan en la tabla 7.3.

Mano de obra	€/h	Duración (horas)	Subtotal
Ingeniería (Planificación y diseño)	50.00 €	200	10,000.00 €
Generación de la documentación técnica	25.00 €	160	4,000.00 €
		TOTAL	**14,000.00 €**

Tabla 7.3: Resumen del coste de mano de obra.

7.5 Coste de ejecución por contrata

El coste de ejecución por contrata para la realización del proyecto se desglosan en la tabla 7.4.

Ejecución por contrata	Coste
Coste total de los equipos	761.45 €
Coste del software	750.00 €
Coste de mano de obra	14,000.00 €
Gastos generales y beneficio industial (3 %)	775.15 €
Total Ejecución por contrata	**16,286.60 €**
IVA (21 %)	3,420.19 €
Importe total del proyecto	**19,706.79 €**

Tabla 7.4: resumen de coste de ejecución por contrata

Bibliografía

[1] 2016. [Online]. Available: https://es.wikipedia.org/wiki/Entorno_inteligente

[2] C. Santos Pérez, "Control self-triggered para seguimiento de trayectorias no lineales de robots con adaptación al retardo del canal," Ph.D. dissertation, Universidad de Alcalá, 2016.

[3] C. Santos, M. Mazo, and F. Espinosa, "Adaptive self-triggered control of a remotely operated p3-dx robot: Simulation and experimentation," *Robotics and Autonomous Systems*, vol. 62, no. 6, pp. 847–854, 2014.

[4] P. Tabuada, "Event-triggered real-time scheduling of stabilizing control tasks," *IEEE Transactions on Automatic Control*, vol. 52, no. 9, pp. 1680–1685, 2007.

[5] M. Mazo Jr and P. Tabuada, "Input-to-state stability of self-triggered control systems." in *CDC*. Citeseer, 2009, pp. 928–933.

[6] Z. Heng, P. Chen, Z. Jin, and Z. Chu, "Event-triggered control in networked control systems: A survey," in *The 27th Chinese Control and Decision Conference (2015 CCDC)*. IEEE, 2015, pp. 3092–3097.

[7] M. Mazo, A. Anta, and P. Tabuada, "On self-triggered control for linear systems: Guarantees and complexity," in *Control Conference (ECC), 2009 European*. IEEE, 2009, pp. 3767–3772.

[8] X. Wang and M. D. Lemmon, "Self-triggered feedback control systems with finite-gain stability," *IEEE transactions on automatic control*, vol. 54, no. 3, pp. 452–467, 2009.

[9] P. Ogren, E. Fiorelli, and N. E. Leonard, "Cooperative control of mobile sensor networks: Adaptive gradient climbing in a distributed environment," *IEEE Transactions on Automatic control*, vol. 49, no. 8, pp. 1292–1302, 2004.

[10] K.-E. Årzén, "A simple event-based pid controller," in *14th IFAC world congress*, 1999.

[11] M. Mazo and P. Tabuada, "On event-triggered and self-triggered control over sensor/actuator networks," in *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*. IEEE, 2008, pp. 435–440.

[12] L. Marín, M. Vallés, A. Soriano, A. Valera, and P. Albertos, "Event-based localization in ackermann steering limited resource mobile robots," *IEEE/ASME Transactions on Mechatronics*, vol. 19, no. 4, pp. 1171–1182, 2014.

[13] M. Martínez-Rey, E. Santiso, F. Espinosa, R. Nieto, and A. Gardel, "Smart laser scanner for event-based state estimation applied to indoor positioning," in *Factory Communication Systems (WFCS), 2014 10th IEEE Workshop on*. IEEE, 2014, pp. 1–10.

[14] D. Xu, L. Han, M. Tan, and Y. F. Li, "Ceiling-based visual positioning for an indoor mobile robot with monocular vision," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 5, pp. 1617–1628, 2009.

[15] S. Gupte, O. Masoud, R. F. Martin, and N. P. Papanikolopoulos, "Detection and classification of vehicles," *IEEE Transactions on intelligent transportation systems*, vol. 3, no. 1, pp. 37–47, 2002.

[16] M. Brezak, I. Petrović, and E. Ivanjko, "Robust and accurate global vision system for real time tracking of multiple mobile robots," *Robotics and Autonomous Systems*, vol. 56, no. 3, pp. 213–230, 2008.

[17] G. Klančar, M. Kristan, S. Kovačič, and O. Orqueda, "Robust and efficient vision system for group of cooperating mobile robots with application to soccer robots," *ISA transactions*, vol. 43, no. 3, pp. 329–342, 2004.

[18] T. Krajník, M. Nitsche, J. Faigl, T. Duckett, M. Mejail, and L. Přeučil, "External localization system for mobile robotics," in *Advanced Robotics (ICAR), 2013 16th International Conference on.* IEEE, 2013, pp. 1–6.

[19] M. Jung and W. Günthner, "An accurate and efficient camera-based indoor positioning approach for intralogistic environments," 2015.

[20] S. J. Julier and J. K. Uhlmann, "Unscented filtering and nonlinear estimation," *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401–422, 2004.

[21] M. Martínez-Rey, F. Espinosa, A. Gardel, C. Santos, and Á. Salcedo, "Computation reduction of smart sensors for ebse using mahalanobis sampling," in *Event-based Control, Communication, and Signal Processing (EBCCSP), 2015 International Conference on.* IEEE, 2015, pp. 1–8.

[22] M. Martínez-Rey, F. Espinosa, A. Gardel, and C. Santos, "On-board event-based state estimation for trajectory approaching and tracking of a vehicle," *Sensors*, vol. 15, no. 6, pp. 14 569–14 590, 2015.

[23] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," *Computer networks*, vol. 52, no. 12, pp. 2292–2330, 2008.

[24] J. Torres-Solis, T. H. Falk, and T. Chau, *A review of indoor localization technologies: towards navigational assistance for topographical disorientation.* INTECH Open Access Publisher, 2010.

[25] R. Mautz, "Indoor positioning technologies," 2012.

[26] A. Beetz, "Separation of control quality and measurement accuracy for guiding control tasks of an indoor construction machine simulator," in *Indoor Positioning and Indoor Navigation (IPIN), 2012 International Conference on.* IEEE, 2012, pp. 1–11.

[27] H. Hashimoto, "Intelligent space: Interaction and intelligence," *Artificial Life and Robotics*, vol. 7, no. 3, pp. 79–85, 2003.

[28] H.-g. Kang, B.-k. Kim, and W. Lee, "Indoor localization of dron using vision based sensor fusion," in *Control, Automation and Systems (ICCAS), 2015 15th International Conference on.* IEEE, 2015, pp. 932–934.

[29] Z. Jun and L. Guo-ping, "A novel localization method for indoor mobile robot based on odometry and ceiling visual features," in *Control Conference (CCC), 2015 34th Chinese.* IEEE, 2015, pp. 5941–5947.

[30] V. Malyavej and P. Udomthanatheera, "Rssi/imu sensor fusion-based localization using unscented kalman filter," in *The 20th Asia-Pacific Conference on Communication (APCC2014)*. IEEE, 2014, pp. 227–232.

