UNIVERSIDAD DE ALCALÁ
Escuela Politécnica Superior

INGENIERÍA TÉCNICA DE TELECOMUNICACIÓN
ESPECIALIDAD EN SISTEMA ELECTRÓNICOS

Trabajo Fin de Carrera

“Control y Guiado de una Silla de Ruedas en Entornos Interiores”

Pedro del Moral Peña

2013

UNIVERSIDAD DE ALCALÁ

Escuela Politécnica Superior

INGENIERÍA TÉCNICA DE TELECOMUNICACIÓN
ESPECIALIDAD EN SISTEMA ELECTRÓNICOS

Trabajo Fin de Carrera

Control y Guiado de una Silla de Ruedas en Entornos Interiores

Autor: Pedro del Moral Peña

Directores: Marta Marrón Romera
Juan Carlos García García

TRIBUNAL:

Presidente: Ignacio Fernández Lorenzo

Vocal 1º: Ana Isabel de Andrés Rubio

Vocal 2º: Marta Marrón Romera

CALIFICACIÓN:……
FECHA:……

A mis padres.

Agradecimientos.

En primer lugar agradecer a mis tutores su ayuda y la confianza puesta en mí, y en especial a Marta por su infinita paciencia y por animarme a continuar con este trabajo en los momentos bajos.

A mis padres, a quienes les debo todo, por haberme dado la oportunidad de llegar a ser quien soy.

Y al resto de familia y por supuesto mis amigos, por haberse interesado por mí, por los ánimos y por todos los buenos momentos que pasamos juntos.

RESUMEN

El presente trabajo detalla el desarrollo de un sistema capaz de controlar una silla de ruedas y guiarla a través de un edificio. Partiendo de una aplicación generadora de rutas ya diseñada anteriormente, donde se indican un origen y un destino dentro de un edificio, se crean las trayectorias necesarias para recorrer la ruta obtenida y se aplican las técnicas de control necesarias para que estas sean seguidas hasta alcanzar el destino, a partir del modelo cinemático de la silla identificado en una tarea previa. Para ello se emplean técnicas de deadreckoning basadas en odometría, y métodos de navegación clásica.

Palabras clave:

- Identificación de modelos.
- Deadreckoning.
- Odometría.
- Control de posición.
- Seguimiento de trayectorias.

ABSTRACT

This project details the development of a system capable of controlling a wheelchair and guides it through a building. Starting from an application generator of routes previously designed, where an origin and a destination within the building are indicated, necessary paths are created to roam the routes obtained and the necessary control techniques are applied so that these are followed until the destination is reached parting from the kinematic model of the wheelchair identified in a previous task. This is done using deadreckoning techniques based on odometry and classical navigation methods.

Keywords:

- Model identification.
- Deadreckoning.
- Odometry.
- Position control.
- Route tracking.

Índice.

Índice De Figuras.

Tema I:

Tema II:

Tema V:

Índice De Tablas.

I. RESUMEN EXTENDIDO

El proyecto del cual trata este documento tiene como objetivo el manejo autónomo de una silla de ruedas a través de un edificio. Está enfocado a ayudar a aquellas personas incapaces de desplazarse por si solas y además no puedan manejar de forma adecuada una silla de ruedas.

No solo se trata de encontrar una posible solución a los problemas de maniobrabilidad de la silla, sino que además, gracias a los trabajos realizados con anterioridad, también se facilita la circulación de los usuarios por las zonas más apropiadas aunque éstas sean desconocidas para ellos. Todo ello permite que el usuario final tan solo tenga que seleccionar en la interfaz de usuario mostrada en la figura I.1 el lugar en el que se encuentra y el sitio del edificio al que desea desplazarse, ocupándose el sistema de todo lo demás.

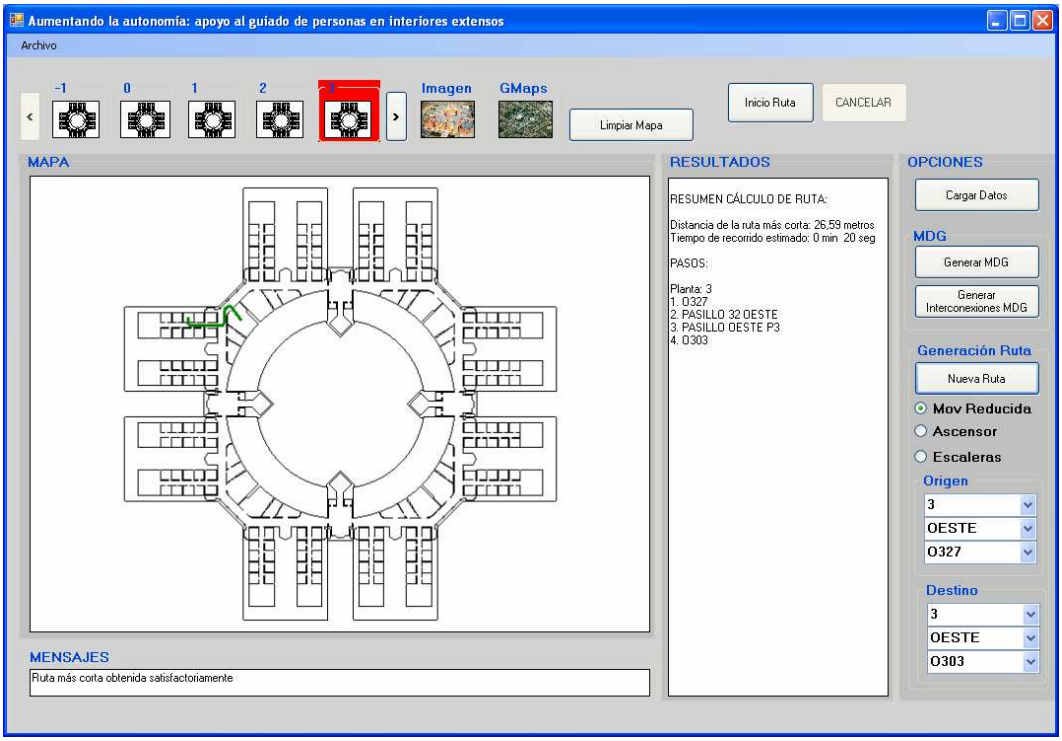


Figura I.1. Selección del de origen y el destino de la ruta.

Seleccionados los nodos de origen y destino, la aplicación calcula la ruta óptima para el usuario tal como se muestra en la interfaz de usuario, donde se dibuja la ruta sobre el mapa del edificio. Se incluye una ampliación de dicha ruta en la figura I.2, en la que se representan los puntos llamados nodos por los que esta definida.

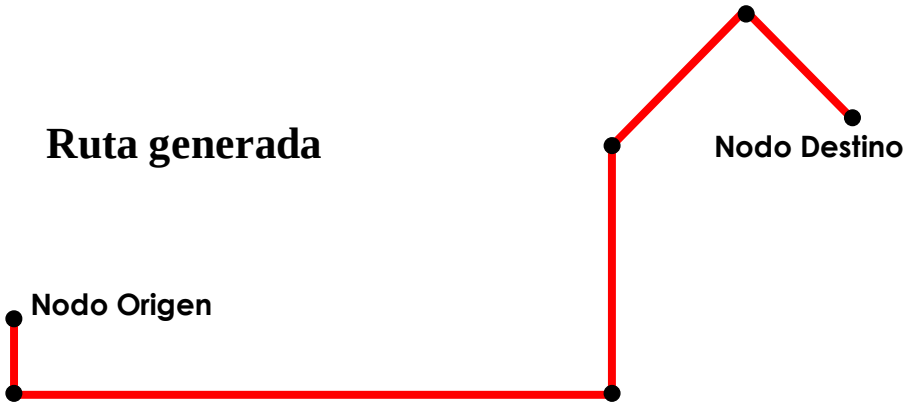


Figura I.2. Ruta óptima generada.

A partir de la ruta obtenida se calculan las sucesivas trayectorias que debe seguir la silla de ruedas hasta alcanzar el destino deseado. En la figura I.3 se representan dichas trayectorias, donde puede apreciarse que están formadas por tramos rectos y curvas de radio constante.

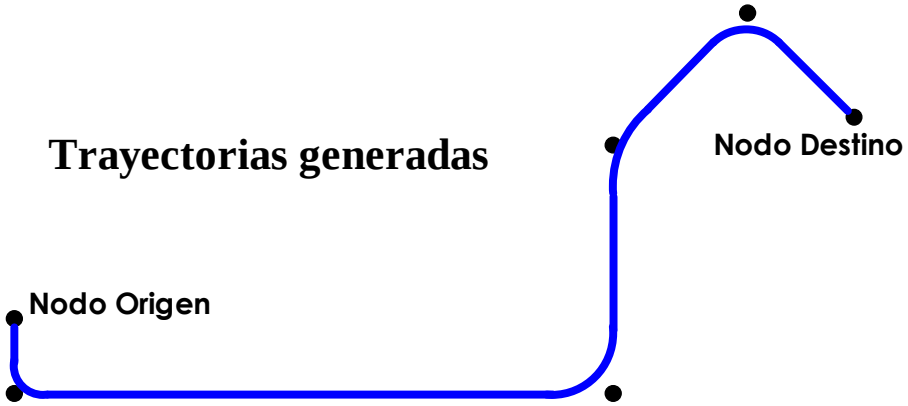


Figura I.3. Trayectorias calculadas para recorrer la ruta.

Una vez diseñados los desplazamientos que ha de realizar el móvil, se controlan sus movimientos mediante el controlador implementado y a través del modelo de cinemática odométrica de la silla.