[31] 2016. [Online]. Available: http://www.buffalotech.com/products/wireless/wireless-g-mimo-performance/wireless-g-mimo-performance-ethernet-converter

[32] 2016. [Online]. Available: http://www.viatech.com/en/support/eol/epia-en-eol/

[33] J. P. Hespanha, P. Naghshtabrizi, and Y. Xu, "A survey of recent results in networked control systems," *PROCEEDINGS-IEEE*, vol. 95, no. 1, p. 138, 2007.

[34] C. Santos Pérez, "Propuesta de control descentralizado con solapamiento para guiado en convoy de unidades p3-dx," Master's thesis, Universidad de Alcalá, 2010.

[35] A. Domingo Ajenjo, J. M. Arco Rodríguez, and B. Alarcos Alcázar, *Programación de aplicaciones en redes de comunicaciones bajo entorno UNIX*. Servicio de Publicaciones de la UAH, 1997.

[36] M. Martínez-Rey, F. Espinosa, A. Gardel, C. Santos, and E. Santiso, "Mobile robot guidance using adaptive event-based pose estimation and camera sensor," in *Event-based Control, Communication, and Signal Processing (EBCCSP), 2016 International Conference on*. IEEE, 2016.

[37] M. R. Salazar Arcucci, "Integración de matlab/simulink/rtw y linux/rtai para aplicaciones de control y comunicaciones en robótica," Master's thesis, Universidad de Alcalá, 2011.

Parte II

Apéndices

Apéndice A

Revisión de control de robot P3-DX

A.1 Fundamentos de control

La unidades móviles utilizadas en este trabajo son del fabricante MobileRobots, concretamente el modelo Pioneer P3-DX. Son robots de tracción diferencial, aplicando por tanto consignas de control a cada una de las ruedas activas de la unidad móvil. El control del robot incluye varios niveles [37]. El bajo nivel es el relativo al control PID independiente de cada rueda. El medio nivel es el de seguimiento de velocidades lineal y angular. El alto nivel es el correspondiente al seguimiento de trayectorias con realimentación de la pose del robot (posición y orientación).

El realizar el control remoto del robot, teniendo en cuenta el enlace inalámbrico, resulta un conjunto de estados que hace al sistema complejo. Para abordar esta complejidad, el enfoque de control planteado se basa en un modelo de variables de estados (VVEE).

Este modelo de VVEE se basa en la descripción de la ecuaciones de un sistema en términos de en ecuaciones diferenciales de primer orden, combinadas en una ecuación diferencial vectorial de primer orden.

El objetivo de esta sección es presentar los principios de control necesarios que se han aplicado en los siguientes capítulos.

A.1.1 Servosistema con realimentación de estados

La aplicación de un servosistema permite, gracias al integrador que incluye, que para un valor de error nulo, se obtiene una salida de control no nula, aunque se encuentre en presencia de perturbaciones que no varían o lo hacen lentamante.

En la figura A.1 se muestra el esquema general de un servosistema, donde u_{s_k} el vector de consignas de velocidad lineal y angular para el servosistema y u_{r_k} representa el vector de control del robot. X_k representa el vector de estados del modelo del robot, η_k el vector de los nuevos estados introducidos por los integradores e y_k es el vector de las medidas. K_R es la matriz de ganancia de realimentación del estado mientras que la matriz K_i es la matriz con las constantes de ganancia integral.

$$\begin{bmatrix} X_{k+1} \\ \eta_{k+1} \end{bmatrix} = \underbrace{\begin{bmatrix} G & 0 \\ -CG & I \end{bmatrix}}_{G_S} \begin{bmatrix} X_k \\ \eta_k \end{bmatrix} - \underbrace{\begin{bmatrix} H \\ -CH \end{bmatrix}}_{H_S} \underbrace{\underbrace{\begin{bmatrix} K_R & -K_I \end{bmatrix}}_{K_S} \begin{bmatrix} X_k \\ \eta_k \end{bmatrix}}_{u_{r_k}} + \begin{bmatrix} 0 & I \end{bmatrix} u_{s_{k+1}} \tag{A.1}$$

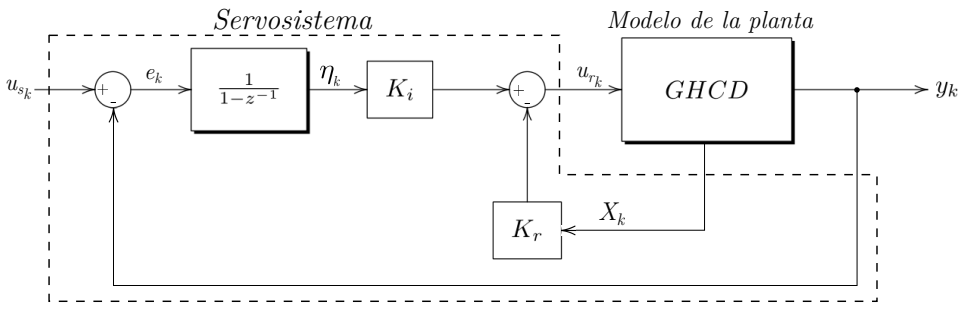

Figura A.1: Diagrama de bloques del servosistema con realimentación de estados.

$$y_k = \underbrace{\begin{bmatrix} G & 0 \end{bmatrix}}_{C_S} \begin{bmatrix} X_k \\ \eta_k \end{bmatrix} \tag{A.2}$$

Las ecuaciones que modelan el servosistema son las mostradas en A.1 y A.2. Se puede comprobar que la planta (robot más canal inalámbrico) es controlable, obteniendo un rango de la matriz de controlabilidad mostrada en A.3 y debe ser igual al número total de estados que en el caso de estudio es cuatro. Estos cuatro estado corresponden a la velocidad lineal y angular más el retardo de cada entrada (dos) debido al canal inalámbrico.

$$C_o = \begin{bmatrix} H & GH & G^2H & \dots & G^{n-1}H \end{bmatrix} \tag{A.3}$$

El servosistema, cuenta con la matriz K_S que está formada por la ganancia de realimentación de estados K_r y por la ganancia del integrador K_i. Para obtener dicha matriz existen múltiples técnicas, como la ubicación de autovalores o la técnica Linear Quadratic Regulator (LQR). En este caso se optó por la segunda. Por último, para garanticar la estabilidad del servosistema diseñado, el módulo de los autovalores asociados al servosistema debe ser inferior a la unidad.

A.1.2 Servosistema con estimador

Como se ha introducido al comienzo de esta sección, para realizar una prueba en las unidades P3-DX es necesario introducir un driver que permitirá a la unidad móvil realizar el movimiento de acuerdo a las consignas de velocidad lineal y angular, pero la salida de ese driver obtenemos únicamente el valor leído por los encoders, que son velocidad lineal y angular de la planta del sistema, que son dos de los cuatro estados que componen el modelo de la planta presentado en el subsección anterior. Por tanto es necesario solucionar el problema de la realimentación de estados para llevar a cabo los ensayos en el robot, ya que no todas las variables de estado que describen el comportamiento del robot son mediblesmedibles, y las que lo son incorporan ruido de sensado. Para solucionar este problema, es necesario incluir un estimador de estados, que tendrá como objetivo proporcionar las variables intermedias a partir de las consignas de entrada y de la respuesta de la salida del robot. Para poder incorporar un estimador a nuestro modelo, es necesario comprobar que cumple con las condiciones de observabilidad. Se puede comprobar que el sistema es observable, obteniendo el rango de la matriz de observabilidad como se muestra en A.4, el rango de dicha matriz debe ser equivalente al número de estados del modelo, que en este caso es cuatro. A través de $MATLAB^{®}$ usando la función *obsv* e introduciendo las matrices G y C del modelo de variables de estado de la unidad móvil obtenemos el rango de la matriz de observabilidad.

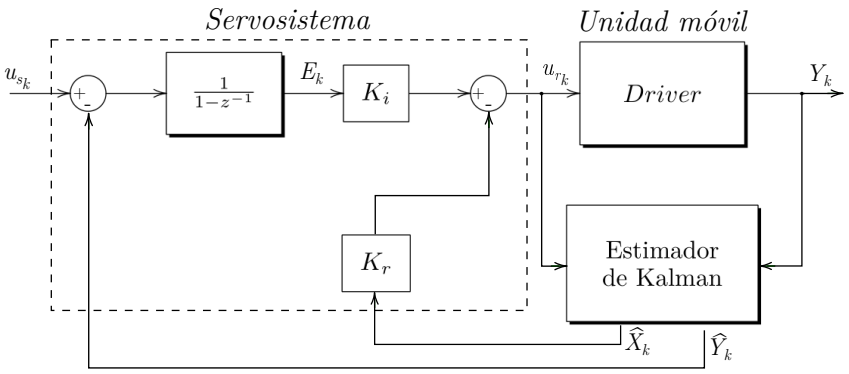

Figura A.2: Diagrama de bloques del servosistema con estimador de estados.

$$O_b = \begin{bmatrix} C \\ CG \\ CG^2 \\ \dots \\ CG^{n-1} \end{bmatrix} \tag{A.4}$$

Comprobada la observabilidad del sistema, el siguiente paso es diseñar el estimador de estados. En la figura A.2, se muestra un esquema del servosistema junto con el estimador.

El estimador seleccionado es un estimador de Kalman estacionario, que establece un valor fijo para la matriz de ganancias de Kalman, evitando el tener que recalcular para cada periodo el valor de la matriz de ganancias de Kalman. Este estimador además, se caracteriza por ser de orden total, ya que se estiman todos los estados, y ser también de tipo actualizado, por lo que tiene una fase de predicción y otra de corrección que se rigen por las ecuaciones A.5 y A.6.

$$\tilde{X}_k = G\hat{X}_{k-1} + HU_{k-1} \tag{A.5}$$

$$\hat{X}_k = \tilde{X}_k + K_e(Y_k - C\tilde{X}_k) \tag{A.6}$$

La matriz de ganancias K_e determinará el comportamiento del estimador. Esta matriz es la encargada de ponderar el peso que se da al modelo frente a la medida en cada instante y por ende la velocidad con la que los estados estimados convergen con las reales. El criterio de decisión de esta matriz de ganancias se basa en las ecuaciones de estado y medidas mostradas en A.7 y A.8.

$$X_{k+1} = GX_k + HU_k + W_K \tag{A.7}$$

$$Y_k = CX_k + DU_k + V_k \tag{A.8}$$

Donde W_K representa el ruido del modelo el cual tiene una matriz de covarianza Q_{est} (A.9) y V_k representa el ruido de medida con una matriz de covarianza R_{est} (A.10), siendo la varianza cruzada entre ambas variables nula. Paraobtener una estimación óptima con el filtro de Kalman ambas variables de ruido deben ser modeladas con una distribución gaussiana de media cero.

Para modelar el ruido de medida V_k basta con tomar medidas de los encoders ante una entrada constante, y obtener du varianza y desviación típica del ruido de medida. Mientras que para modelar el ruido del modelo W_k no es tan sencillo y por tanto este valor se va a modelar de forma heurística. Cabe destacar que lo importante para el cálculo de la matriz de ganancias del estimador de Kalman es

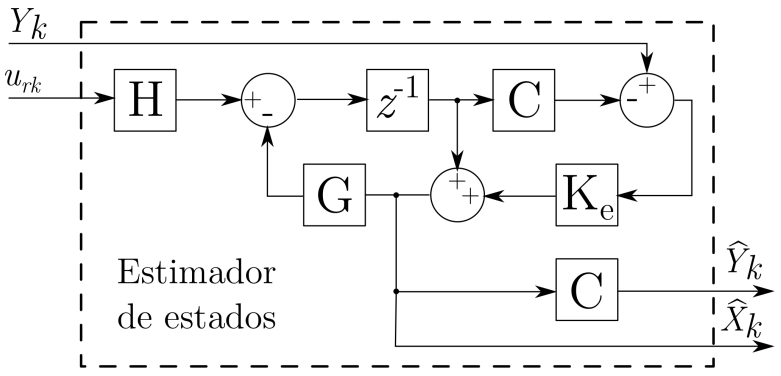

Figura A.3: Diagrama de bloques del estimador de Kalman de estados.

la relación entre ambas varianzas.

$$Q_{est} = \mathbf{I} \cdot 10^{-8} \tag{A.9}$$

$$R_{est} = \mathbf{I} \cdot 10^{8} \tag{A.10}$$

Una vez modelados la varianza del ruido de medida y la del ruido de estado ya se está en disposición de calcular la matriz K_e, que ser hará con la ayuda de la función de MATLAB *dlqe*. Los resultados de esta matriz de muestran en la ecuación A.11.

$$K_e = \begin{bmatrix} 0{,}01 & 0 \\ 0 & 0{,}01 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{A.11}$$

Por último, habría que comprobar que el estimador es estable, para ello hay que fijarse en la posición de los autovalores del observador, para ello dichos autovalores deben situarse en el semiplano izquierdo, es decir deben ser negativos para garantizar la estabilidad del estimador. En A.12 se muestran los autovalores obtenidos, como se pude comprobar al ser todos negativos se garantiza que están en el semiplano izquierdo y por tanto se garantiza así la estabilidad del estimador.

$$\text{Autovalores del estimador} = \begin{bmatrix} 0{,}661612 + 0{,}218808i \\ 0{,}661612 - 0{,}218808i \\ 0{,}651434 + 0{,}221452i \\ 0{,}651434 - 0{,}221452i \end{bmatrix} \tag{A.12}$$

En la figura A.3 se muestra el esquema del estimador diseñado basado en el Filtro de Kalman.