Para llevar a cabo todas las tareas que componen la navegación autónoma se ha desarrollado un sistema de algoritmos sencillos que corren sobre el PC empotrado del que dispone la silla y que permiten de manera fácil alcanzar los objetivos buscados en este trabajo: que la precisión de los movimientos realizados por la silla quede dentro de un margen asumible. Además, el sistema diseñado consigue que la circulación de la silla de ruedas autónoma por las distintas zonas del entorno resulte lo más cómoda posible, realizando movimientos suaves y trayectorias semejantes a las que utilizaría cualquier

usuario de a pie, factor muy importante a tener en cuenta debido a los usuarios hacia los que está orientado el proyecto.

II. MEMORIA

1. Introducción.

En la memoria que aquí se presenta, se describe el trabajo realizado para el desarrollo de un sistema de navegación autónomo en interiores parcialmente estructurados. La plataforma móvil empleada para la implementación del sistema es una silla de ruedas, la cual, como se verá más adelante, fue dotada en anteriores trabajos con todos los elementos necesarios para poder controlar el movimiento del sistema, y por tanto seguir la trayectoria que el sistema diseñado plantee en la tarea de navegación autónoma.

El presente proyecto constituye la continuación del trabajo fin de master (TFM) “Servicio de Navegación para Interiores Extensos” [01], en el que se daba solución a las tareas de creación de las rutas y de puntos intermedios a seguir dentro del entorno escogido para ir de un origen a un destino conocidos, y emplea como base de las tareas de seguimiento a implementar el trabajo [02]. Se pretenden adaptar a las nuevas condiciones de la plataforma hardware y software incluidas en [01] a lo expuesto en [02] y realizar mejoras en los algoritmos de control desarrollados en el último para la navegación autónoma, cómoda y robusta de la silla de ruedas en un entorno conocido.

En el segundo capitulo que se incluye a continuación, se realiza una presentación del estado actual de la silla, explicando las partes que la componen y como trabajan entre sí. También se comentan distintos trabajos realizados con ella anteriormente, que serán empleados como referencia en este proyecto.

El capitulo tercero explica como se realiza la identificación off-line del sistema de bajo nivel, formado por los motores y sus drivers de excitación, empleando para ello los medios de los que ya esta provista la silla, como son la tarjeta de potencia que controla los motores o el Bus-CAN para las comunicaciones desde el PC a esta tarjeta. Este modelo permite simular y ajustar de forma fiable el funcionamiento del resto de algoritmos que componen el sistema de navegación autónomo, que se ejecutan en una tarjeta PC embarcada en la silla.

La generación de rutas, trayectorias para unir los puntos que conforman la ruta y el control de movimiento de la silla en el seguimiento de cada trayectoria, son tratados en el cuarto capítulo. Se realizará, por tanto, en éste una breve descripción del método de obtención de la ruta entre un punto origen y otro de destino elegidos por el usuario, según los procedimientos descritos en [01], y se usa esta ruta como punto de partida para el resto de tareas cuyo funcionamiento se detallara también en este capítulo.

Finalmente, en el quinto capitulo se presentan los resultados obtenidos en la tarea de navegación autónoma, realizando un análisis de éstos para algunas de las rutas llevadas a cabo por distintas partes del edificio.

2. Contexto de Trabajo.

En este capitulo se hace una descripción del estado en el que se encuentra la plataforma móvil empleada a la hora de acometer este trabajo. Inicialmente se presenta todo el hardware y la forma en que este se encuentra instalado, para a continuación pasar a realizar una pequeña mención de los trabajos realizados con anterioridad con la silla que le han permitido alcanzar el estado actual.

2.1. Estado actual de la silla de ruedas.

Para el desarrollo del proyecto, el principal elemento que tenemos es una silla de ruedas eléctrica alimentada por baterías, a la cual se le han instalado distintos elementos en los sucesivos trabajos que se han realizado con ella.

El control de la silla puede realizarse mediante el uso de un joystick, un interfaz de soplido o el PC del que está provista. El sistema de sensores cuenta con acelerómetros para detectar posibles impactos, sensores de proximidad por ultrasonidos para detectar obstáculos durante la navegación y encoders para detectar el giro de las ruedas y tener una realimentación sobre el movimiento de la silla en cada instante. Como actuadores, están los propios motores y la placa de potencia que los maneja. El sistema de proceso que tiene de entrada las señales sensóricas comentadas, y de salida los actuadores descritos es un PC empotrado.

Como se muestra en la figura 2.1, cada uno de los anteriores elementos mencionados son gobernados por distintos microcontroladores que se

interconectan entre sí y con el PC de control mediante el uso de protocolo Bus-CAN.

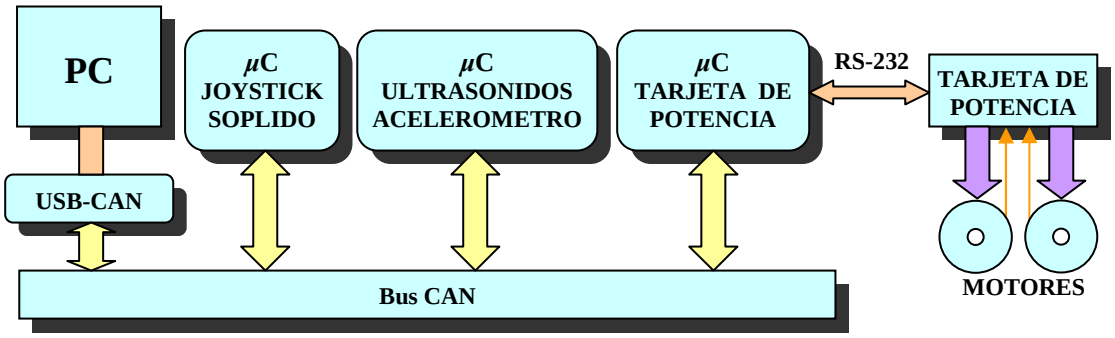


Figura 2.1. Arquitectura hardware de la silla.

En los sucesivos párrafos se describen únicamente aquellos componentes de la silla usados en el desarrollo del proyecto, por lo que los módulos de joystick, acelerómetro, sensores de soplido y ultrasonidos, no serán descritos.

2.1.1. Bus-CAN y conversor USB.

El Bus-CAN (Bus-Controller Area Nentwork), es un protocolo de comunicaciones serie de uso extendido en automatización. Permite transportar una cantidad moderada de información de manera segura y reduciendo el cableado, entre un numero indeterminado de dispositivos asociados a la misma red.

La información transmitida entre los distintos dispositivos conectados al sistema se propaga a través de los dos cables que forman un par trenzado como se muestra en la figura 2.2.En el que en uno de los hilos del par los valores de tensión varían entre 0V y 2.25V, por lo que se denomina CAN L (Low) y en el otro, el CAN H (High), lo hacen entre 2.75V. y 5V. De este modo, si cualquiera de las líneas se interrumpe o se deriva a masa, el sistema puede trabajar con la otra y respecto a masa.

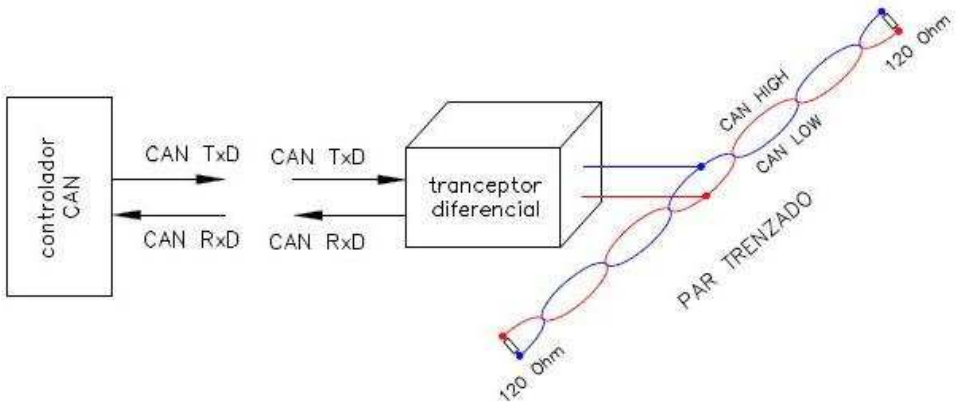


Figura 2.2. Esquema del bus CAN.

La manera en que se realiza el intercambio de información es en forma de paquetes con una longitud limitada y con una estructura definida en campos que conforman los mensajes, tal como muestra la figura 2.3. Uno de esos campos actúa de identificador del tipo de dato que se transporta, de la unidad de mando que lo trasmite, y de la prioridad para trasmitirlo respecto a otros mensajes. Así, los mensajes no van direccionados a ninguna unidad en concreto, si no que cada una de ellas reconocerá mediante este identificador si el mensaje recibido le interesa o debe ignorarlo, de modo que todas las unidades puedan ser trasmisoras y receptoras.

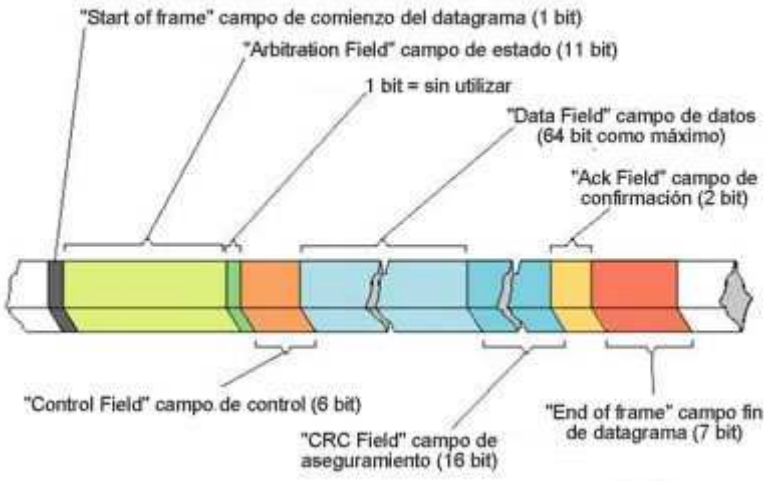


Figura 2.3. Estructura de un mensaje CAN.

Los mensajes CAN empleados están formados por un identificador único para cada mensaje y ocho bytes para datos. Hasta el momento los mensajes que circulan por el bus de la silla son los mostrados en la tabla 2.1, cuya información se extrae de [03] y [04].

IDENTIFICADOR	DATO A (32bits)		DATO B (32bits)		
0x101 (T.Pot)	Tiempo Motor Derecho		Encoder ABS Derecho		
0x102 (T.Pot)	Tiempo Motor Izquierdo		Encoder ABS Izquierdo		
0x110 (Joy)	Potenciómetro	Eje X joystick	Eje Y joystick	Wd	Wi
0x111 (Joy)					Modo
0x120 (PC)				Wd	Wi
0x201 (Sen)	SLDD	SLIT	SDI	STD	
0x202 (Sen)	SLDT	STI	SLID	SDD	
0x203 (Sen)	Acel. X	Acel. Y	Acel.Z	SJOYS	
0x210 (T.Pot)	Tensión del controlador		Tensión de la batería		

Tabla 2.1. Mensajes a través del Bus-CAN.

Leyenda de la tabla 2.1:

***T.Pot:** Tarjeta de potencia.*

***Joy:** Joystick.*

***PC:** Personal Computer.*

***Sen:** Sensores.*

Sensores de ultrasonidos: SLDD, SLIT, SDI, STD, SLDT, STI, SLID, SDD, SJOYS.

Acelerómetro: Acel. X, Acel. Y, Acel.Z.

Como en este proyecto no se empleará todo el hardware instalado en la silla, a continuación se describen únicamente aquellos mensajes de los cuales si se hará uso:

- **Mensaje 0x101 y 0x102:** Los primeros 32 bits de estos mensajes son empleados para almacenar la marca de tiempo del instante en el que se realiza la lectura de los encoders. Los otros 32 bits restantes representan el valor incremental del número de pulsos leídos en los encoders. Por tanto, su valor estará comprendido entre -2147483648 y +2147483647. El identificador 0x101 corresponde al motor derecho y 0x102 al izquierdo. Estos mensajes son volcados al bus por la tarjeta de potencia cada 100 milisegundos por medio de la electrónica que esta lleva asociada.

- **Mensaje 0x120:** Este mensaje generado por el PC únicamente hace uso de dos bytes, uno para mandar el comando de velocidad a la tarjeta de potencia

correspondiente al motor derecho y otro para el motor izquierdo. Sus valores estan acotados entre 0 y 255.

Para conectar el PC al Bus-CAN y poder así comunicarse con el resto de elementos de la silla, se dispone de un conversor Bus-CAN / USB de la marca LAWICEL. Así, mediante el driver y librerías proporcionadas por el fabricante, las comunicaciones entre el ordenador y el resto del hardware no resultan complejas.

2.1.2. Microcontroladores.

Los distintos elementos instalados en la silla como el acelerómetro, los sensores de ultrasonidos, el sensor de soplido, el joystick y la tarjeta de potencia son manejados por las tarjetas LPC2129 de Embedded Artists que se muestra en la figura 2.4, las cuales están basadas en el microcontrolador NXP’s ARM7TDMI LPC2129. A continuación se enumeran las principales características técnicas de estas:

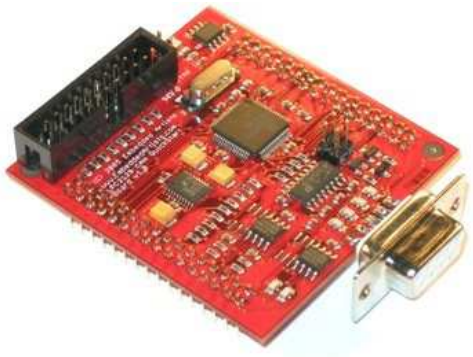


Figura 2.4. Tarjeta LPC2129.

- Memoria FLASH: 256Mb.
- Memoria SRAM: 16Kb.
- CAN: Dos canales con transmisor TJA1040 ó TJA1041.
- Reloj: 12MHz para máxima velocidad de ejecución y velocidad estándar de CAN.
- Dimensiones: 55x58mm.
- Alimentación: 5V DC.
- Conectores: 2x16 puertos de entrada/salida, RS232, DSUB-9.
- Otros: 256Kb de memoria E2PROM para I2C.
- 4 capas de PCB para una mejor inmunidad al ruido.

Cada una de estas tarjetas se conectan al Bus-CAN, de manera que son capaces de componer los mensajes para este protocolo con los datos provenientes de los sensores u otros elementos que tienen bajo su control, y enviarlos por el bus para que el dispositivo que tenga que hacer uso de ellos los reciba (entre ellos el PC embarcado en la silla gracias al adaptador CAN-USB comentado). De igual manera, son capaces de tomar los mensajes que reciben por el mismo bus y determinar como actuar ante estos.

2.1.3. Tarjeta de potencia y encoders.

La tarjeta de potencia usada es una AX3500 de ROBOTEQ (en la figura 2.5) y es la encargada de controlar los motores de la silla de ruedas a partir de las órdenes que recibe de la tarjeta de microcontrolador que la enlaza al Bus-CAN.

Figura 2.5. Tarjeta de potencia AX3500.

La tarjeta está diseñada para trabajar con niveles de alimentación entre 12V y 24V, además de soportar corrientes de hasta 60A. Permite controlar dos motores simultáneamente y trabajar con ellos en lazo abierto. Las principales características técnicas de esta tarjeta son:

- Voltaje de funcionamiento: 12V-24V DC.
- Número de canales de lectura de encoder: 2.
- Corriente máxima: 60A.
- 8 MOSFETs por canal.
- Resistencia de ON: 5mΩ.
- Rectificación síncrona.
- Límite de corriente por la reducción automática de la potencia de salida en función de la carga y el tiempo transcurrido.
- Temperatura de protección: 80ºC.

- Protección de voltaje apagando la salida por debajo de 13V y por encima de 43V.

En la imagen de la figura 2.6 se muestra el conexionado realizado en la tarjeta de potencia acoplada a los motores de la silla de ruedas, donde cada una de las partes enumeradas se corresponde con las nombradas a continuación:

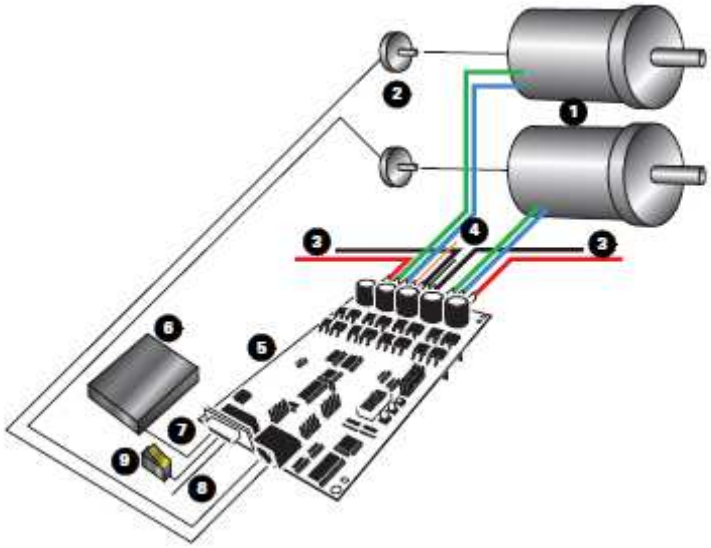


Figura 2.6. Conexionado de la tarjeta de potencia.

1-Motores de continua.
2-Encoder óptico.
3-Alimentación de potencia para los motores.
4-Alimentación para el controlador.
5-Tarjeta AX3500.
6-Unidad controladora.
7-Interfaz RS-232.
8-Otras entradas.
9-Interruptor de parada.

Como puede verse, el excitador de los motores incluido en la tarjeta de potencia tiene una entrada de alimentación independiente para cada motor y otra propia para la parte de control de la placa.

También se observa el conector a través del cual la tarjeta se comunica con distintos dispositivos. Esta comunicación la realiza mediante el protocolo serie

RS-232, y es la forma que se emplea para el intercambio de datos entre esta tarjeta de potencia y la tarjeta de microcontrolador que la gobierna, y que la conecta con el Bus-CAN-USB.

El otro elemento a conectar a esta tarjeta son los dos encoders ópticos incrementales de dos canales en cuadratura empleados para poder determinar la velocidad de giro de los respectivos motores. Estos encoders están formados por un disco circular como el que se muestra en la figura 2.7 que está acoplado al eje de motor, y que cuenta con perforaciones en su perímetro que son atravesadas por dos haces de luz, convertidos en señales cuadradas a través de circuitos optoelectrónicos, de modo que por sus 2 canales de salida se generen señales de frecuencia proporcional a la velocidad de movimiento del motor.