A.1.3 Lyapunov Based Controller

Como se ha visto hasta ahora, el controlrealiza un seguimiento de velocidad lineal y velocidad angular, pero esto hace que no se pueda realimentar el lazo de control con la posición relativa del robot, ya que el paso de velocidad lineal y angular a posición relativa (x, y, θ) no se trata de una transformación lineal ya que el modelo cinemático del robot se rige por la ecuación A.13, la cual incluye los operadores no lineales senos y cosenos.

$$\begin{cases} \dot{x} = \upsilon \ \cos(\theta) \\ \dot{y} = \upsilon \ \sin(\theta) \\ \dot{\theta} = \omega \end{cases} \tag{A.13}$$

Para hacer posible la realimentación del control con la posición relativa del robot, es necesario un modelo cinemático que se ha descrito en la ecuación A.13, pero además un controlador que permita a partir del error de posición obtener una consigna de velocidad lineal y angular que permita al servosistema con estimador presentado en la sección A.1.2, realizar las labores de control oportunas. El Lyapunov Based Controller es el controlador, que va a conseguir dicho objetivo.

A lo largo de esta sección se van a describir los aspectos generales, así como los aspectos relacionados con la estabilidad del Lyapunov Based Controller. También se abordará la forma de alcanzar un punto relativo a la posición inicial del robot, así como la forma de realizar el seguimietno de una trayectoria.

A.1.3.1 Aspectos generales y concepto de estabilidad

Aleksandr Mikhailovich Lyapunov nació en junio de 1857, fue un matemático ruso que falleció a los 61 años de edad, pero contribuyó de forma considerablemente en la creación de la teoría moderna de la estabilidad de sistemas dinámicos, gracias a su función de Lyapunov.

Estas funciones de Lyapunov que planteó, demuestran la estabilidad de un cierto punto fijo en un sistema dinámico o en las ecuaciones diferenciales autónomas. Todas aquellas funciones que puedan probar la estabilidad de un punto de equilibrio cualquiera se llama candidatas a funciones de Lyapunov.

No existe un método general para encontrar una función candidata que permita demostrar la estabilidad del sistema, pero en caso de no encontrar dicha función de Lyapunov no implica la inestabilidad del sistema.

Para un espacio de estados de un sistema no lineal que viene dado por A.14, donde f es una función no lineal y variante en el tiempo. x representa el vector de estados, u la ley de control y t el tiempo.

$$\dot{x}(t) = f[x(t), u(t), t] \quad | \quad x \in \mathbb{R}^n, \quad u \in \mathbb{R}^m, \quad t \in \mathbb{R} \tag{A.14}$$

Se va a definir el sistemas autónomo como un sistema invariante y no lineal como se muestra en A.15.

$$\dot{x} = f(x) \quad | \quad x \in \mathbb{R}^n \tag{A.15}$$

Se va a definir un estado, x_e como punto de equilibrio si $x(t) = x_e$ y permanece en dicho estado cuando $t \to \infty$.

Lyapunov se concentra en el análisis de la estabilidad de los sistemas, para ello se define como estabilidad al estado que parte del equilibrio ($x = 0$) de un sistema dinámico autónomo y no lineal, se dice que es estable si:

$$\forall R > 0, \quad \exists r > 0, \quad \{\|x(0)\| < r\} \implies \{\forall t \geq 0, \|x(t)\| < R\} \tag{A.16}$$

En la figura A.4 se muestra una representación gráfica de la definición anterior.

Se puede definir que un un punto de equilibrio es asintóticamente estable si este estable y además:

$$\exists r > 0, \quad \{\|x(0)\| < r\} \implies \{\lim_{t \to \infty} \|x(t)\| = 0\} \tag{A.17}$$

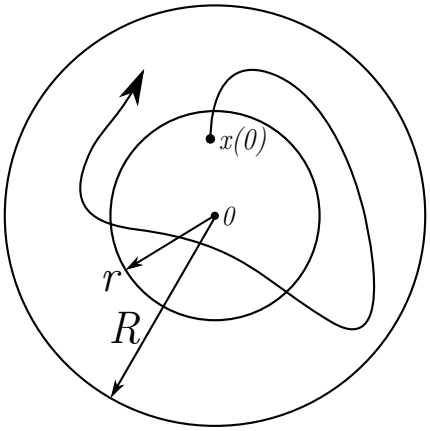

Figura A.4: Descripción gráfica de la estabilidad.

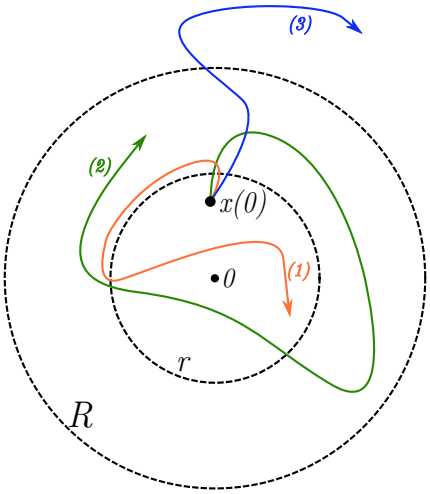

Figura A.5: Descripción gráfica de las definiciones de estabilidad.
(1) punto de equilibrio asintóticamente estable, (2) punto de equilibrio matginalmente estable, (3) punto de equilibrio inestable

Cuando el punto de equilibrio es estable pero no es asintóticamente estable se le llama marginalmente estable. En la figura A.5 se muestra de forma gráfica estas definiciones. La curva 1 (en color naranja) se corresponde con la definición de un un punto de equilibrio asintóticamente estable, la curva 2 (en color verde) es lo que se ha definido como un punto de equilibrio marginalmente estable y por último la curva 3 (en color azul) muestra un punto de equilibrio inestable.

Se denomina que el punto de equilibrio es globalmente asintótico estable si la estabilidad asintótica es válida para cualquier estado inicial. Para sistemas LTI, independientemente de cómo se defina la estabilidad del punto de equilibrio (marginalmente estable o inestable), la estabilidad es siempre global. Los sistemas no lineales por el contrario tiene puntos de equilibrio localmente estables.

A.1.3.2 Métodos de análisis de la estabilidad

En el apartado anterior se han definido la estabilidad según la teoría de Lyapunov, pero hasta ahora no se ha hablado de cómo realizar un análisis de la estabilidad de los sistemas. Para ello, Lyapunov tiene dos métodos, lo que llamaremos método indirecto y directo. Abordaremos en primer lugar el método indirecto y continuaremos con el otro de los métodos.