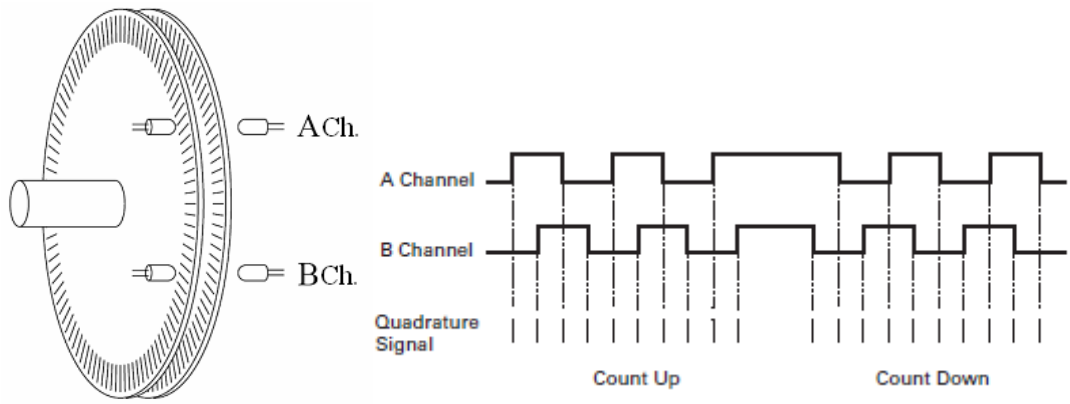


Figura 2.7. Encoder incremental de dos canales y sus pulsos generados.

La tarjeta de potencia dispone de un contador para cada encoder de 32 bits con signo que se incrementa o decrementa con cada flanco de alguno de los dos canales de los encoders conectados a ésta, por lo que tras completarse una vuelta, al disponer el encoder empleado de 200 perforaciones, se realizarán un total de 800 cuentas.

Esta tarjeta también cuenta con un controlador clásico (PID) independiente para cada motor, con los que es posible conseguir que la respuesta de los motores se ajuste a una respuesta deseada. Los valores que se emplean para este controlador provienen de los trabajos realizados anteriormente con la silla, y no han sido modificados pues la respuesta ofrecida por los motores resulta satisfactoria, de modo que son los que se incluyen en el modelo del sistema de bajo nivel identificado en este proyecto (ver capítulo siguiente).

Los valores de las constantes de control K_P, K_I y K_D son los siguientes:

- K_P, Ganancia Proporcional = 2,0
- K_I, Ganancia Integral = 1,5
- K_D, Ganancia Diferencial = 0,0

2.2. Trabajos previos en los que se sustenta este proyecto.

Descritos los principales elementos instalados en la silla, de los que se hace uso en este trabajo, a continuación se realiza una breve mención de los desarrollos realizados con ella que forman parte de las bases de las que parte el presente proyecto.

2.2.1. Preparación de la plataforma hardware.

Como se comentó anteriormente, nuestro móvil consiste en una silla de ruedas eléctrica convencional, a la que se le ha agregado el resto de elementos descritos.

La instalación y programación del sistema de sensores y actuadores, buses de comunicación, tarjeta de potencia y cada una de las tarjetas con microcontrolador que realiza algún tipo de proceso sobre el resto de componentes, fue llevada a cabo en los proyectos [03] y [04], de modo que tras su finalización la silla quedó en pleno funcionamiento y disponible para continuar realizando trabajos con ella. Así pues, no ha sido necesario desarrollar nada a nivel de hardware y software a bajo nivel.

2.2.2. Identificación de los motores.

Respecto a la identificación del modelo de velocidad los motores de la silla, existe ya una identificación previa desarrollada en el proyecto [05]. En él se realizó la identificación paramétrica basada en estructuras ARX y OE exclusivamente del comportamiento de los motores en tiempo discreto, empleando un hardware diseñado única y exclusivamente para tal efecto.

Los resultados y conclusiones de dicho proyecto serán útiles para contrastar los obtenidos en el presente, en los que el propósito es subir de nivel de abstracción en la identificación para obtener un modelo de comportameinto discreto de todo

el bajo nivel (controlador de velocidad implementado por las tarjetas de potencia, como se explica en el punto 2.1.3 anterior, y modelo de velocidad de los motores), tal y como se explica en el capítulo siguiente de esta memoria.

2.2.3. Control y guiado.

Referente al control de trayectorias, existen gran variedad de trabajos sobre el tema. El que se utilizará como principal referencia en el proyecto será el trabajo [02], por emplear como plataforma móvil una silla de ruedas semejante a la empleada en el actual trabajo, pero partiendo de distinta estructura hardware. El hecho de usar el mismo dispositivo móvil hace que el comportamiento dinámico del sistema sea similar, por lo en el tema 4, en el apartado referente a la generación y control de trayectorias, se realizarán distintas reseñas a este.

2.2.4. Generación de rutas.

Como se explica en el capítulo de introducción, el fin de este trabajo es que la silla de ruedas sea capaz de seguir lo más fielmente posible y de forma autónoma una serie de trayectorias que unan dos puntos de un mapa, definidos como origen y destino de una ruta. Estas rutas serán generadas según el proyecto [01], tal como se menciona también en la introducción, por lo que la implementación de los procesos para su obtención queda al margen de los desarrollos acometidos en este proyecto.

Tras obtenerse la ruta como se muestra en la figura 2.8, dan comienzo las distintas tareas y procesos desarrollados en el presente trabajo que hacen posible que la silla de ruedas alcance satisfactoriamente su destino.

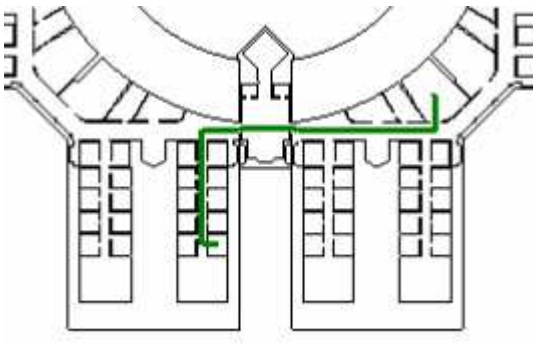


Figura 2.8. Generación de ruta.

3. Identificación del Sistema en Movimiento.

En este trabajo se realiza el estudio y la obtención del modelo para el sistema de bajo nivel de la silla, formado por los motores y el driver excitador, para que junto con el resto de los procesos implementados en la generación y seguimiento de trayectorias, que se detallaran en el siguiente capítulo, se disponga de todo el sistema de la silla completamente definido.

3.1. Introducción.

Para abordar la tarea de identificación del sistema bajo estudio, previamente se expone una breve presentación acerca de la teoría sobre los procesos de identificación y los procedimientos a realizar para obtener un modelo de un sistema.

Un modelo es una herramienta para describir un sistema, de modo que permite predecir su comportamiento a partir de sus variables de entrada conocidas. En este caso, al tratarse de motores, dichas variables son la tensión de excitación, que junto con el modelo adecuado, hacen posible prever la velocidad de giro del motor en cada instante.

Posteriormente, se detallan los procesos realizados en el actual trabajo para conseguir el modelo del bajo nivel que mejor describa su comportamiento, mostrándose los resultados obtenidos en dichas tareas.

3.2. Tipos de modelo.

El objetivo de este estudio es conocer y representar un sistema a través de un modelo que lo represente, de modo que quede descrito su comportamiento. Según su formalismo matemático, destacan dos grandes grupos:

a) Modelos no parametricos: Son los que representan un sistema sin hacer uso de expresiones matemáticas, empleando en su lugar gráficos o tablas con valores finitos, en las que el comportamiento queda suficientemente determinado.

b) Modelos paramétricos o matemáticos: Emplean expresiones matemáticas, como ecuaciones diferenciales o en diferencia, lo que permite modelar sistemas más complejos. Estos a su vez pueden clasificarse en:

   b.1) Determinísticos o estocásticos: Un modelo es determinísticos cuando esta definido de forma exacta por una ecuación. Por el contrario, se trata de un sistema estocástico si esta definido por conceptos probabilísticos o estadísticos, por lo que poseen cierto grado de incertidumbre.

   b.2) Dinámicos o estáticos: Los sistemas dinámicos son aquellos en los que en sus salidas influye el tiempo, es decir la salida en un instante, depende de la entrada en ese instante y en los instantes de tiempo anteriores. Esto no ocurre en los sistemas estáticos, cuya salida en un determinado instante solo depende de la entrada en ese momento.

   b.3) Continuos o discretos: Modelos continuos son los que trabajan con señales continuas, por lo que suelen definirse a través de ecuaciones diferenciales. Los modelos discretos son los que usan señales discretas y emplean ecuaciones en diferencia.

3.3. Obtención de un modelo.

Un modelo se pude obtener principalmente de dos métodos:

a) Modelado teórico: en el que se recurre a leyes físicas para describir el comportamiento de un proceso. Únicamente es aplicable a procesos muy sencillos, o que requieran poca exactitud.

b) Identificación del sistema: se obtiene un modelo del sistema a partir de datos recogidos de las salidas como resultado de procesos experimentales. En este caso es de gran ayuda también tener algún conocimiento físico de la planta a estudio.

3.3.1. Proceso de identificación.

Para conseguir una identificación del sistema correcta, es preciso seguir los pasos que se describen a continuación:

a) Obtención de datos de entrada – salida: se han de registrar las salidas del sistema, a la vez que se aplican en sus entradas señales conocidas que también han de ser almacenadas.

b) Tratamiento previo de los datos registrados: los datos obtenidos en el paso anterior, pueden presentar alteraciones como ruido, niveles de continua, etc. Para disponer de datos adecuados para el modelado será necesario acondicionarlos, eliminando en lo posible las imperfecciones que puedan presentar.

c) Elección de la estructura del modelo: como se mencionó en el apartado anterior, conocer las leyes físicas que gobiernan el proceso ayuda en gran medida en la elección de la estructura final que tendrá el modelo.

d) Obtención de los parámetros del modelo: Consiste en obtener los parámetros de la estructura elegida, a partir de los datos obtenidos experimentalmente y de distintos algoritmos matemáticos.

e) Validación del modelo: El modelo obtenido, pude no ser satisfactorio con respecto a lo exigido. Esto puede estar provocado por que los datos registrados en las entradas y salidas no aportan la información necesaria del comportamiento del sistema, la estructura elegida no es la adecuada para el modelo o porque no se ha elegido bien el criterio de ajuste de parámetros.

3.3.2. Identificación paramétrica.

La identificación paramétrica, permite identificar un modelo mediante el uso de una estructura y un número finito de parámetros, que relacionan las señales de entrada, salida y perturbaciones.

Si el sistema real responde a comportamientos lineales e invariantes en el tiempo, es posible realizar su modelado, según sea su naturaleza, mediante ecuaciones diferenciales o en diferencia, siendo la mayoría de los modelos paramétricos descritos en el dominio discreto, debido a que la obtención de los datos para la identificación se realiza mediante muestreo de las distintas señales empleando un determinado periodo.

En muchos casos, se debe realizar la identificación de sistemas de los cuales no se tienen conocimientos. Para ello se emplean modelos estándar, validos para un gran rango de sistemas dinámicos de manera experimental. La dificultad en el empleo de estos modelos se encuentra en determinar el tipo de modelo, definido por su orden, el número de parámetros, su valor, etc.

La estructura genérica de estos modelos es:

$$A(q^{-1}) \cdot y(t) = \frac{B(q^{-1})}{F(q^{-1})} \cdot u(t) + \frac{C(q^{-1})}{D(q^{-1})} \cdot e(t) \qquad <3.1>$$

En ella $u(t)$ representa la señal de entrada, $e(t)$ la señal de ruido acoplado al sistema, e $y(t)$ la señal de salida. $A(q^{-1}), B(q^{-1}), C(q^{-1}), D(q^{-1}), F(q^{-1})$ son los polinomios de los cuales se debe determinar su orden y posteriormente obtener el valor de sus coeficientes que hacen que el modelo se ajuste al sistema real. También es necesario determinar el retardo existente entre la señal de entrada y la de salida.

En muchos casos, alguno de los polinomios antes mencionados no son incluidos en el modelo, dando lugar a estructuras más sencillas como es el caso de los modelos ARX, OE, ARMAX, BJ, en los que resulta más sencillo el proceso de ajuste de parámetros. En la tabla 3.1 se detallan estos cuatro modelos nombrados.

La identificación del modelo usando estas estructuras quedará ampliamente facilitada por el uso herramientas software como la ToolBox de Identificación de Matlab.

TIPO DE MODELO	CONDICION	ESTRUCTURA RESULTANTE
ARX	$C(q^{-1}) = D(q^{-1}) = F(q^{-1}) = 1$	$A(q^{-1}) \cdot y(t) = B(q^{-1}) \cdot u(t) + e(t)$
OE	$A(q^{-1}) = C(q^{-1}) = D(q^{-1}) = 1$	$y(t) = \frac{B(q^{-1})}{F(q^{-1})} \cdot u(t) + e(t)$
ARMX	$D(q^{-1}) = F(q^{-1}) = 1$	$A(q^{-1}) \cdot y(t) = B(q^{-1}) \cdot u(t) + C(q^{-1}) \cdot e(t)$
BJ	$A(q^{-1}) = 1$	$y(t) = \frac{B(q^{-1})}{F(q^{-1})} \cdot u(t) + \frac{C(q^{-1})}{D(q^{-1})} \cdot e(t)$

Tabla 3.1. Algunos tipos de modelos.

3.4. Proceso de identificación de los motores.

En este apartado se detallarán los distintos procesos realizados para obtener la identificación del sistema de bajo nivel.

3.4.1. Obtención de las muestras.

Para obtener el modelo de la planta del sistema, en nuestro caso los motores de la silla, se debe de conseguir las muestras de las señales que permitan realizar la identificación. Estas señales son Vcod como señal de entrada y W como señal de salida, ambas representadas en la figura 3.1. Cada motor tendrá su correspondiente par de señales, donde Vcod es el código de excitación de los motores enviado a la placa de potencia y W representa la velocidad de de giro de estos.

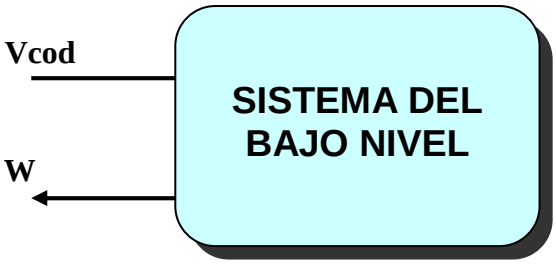


Figura 3.1. Entrada y salida del sistema de bajo nivel.

Para la obtención de estas, se crea un perfil de velocidades que es empleado como entrada del sistema. Para ello se desarrolla en el entorno de programación Visual Studio una aplicación en el lenguaje C-Sharp (C#) que se encargara de toda la gestión del proceso. Dicha aplicación que corre sobre un PC, toma de un archivo fuente los datos de los perfiles a crear, construye con ellos los mensajes CAN necesarios y se los manda a la silla de ruedas por medio del Bus-CAN.

3.4.1.1. Descripción del archivo fuente.

El archivo fuente es un archivo de texto sin formato, el cual siguiendo un prototipo de archivo, es capaz de indicar a la aplicación la consigna que se ha de enviar a los motores en cada momento. De esta forma se configura el perfil de escalones que se desea aplicar.

También cuenta con una etiqueta que indica el archivo destino donde se han de guardar los datos que se obtengan del experimento, además de otra que indica el periodo de muestreo que se empleará para realizar las lecturas de las marcas de los encoders y otra etiqueta que indica la carga de peso manejada por la silla para dicha prueba. A continuación en la imagen de la figura 3.2 podemos ver un ejemplo de un archivo fuente.

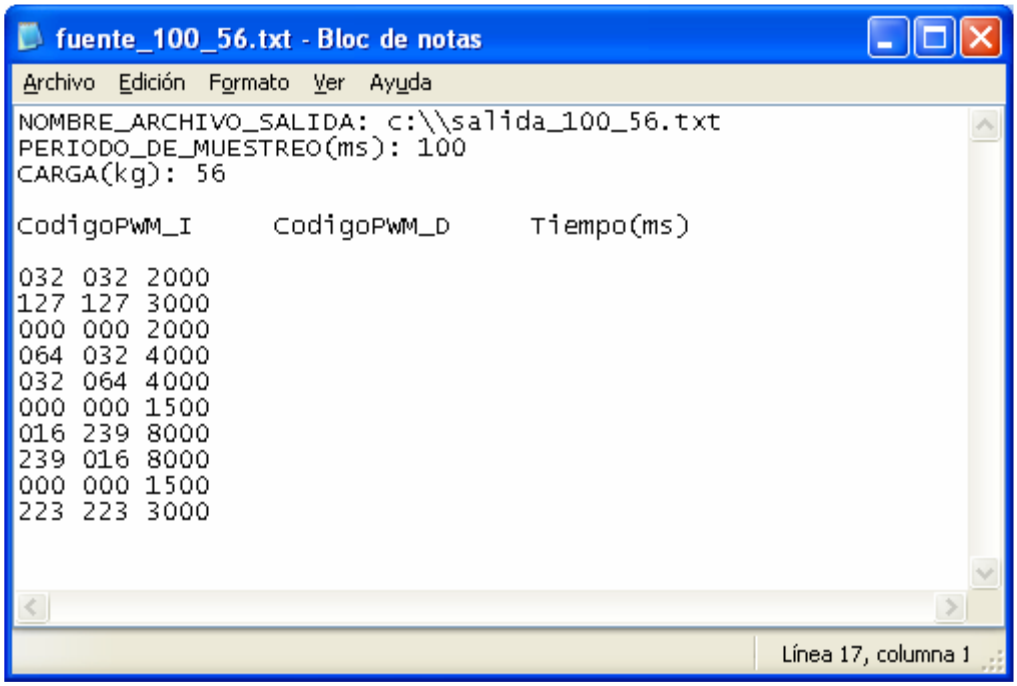


Figura 3.2. Ejemplo de un fichero fuente.

En las tres primeras líneas aparecen las etiquetas informativas y a continuación, las dos primeras columnas son el código a enviar a la tarjeta de potencia para excitar cada uno de los motores mientras que la tercera indica el periodo de tiempo, en milisegundos, durante el cual los motores estarán a ese nivel de trabajo.

3.4.1.2. Descripción de la aplicación de toma de muestras.

Al iniciarse el proceso, la aplicación toma del archivo fuente indicado el nombre del archivo donde se almacenarán los resultados y el periodo de muestreo que se ha de emplear. Estos valores, así como el valor de la carga transportada por la silla y el dato que se envía en cada momento a los motores son mostrados en cada campo de la interfaz gráfica.