El método indirecto, se basa en las propiedades de estabilidad de un estado de un sistema no lineal

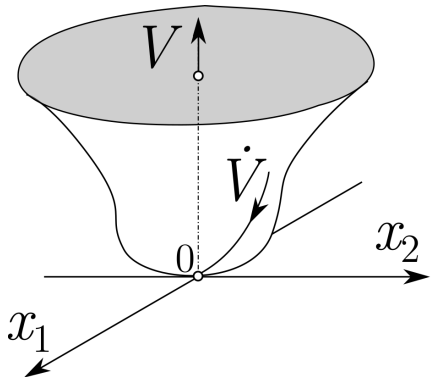

Figura A.6: Descripción gráfica del Teorema de estabilidad global.

en función de su linealización sobre un punto de equilibrio dado. Considerando un sistema dinámico no lineal $\dot{x} = f(x)$, asumiendo que $f(x)$ es continuamente diferenciable. Se procede a realizar la linealización de la función:

$$\dot{x} = \underbrace{\left(\frac{\partial f(x)}{\partial x}\right)_{x=0}}_{A} x + \underbrace{\cancelto{0}{f_{h.o.t.}(x)}}_{higher-order\ terms} \cong Ax \tag{A.18}$$

De la misma forma si se define $\dot{x} = f(x, u)$:

$$\dot{x} = \underbrace{\left(\frac{\partial f(x)}{\partial x}\right)_{\substack{x=0\\u=0}}}_{A} x + \underbrace{\left(\frac{\partial f(x)}{\partial u}\right)_{\substack{x=0\\u=0}}}_{B} u + \underbrace{\cancelto{0}{f_{h.o.t.}(x,u)}}_{higher-order\ terms} \cong Ax + Bu \tag{A.19}$$

El segundo de los métodos, el método directo. Se basa en la generalización de los conceptos de energía. Se va a definir entonces una función de energía $V(x)$, y la variación de energía de la función anterior como $\dot{V}(x)$.

La función V, se define positiva si $V(0) = 0 \ \wedge \ \forall \ x \neq 0 \ , \ V(x) > 0$. De la misma forma, V se define negativa si $V(0) = 0 \ \wedge \ \forall \ x \neq 0 \ , \ V(x) < 0$.

Si la función $V(x)$ es definida positiva, la cual tiene derivadas parciales continuas semidefinidas negativas ($\dot{V}(x) \leq 0$), entonces a la función $V(x)$ se puede llamar función de Lyapunov.

El teorema de la estabilidad global dice que si en una región B_R existe una función $V(x)$ con derivadas parciales continuas tal que $\forall x \in B_R : \ V(x) > 0 \ \wedge \ \dot{V}(x) \leq 0$, entonces el punto de equilibrio es estable.

En la figura A.6 se muestra una representación gráfica de este teorema. Lo que viene a decir el teorema es que si la energía se disipa continuamente, el sistema va a tender a estabilizarse en un punto de equilibrio.

Por tanto, se puede decir que energía nula se corresponde a un punto de equilibrio ($x = 0$). Además, si se tiene una estabilidad asintótica implica que la función energía convergerá a cero. Por último, es trivial que la inestabilidad implica un incremento de energía.

A.1.3.3 LBC de sistema no lineal

En el punto anterior, se ha definido la función de Lyapunov que garantiza la estabilidad de un sistema no lineal. En este apartado se va a particularizar para el sistema de control presentado en la sección A.1.2.

Si se recuerda la cinemática que rige los movimientos del robot, mostrada en A.20.

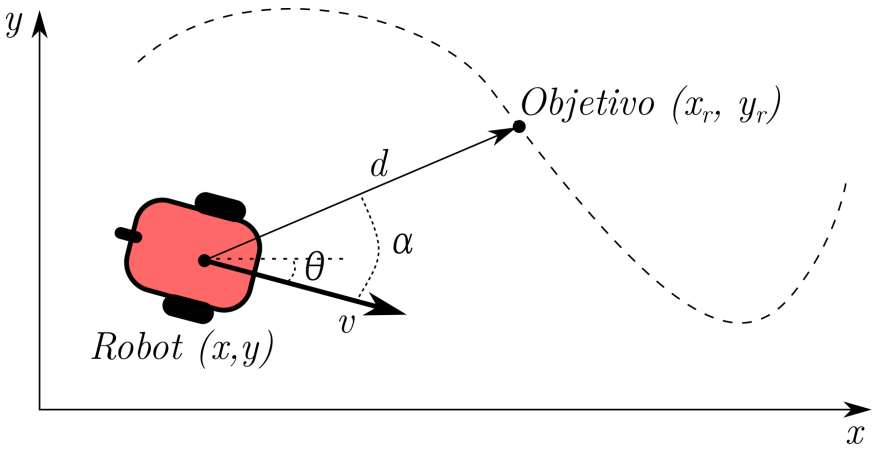

Figura A.8: Descripción gráfica aproximación a un punto.

$$\begin{cases} \dot{x} = v(t)\ \cos(\theta(t)) \\ \dot{y} = v(t)\ \sin(\theta(t)) \\ \dot{\theta} = \omega(t) \end{cases} \tag{A.20}$$

En base a la cinemática descrita, se va a tratar que el robot realice una trayectoria a partir de sus coordenadas relativas (x, y, θ). En los siguientes apartados se describen de forma matemática cómo se alcanza un punto de referencia y realizar el seguimiento de una trayectoria.

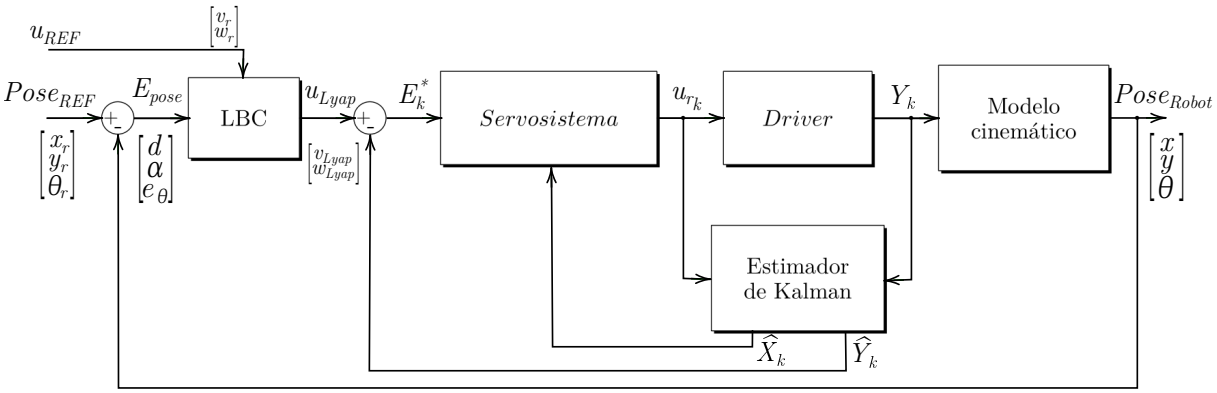

Figura A.7: Diagrama de bloques del LBC junto con el servosistema y estimador de estados.

A.1.3.3.1 Objetivo de aproximación a un punto

El objetivo en este apartado es diseñar la ley de control $u = [v\ \omega]^T$ que permita alcanzar el punto de referencia $q_r = [x_r\ y_r]^T$ a partir de una pose inicial $q_0 = [x\ y\ \theta]^T$.

La situación de partida es la mostrada en la figura A.8, dónde el robot situado en una posición inicial, tiene que realizar una trayectoria que le permita alcanzar el objetivo, que es un punto de referencia dado. En la figura, se define d como la distancia que existe entre el robot y el objetivo dado.

Para simplificar la representación se va a tomar la figura A.9 para describir las variables de la función de Lyapunov para establecer la ley de control que se tiene como objetivo.