A continuación la aplicación toma de dicho archivo el código de velocidad para cada motor y el tiempo durante el que tiene que mantener ese valor y se los envía a la silla, de manera que los motores han de girar a esa velocidad durante ese tiempo. Transcurrido este periodo se envían los siguientes valores, produciendo que los motores giren a distintas velocidades en un sentido u otro.
Al completarse el último periodo dado por el archivo fuente el proceso habrá llegado a su fin.

La captura de los encoders da comienzo en el momento en que se inicia el envío de datos a los motores de la silla y se realiza de manera periódica según lo indicado por la correspondiente etiqueta del archivo fuente.

La interfaz grafica creada para la aplicación es la que se muestra en la figura 3.3.

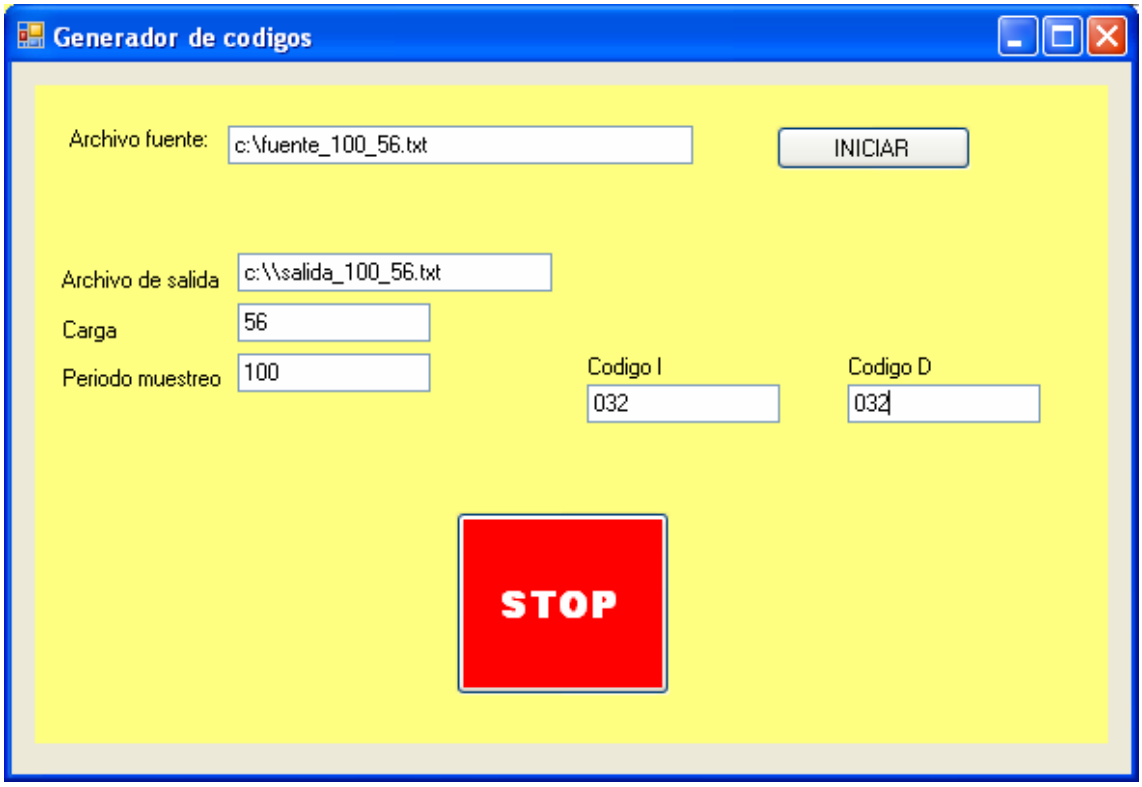


Figura 3.3. Interfaz de la aplicación para identificaciones.

3.4.1.3. Descripción operativa.

Una vez arrancada la aplicación se le debe indicar mediante su correspondiente campo el archivo fuente que se desea emplear para el proceso y posteriormente se activará el proceso de adquisición actuando sobre el botón **"INICIAR"**. Si en cualquier momento es necesario parar su transcurso, bastara con pulsar sobre **"STOP"** y la aplicación se detendrá así como la silla de ruedas. Para continuar, será necesario empezar de nuevo todo el proceso de adquisición, no pudiendo continuar desde el punto en que se abortó. Se ha realizado así ya que de lo contrario, al arrancar de nuevo la silla, los motores no dispondrían de la inercia con la que contaban cuando se paró, lo que influiría en el proceso de identificación. Una vez finalizado el proceso, la silla quedará finalmente en reposo.

3.4.1.4. Descripción del archivo de salida.

El archivo de salida que se obtiene tras los experimentos es otro fichero de texto sin formato que esta formado por siete columnas, un ejemplo es el mostrado en la figura 3.4.

La estructura de estos ficheros es la descrita en las siguientes líneas:

- **Primera columna:** Código enviado para el motor izquierdo.
- **Segunda columna:** Etiqueta del encoder izquierdo.
- **Tercera columna:** Etiqueta de tiempo.
- **Cuarta columna:** Dato enviado para el motor derecho.
- **Quinta columna:** Etiqueta del encoder derecho.
- **Sexta columna:** Etiqueta de tiempo.
- **Séptima columna:** Periodo de muestreo empleado.

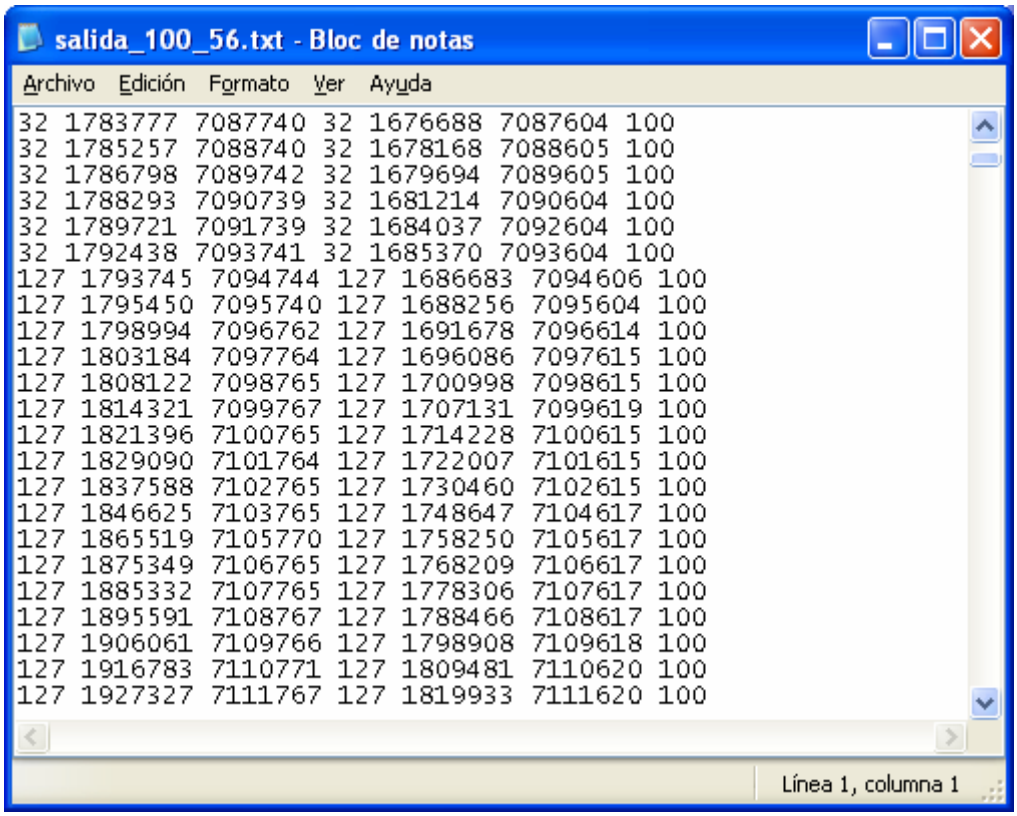


Figura 3.4. Ejemplo del fichero de salida.

En cada línea se encuentran las sucesivas lecturas que se realizan. Como los valores de las lecturas de los encoders y las marcas de tiempo son acumulativos, para poder representar los datos en una grafica y ser capaces de ver como

responden los motores a los distintos escalones, será preciso hacer un tratado de los datos antes de visualizarlos.

3.4.2. Tratamiento previo.

Tras extraer las muestras del archivo destino y almacenarlas cada una en su correspondiente vector de muestras, se realiza el cálculo de la velocidad de giro de los ejes de los motores según la siguiente expresión:

$$VelocidaAngular(rad/seg) = \frac{P[n+1]-P[n]}{T[n+1]-T[n]} \cdot \frac{2\pi}{Np \cdot Tps} \quad <3.2>$$

Donde:

- **P[n]** representa el valor del contador de pulsos del encoder de la muestra **n** y **P[n+1]** el de la siguiente muestra.
- **T[n]** es la marca temporal de la muestra **n** y **T[n+1]** la de la siguiente.
- **Np** es el número de cuentas por cada vuelta del encoder. En nuestro caso 800, ya que genera 200 pulsos y por cada uno se producen 4 cuentas.
- **Tps** es la variable que permite pasar a segundos las marcas temporales. Su valor es 90,15·10-6.

A partir de lo expuesto anteriormente, haciendo uso de los datos almacenados en los ficheros de salida y junto con la ecuación <3.2> se obtienen los resultados que se presentan a continuación.

Como se observa en las grafica de la figuras 3.5 y 3.6, en las que se relaciona la respuesta del motor izquierdo y derecho respecto a sus correspondientes escalones de entrada, mediante el cursor de posición se puede ver que para un código de 127 que implica la máxima velocidad de los motores, tenemos una velocidad giro de 928,2rad/s para el motor izquierdo y 918,4rad/s para el derecho. También se observa como cuado el código de entrada supera el valor de 127, el sentido de giro de las ruedas cambia.

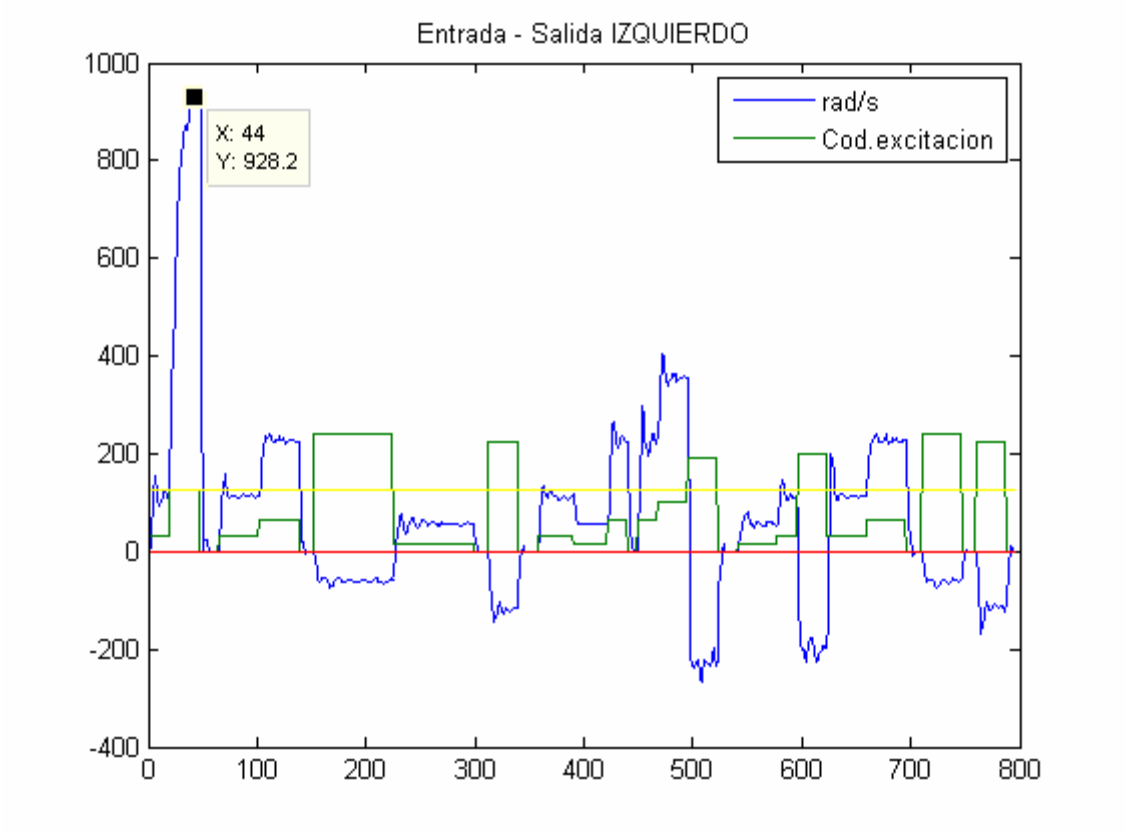


Figura 3.5. Respuesta del motor izquierdo a los impulsos de entrada.

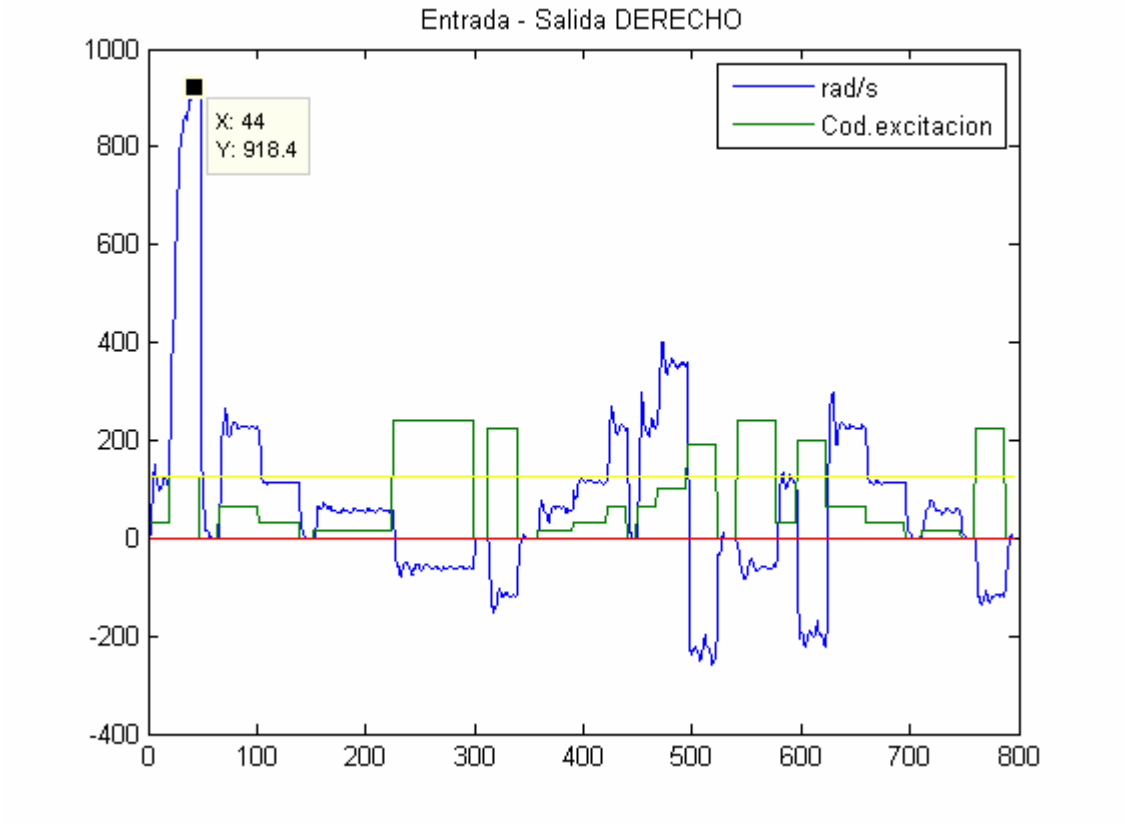


Figura 3.6. Respuesta del motor derecho a los impulsos de entrada.

El tratado que se realiza a las muestras es un rectificado para adaptar esos cambios y hacer que los resultados obtenidos sean válidos para seguir con la identificación. Así, se realiza la transformación “codigo=255-codigo” de los códigos de excitación comprendidos entre 128 y 255, ambos incluidos, y se invierte el signo a los datos de salida en los intervalos en que sean negativos, intervalos que por supuesto coinciden con los intervalos pertenecientes al rango de códigos descritos. Tras este procesado, los resultados de las muestras adquiridas quedan por tanto como se muestran en las graficas de las figuras 3.7 y 3.8.

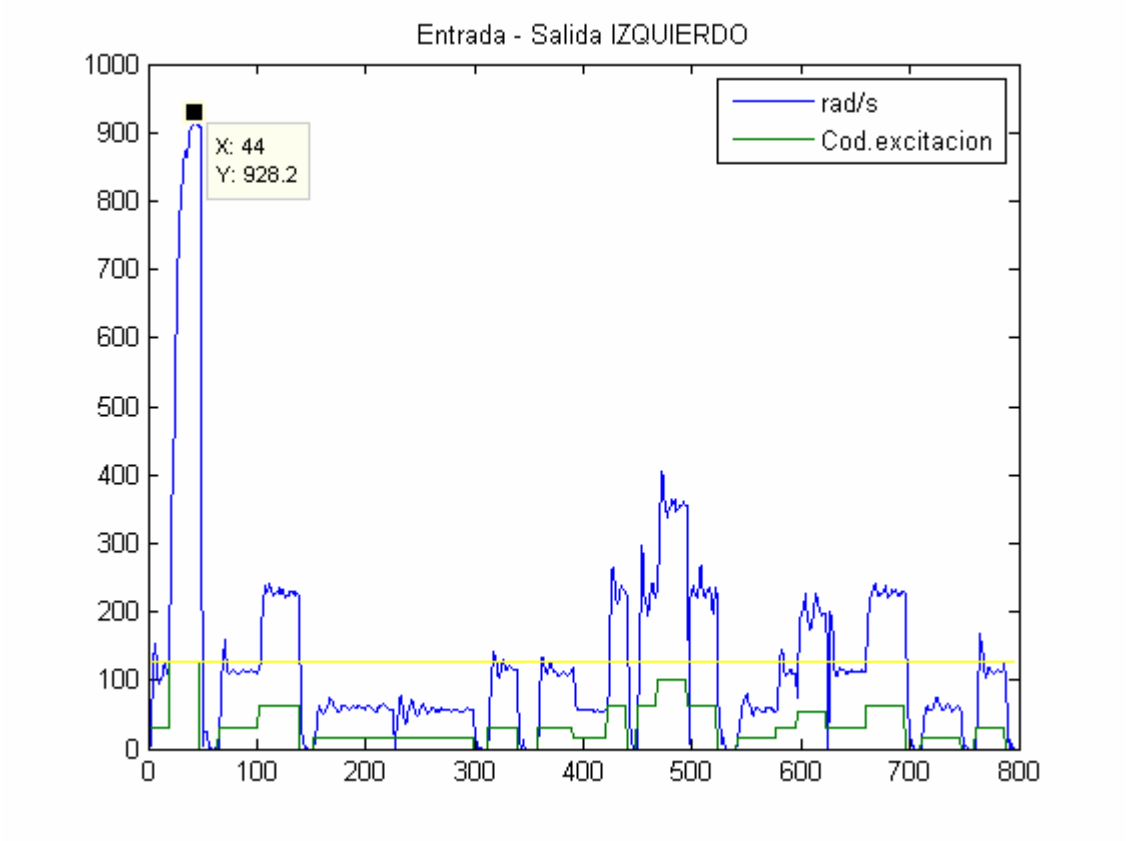


Figura 3.7. Respuesta del motor izquierdo tras el pretratado.