En la figura A.9 se define e_x como el la diferencia en la componente x entre el objetivo y la posición del robot. Del mismo modo e_y para la componente y. Se puede definir el error de seguimiento, en coordenadas

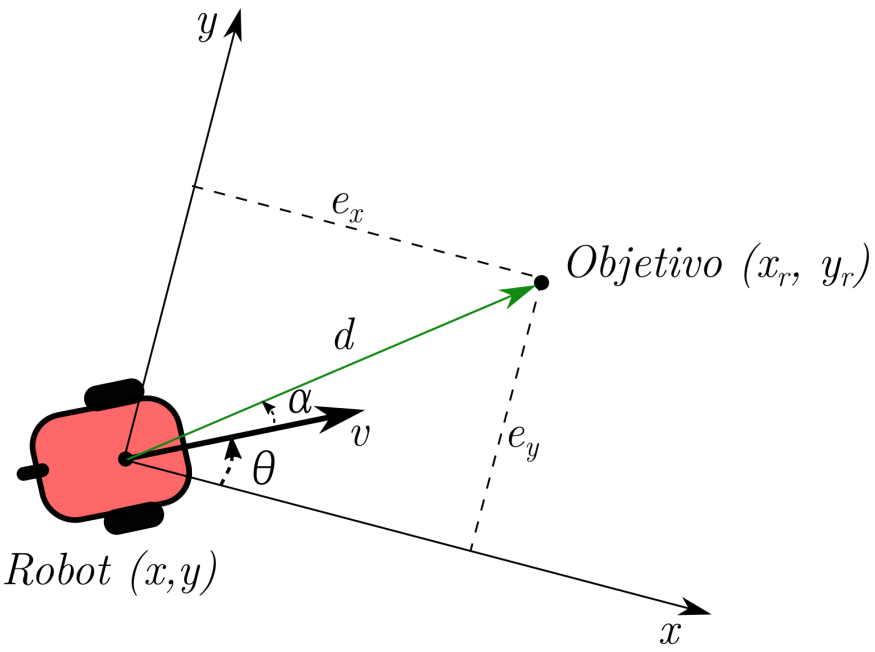

Figura A.9: Descripción gráfica aproximación a un punto con detalle de las variables.

polares, al punto $q_e = [d\ \alpha]^T$, donde d y α se definen:

$$\begin{aligned} d &= \sqrt{e_x^2 + e_y^2} \\ \alpha &= \text{atan2}(e_y, e_x) - \theta \end{aligned} \tag{A.21}$$

Se define la derivada del error de seguimiento como $\dot{q_e} = [\dot{d}\ \dot{\alpha}]^T$, donde $\dot{d}$ y $\dot{\alpha}$ resultan:

$$\dot{d} = -v\ \cos\alpha \tag{A.22}$$

$$d\dot{\alpha} = -\omega d + v\ \sin\alpha \tag{A.23}$$

El objetivo es encontrar un función de Lyapunov que se defina positiva y cuya derivada esté definida negativa, para así conseguir la condición de estabilidad expuesta anteriormente. Esa función V, está definida en la ecuación A.24, dónde se ve claramente que se trata de una función definida positiva.

$$V = \frac{1}{2}d^2 + \frac{1}{2}\alpha^2 \tag{A.24}$$

$$\dot{V} = d\dot{d} + \alpha\dot{\alpha};\ \dot{V} = f(v, \omega) \tag{A.25}$$

De la ecuación A.25 se puede deducir gracias a las expresiones A.22 y A.23 es definida negativa, ya que d y α son negativas, por tanto su derivada también lo será. La ecuación A.26 comprueba esta para valores positivos de las ganancias K_v y K_ω. Por tanto, se puede decir que existe un punto de estabilidad definido por la función V.

$$\dot{V} = -K_{v1}d^2\cos^2(\alpha) - K_{\omega 1}\alpha^2 \leq 0 \tag{A.26}$$

Con todo esto, la ley de control que va a permitir alcanzar un punto de referencia a partir de una posición de inicio del robot se muestra en A.27

$$\begin{cases} v = K_{v1} d \cos(\alpha) \\ \omega = K_{\omega 1}\alpha + K_{v1}\cos(\alpha)\sin(\alpha) \end{cases} \tag{A.27}$$

A.1.3.3.2 Seguimiento de trayectoria

El punto de referencia definido en el apartado anterior, es un punto estático, es decir que no cambia. En este apartado, se va a diseñar una ley de control $u = [v\ \omega]^T$ que logre alcanzar un punto de referencia dinámico $q_r = [x_r\ y_r\ \theta_r]^T$, es decir que cambia, a partir de la velocidad lineal y angular de referencia $u_{ref} = [v_r\ \omega_r]^T$ y de la pose real del móvil $q = [x\ y\ \theta]^T$. Uno de los aspectos diferenciales respecto al punto anterior es que la orientación del móvil ahora importa, de modo que el ángulo de orientación que tenga el punto de referencia (θ_r) se verá implicado en las ecuaciones.

Al igual que antes, con el apoyo de la figura A.10, se va a definir en coordenadas polares el error de seguimiento como $q_e = [d\ \alpha\ e_\theta]^T$. Dónde d, α y e_θ se definen como:

$$\begin{gathered} d = \sqrt{e_x^2 + e_y^2} \\ \alpha = \text{atan2}(e_y, e_x) - \theta \\ e_\theta = \theta_r - \theta \end{gathered} \tag{A.28}$$

Al igual que antes, la derivada del error de seguimiento se define como $\dot{q}_e = [\dot{d}\ \dot{\alpha}\ \dot{e}_\theta]^T$, donde $\dot{d}$,$\dot{\alpha}$ y$\dot{e}_\theta$, con el apoyo de la figura A.11 se pueden definir las siguientes expresiones:

$$\dot{d} = V_r \cos(\alpha - e_\theta) - v\ \cos\alpha \tag{A.29}$$

$$d\dot{\alpha} = v\sin(\alpha) - \omega d + V_r\ \sin(\alpha - e_\theta) \tag{A.30}$$

$$\dot{e}_\theta = \omega_r - \omega \tag{A.31}$$

Otras expresiones que se pueden extraer a partir de la figura A.11, son las siguientes:

$$V_{ls} = k_{v2} d$$

$$V_{md} = \sqrt{V_r^2 + V_{ls}^2 + 2V_r V_{ls}\cos(\alpha - e_\theta)} \tag{A.32}$$

$$\theta_{md} = \text{atan2}\left(\frac{V_{ls}\sin(\alpha - e_\theta)}{V_r + V_{ls}\cos(\alpha - e_\theta)}\right) + \theta_r$$

El siguiente paso a realizar es obtener la función de Lyapunov definida positiva y cuya derivada esté definida negativa, para garantizar la estabilidad. La función de Lyapunov que cumple dichos criterios es la mostrada en A.33 y se puede comprobar que cumple la condición de estabilidad tal y como se muestra en A.35.

$$V = \frac{1}{2}d^2 + 1 - \cos(\theta_{md} - \theta) \tag{A.33}$$

$$\dot{V} = d\dot{d} + (\dot{\theta}_{md} - \omega)\sin(\theta_{md} - \theta);\ \dot{V} = f(v, \omega) \tag{A.34}$$

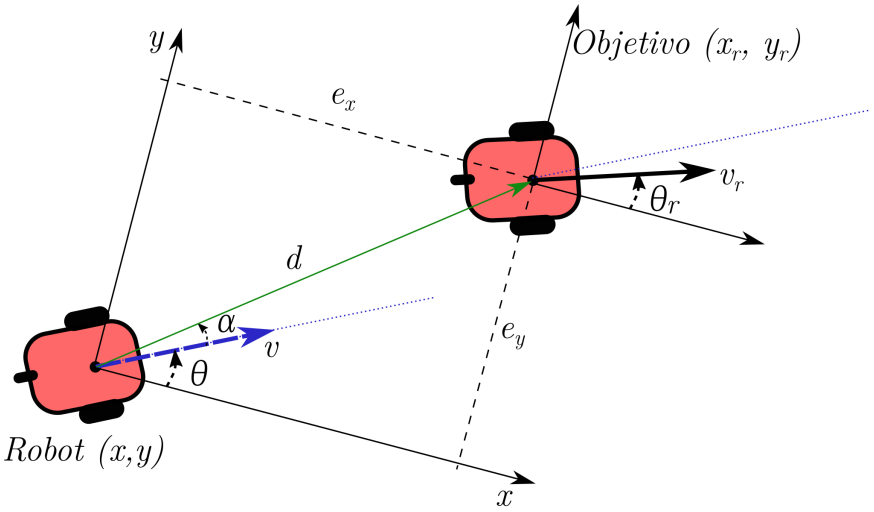

Figura A.10: Descripción gráfica aproximación a un punto relacionando las variables con el objetivo.

$$\dot{V} = -K_{v2}d^2 - K_{\omega 2}(K_{v2}d\sin\alpha + v_r\sin e_\theta)^2 \leq 0 \tag{A.35}$$

Por tanto, la ley de control que va a permitir alcanzar el objetivo de este apartado, que es el realizar el seguimiento de la trayectoria a partir de la información que proporciona una cámara, se define en las ecuaciones A.36.

$$\begin{cases} v = V_{ls}\cos\alpha + V_r\cos e_\theta \\ \omega = \dot{\theta}_{md} + V_{md}\Big(K_{\omega 2}\left(V_{ls}\sin\alpha + V_r\sin e_\theta\right) + d\sin\alpha\Big) \end{cases} \tag{A.36}$$

Donde

$$\dot{\theta}_{md} = \frac{(V_r\dot{V}_{ls} - \dot{V}_rV_{ls})\sin(\alpha - e_\theta) + V_{ls}^2(\dot{\alpha} - \dot{e}_\theta) + V_rV_{ls}(\dot{\alpha} - \dot{e}_\theta)\cos(\alpha - e_\theta)}{V_{md}^2} + \omega_r \tag{A.37}$$

Una vez que se ha definido el controlador basado en Lyapunov, se podrá realizar el control de la unidad móvil P3-DX, realimentando al controlador de Lyapunov con la pose del robot que proporciona una cámara.

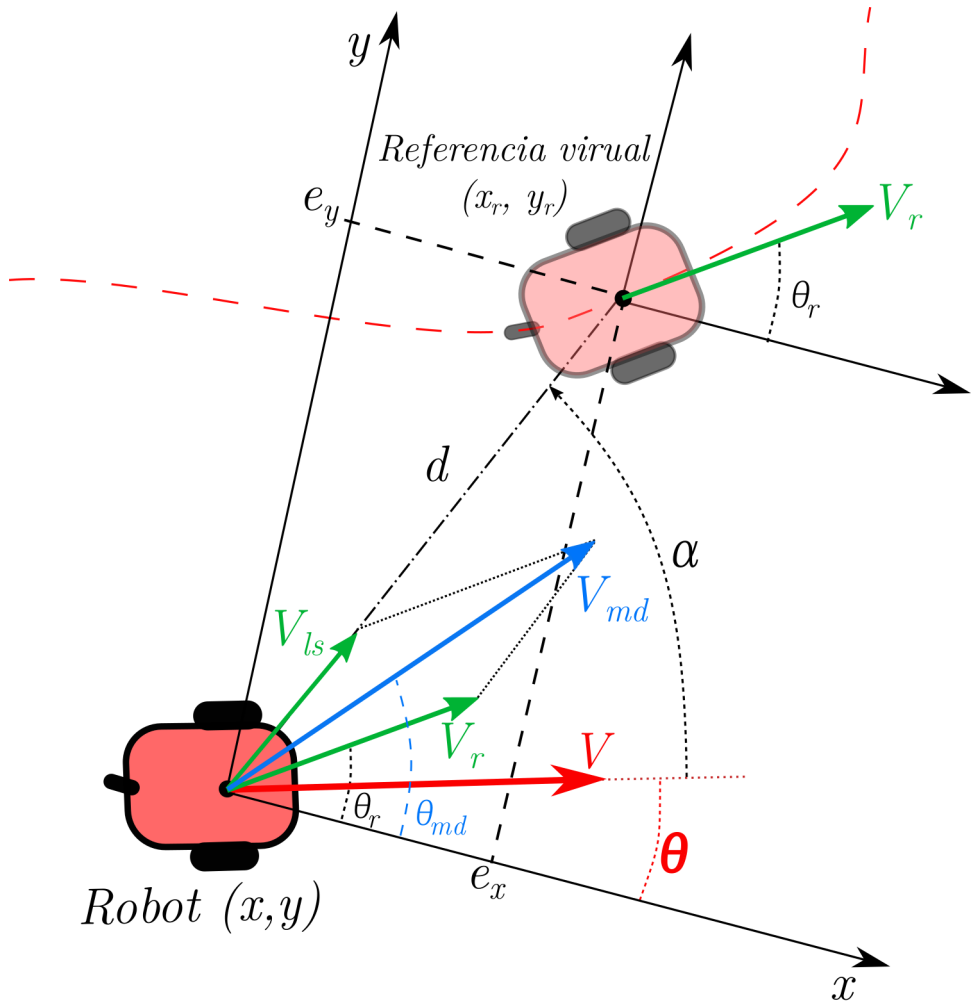

Figura A.11: Descripción gráfica de seguimiento de trayectoria.

Apéndice B

Instalación Ubuntu 12.04 LTS - RTAI

En este apéndice se va a describir la instalación realizada de Ubuntu 12.04 LTS - RTAI junto con la instalación de MATLAB r2012a. A continuación se describen los pasos a realizar.

Paso 1 Instalar Ubuntu 12.04 x86 (32 bits). En la documentación adjunta se encuentra una copia de la versión disponible en la web de Ubuntu.

Para el caso de los mini PCs Intel NUC se ha de realizar con la versión 12.04.1.

Para realizar la instalación, se puede hacer uso de la herramienta UNetbootbin para crear un disco USB booteable y realizar la instalación normalmente.

Durante la instalación se recomienda realizar las siguientes particiones al disco:

1. Primaria de 1024 MB en formato *ext*4 y `/boot`.
2. Lógica del tamaño de dos veces el tamaño de RAM utilizada (2x4GB), es decir 8192MB en formato de área de intercambio.
3. Lógica en formato *ext*4 y `/` que se usará para la instalación del Sistema Operativo (SO). Aproximadamente se le puede dar un tamaño de 50GB.
4. Lógica en formato *ext*4 y en `/home` que será utilizada para los "Datos de usuario".

Una vez completada la instalación continuamos con el siguiente paso.

Paso 2 Instalación del kernel RTAI.

1. Actualizar Ubuntu con los últimos paquetes de Ubuntu Precise:
   ```
   sudo apt-get update
   sudo apt-get dist-upgrade
   ```
2. Añadir la clave para comprobación de autenticación de paquetes:
   ```
   sudo apt-key adv -keyserver hkp://keys.gnupg.net -recv-key
   3cb9fd148f374fef
   ```
3. Añadir la nueva fuente apt:
   ```
   sudo add-apt-repository "deb http://linuxcnc.org/ precise base
   2.7-rtai"
   ```

4. Actualizar la lista de paquetes:
   `sudo apt-get update`
5. Instalar el kernel RTAI y módulos necesarios:
   `sudo apt-get install linux-image-3.4-9-rati-686-pae`
   `rtai-modules-3.4-9-rtai-686-pae`
6. Terminar de instalar el kernel con los headers:
   `sudo apt-get install linux-headers-3.4-9-rtai-686-pae`
7. Reiniciar el ordenado y automáticamente arrancará con el kernel RTAI. Para comprobar que es así, una vez reiniciado, abrir un terminal y ejecutar:
   `uname -r`
   Deberemos ver la siguiente respuesta: `3.4-9-rtai-686-pae`

Paso 3 Instalación de tarjeta de red y tarjeta WiFi para los mini PC's Intel NUC. Al instalar el kernel RTAI, el sistema no es capaz de reconocer ninguna tarjeta de red. Por tanto, es necesario instalar los drivers de forma manual.

1. Instalación de la tarjeta de red Ethernet.
   (a) Descomprimir el driver para la tarjeta de red Ethernet:
   `tar xvf e1000e-3.3.4.tar.gz`
   (b) Abrir el directorio para realizar la instalación:
   `cd e1000e-3.3.4/src/`
   (c) Realizar la instalación:
   `sudo make install`
   (d) Reiniciar el ordenador.

2. Instalación de la tarjeta WiFi.
   (a) Instalar los paquetes necesarios:
   `sudo apt-get install linux-headers-generic build-essential`
   (b) Descomprimir el driver de la tarjeta WiFi:
   `tar xvf backports-3.18.1-1.tar.xz`
   (c) Abrir el directorio:
   `cd backports-3.18.1-1`
   (d) Comprobar los drivers:
   `sudo make defconfig-iwlwifi`
   (e) Preparar la instalación:
   `sudo make`
   (f) Realizar la instalación:
   `sudo make install`
   (g) Reiniciar el ordenador.

Paso 4 Una vez comprobada la correcta instalación de las tarjetas de red, es posible comenzar con la instalación de MATLAB y los paquetes necesarios.

1. Es necesario instalar una versión previa de compilador de C/C++ para garantizar el correcto funcionamiento de los ficheros del sistema Ubuntu RTAI previo.

```
sudo apt-get install gcc-4.4
sudo apt-get install g++-4.4
```

2. Es necesario enlazar los nuevos compiladores ya que si comprobamos la versión de compilador que está utilizando el sistema (`gcc -v`) indicará que se está usando una versión más actual.
   ```
   sudo rm /usr/bin/gcc
   sudo rm /usr/bin/g++
   ```
3. creamos los enlaces simbólicos:
   ```
   sudo ln -s /usr/bin/gcc-4.4 /usr/bin/gcc
   sudo ln -s /usr/bin/g++-4.4 /usr/bin/g++
   ```
4. Comprobar la versión del compilador es la deseada (`gcc -v`)

Paso 5 Instalación de MATLAB r2012a

1. Crear un enlace simbólico a `/usr/realtime`
   ```
   ln -s /usr/realtime-3.4-9-rtai-686-pae /usr/realtime
   ```
2. Añadir las librerías compartidas:
   ```
   sudo gedit /etc/ld.so.conf
   ```
3. Agregar al final del documento la ruta de las librerías dinámicas de RTAI:
   ```
   /usr/realtime/lib
   ```
4. Actualizar la ruta del sistema operativo.
   ```
   sudo ldconfig
   ```
5. Crear la carpeta de instalación de MATLAB.
   ```
   sudo mkdir /opt/matlab
   ```
6. Descomprimir MATLAB y acceder a la carpeta donde se ha descomprimido.
   ```
   cd "MATLAB folder"
   sudo chmod +x install
   sudo chmod +x /sys/java/jre/glnx686/jre/java/bin/java
   sudo ./install
   ```
7. Realizar la instalación con el interfaz, asegurandose que la dirección de instalación se realiza en /opt/matlab.

8. Una vez instalado, se deben dar permisos de lectura y escritura al usuario del sistema.
   ```
   chown -R alcor:alcor /opt/matlab
   ```
9. Crear un enlace simbólico de MATLAB para ejecutarlo a través de terminal.
   ```
   sudo ln -s /opt/matlab/bin/matlab /usr/local/bin/matlab
   ```

Paso 6 Instalación de 'target' RTAI para Simulink.

(a) Descarga la versión de RTAI 4.1.1 desde el repositorio de `rtai.org`. Descomprimir y ejecutar:
```
sudo mkdir /opt/matlab/rtw/c/rtai
sudo cp -R /"Directorio"/rtai-4.1.1/rtai-lab/matlab/*
/opt/matlab/rtw/c/rtai/
```

(b) Iniciar MATLAB y ejecutar el script: `/opt/matlab/rtw/c/rtai/setup.m`

Paso 7 Si al iniciar MATLAB desde el terminal aparece un mensaje de que no es posible encontrar la librería `libc.so.6` ejecutar:
`sudo ln -s /lib/i386-linux-gnu/libc.so.6 /lib/libc.so.6`

Paso 8 Si al generar los MEXglx, aparece algún mensaje de error:
`sudo apt-get install ia32-libs-multiarch`
y reiniciar.

Paso 9 Para migrar cualquier fichero de alguna versión previa de Ubuntu RTAI, es necesario cambiar algunos parámetros de los Makefiles:

1. Del Makefile utilizado para compilar los drivers del robot y generar los distintos `mexglx` sustituir:
   `MEX=mex` por `MEX=/opt/matlab/bin/mex`
   `CXX=gcc-4.1` por `CXX=gcc-4.4`
2. De fichero `rtai-custom-tmf` usado para generar código mediante simulink, sustituir donde aparezca `gcc-4.1` por `gcc-4.4` en todo el fichero.