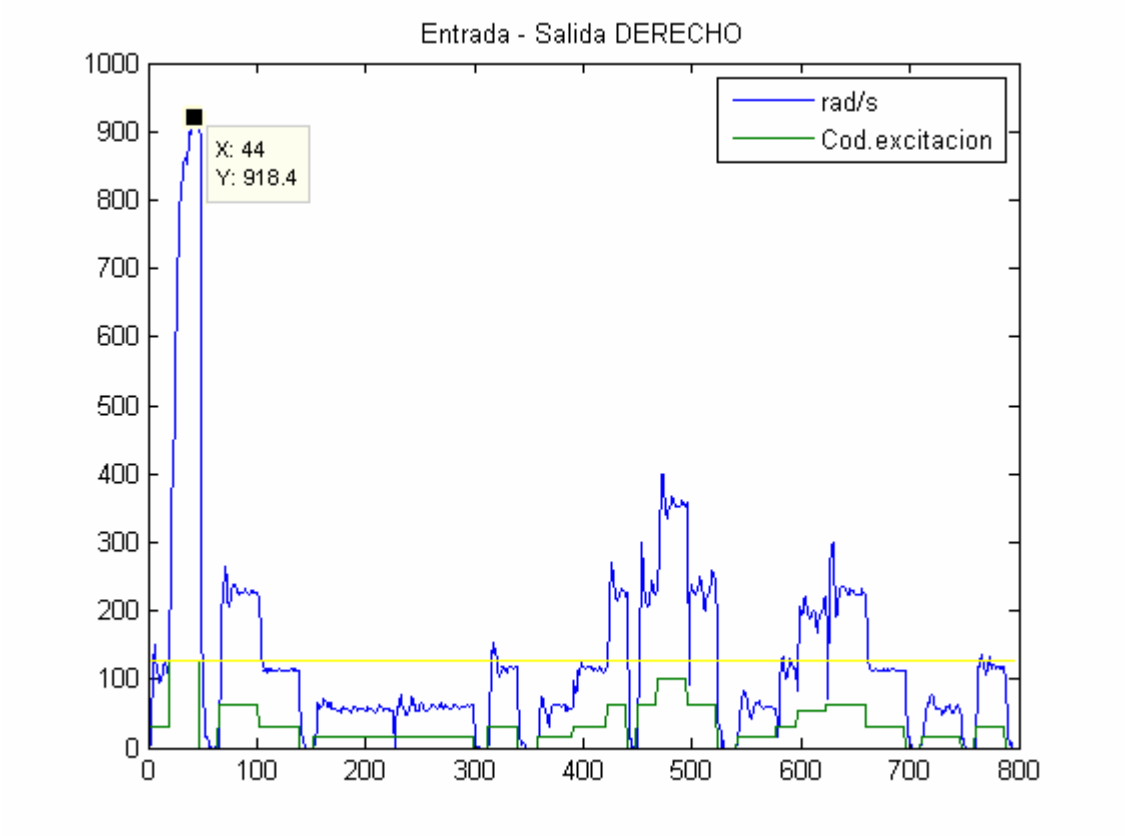


Figura 3.8. Respuesta del motor derecho tras el pretratado.

Dado que en las muestras no hay presencia de niveles de continua ni se observa un alto nivel de ruido, no es necesario realizar ningún otro tipo de pretratado a estas antes de continuar con el proceso de identificación.

3.4.3. Elección de la estructura.

Para elegir la estructura más adecuada para nuestro modelo, se hace uso de los conocimientos teóricos que se tienen del sistema. A partir del diagrama de la figura 3.9 que representa el esquema que modela el sistema del bajo nivel de cada motor, se procede para obtener la ecuación que lo describa.

En esta figura se muestran cada uno de los bloques que conforman todo el sistema del bajo nivel, formado como ya se dijo anteriormente por los motores, sus encoders y el driver de excitación. A continuación se describe cada uno de ellos.

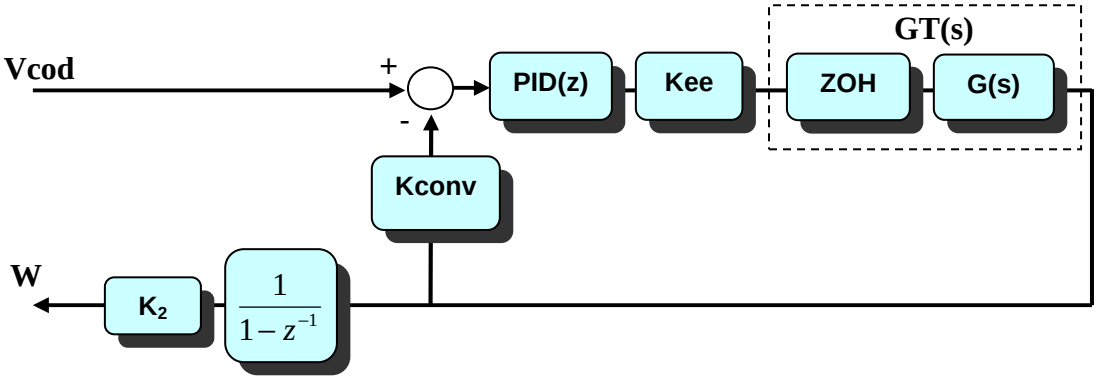


Figura 3.9. Diagrama del sistema del bajo nivel.

- **PID(z):** Es el controlador integrado en el driver de excitación de cada motor. Los valores de sus constantes para este trabajo son:

  Kp (constante parte proporcional) = 2

  Ki (constante parte integral) =1,5

  Kd (constante parte derivativa) = 0

- **Kee:** Constante para obtener la tensión en el motor a partir de un código (0-127). Viene dada por la siguiente ecuación, donde **Vn** es el voltaje nominal de los motores (24v) y **Fs** el fondo de escala (127 para nuestro caso).

$$Kee = \frac{Vn}{Fs} \qquad <3.3>$$

- **ZOH:** Representa un mantenedor de orden cero. Su uso se debe a que todo el sistema es discreto mientras que la planta de este, los motores, es un sistema continuo.

- **G(s):** Es el bloque correspondiente a la planta del sistema, es decir, los motores.

- **Kconv:** Constante del driver excitador de los motores que le permite a este obtener la realimentación para el valor de **Vcod** y poder realizar sus procesos internos para el control del bajo nivel [08].

$$Kconv = \frac{PPR \cdot (TimeBase + 1)}{58593{,}75} \cdot \frac{60}{2\pi} \qquad <3.4>$$

Donde **PPR** es el número de pulsos por revolución del encoder (200pulsos) y **TimeBase**, cuyo valor es 8, es el factor que multiplica al periodo de 256μs para obtener el intervalo de tiempo con el que la tarjeta de potencia realiza el muestreo de los encoders.

- **1/(1-z-1):** Función de transferencia del bloque que representa un muestreador ideal, necesario ya que la salida de nuestro sistema debe de ser discreta.

- **K_2:** Constante para obtener la velocidad de giro de los motores a partir de las cuentas de flancos realizadas en los encoders y sus marcas temporales.

$$K_2 = \frac{2\pi}{Np \cdot Tps} \qquad <3.5>$$

Donde como ya se vio en la ecuación <3.2>, **Np** representa el número de cuentas por cada revolución del encoder (800) y **Tps** es la constante de tiempo que nos permite pasar a segundos la diferencia entre las marcas temporales (90,15·10-6).

Las siguientes ecuaciones modelan cada uno de los bloques mencionados, y a partir de ellas se continúa el proceso de obtención de la función de transferencia de todo nuestro sistema del bajo nivel.

$$PID(z) = Kp + \frac{Ki}{1 - z^{-1}} \qquad <3.6>$$

$$G_{ZOH}(s) = \frac{1 - e^{sT}}{s} \qquad <3.7>$$

$$G(s) = \frac{A}{\tau s + 1} \qquad <3.8>$$

$$GT(z) = (1 - z^{-1}) \cdot Z\left\{\frac{G(s)}{s}\right\} = (1 - z^{-1}) \cdot Z\left\{\frac{A}{\tau s(z+1)}\right\} = A \cdot (1 - z^{-1}) \frac{(1 - e^{-(1/\tau)T}) z^{-1}}{(1 - z^{-1})(1 - e^{-(1/\tau)T} z^{-1})}$$

$$GT(z) = A \frac{(1 - e^{-T/\tau}) z^{-1}}{(1 - e^{-T/\tau} z^{-1})} \qquad <3.9>$$

Por tanto, la función de transferencia del sistema a estudio es la siguiente:

$$\frac{W}{Vcod} = \frac{PID(z)\cdot Kee\cdot GT(z)}{1+PID(z)\cdot Kee\cdot Kconv\cdot GT(z)}\cdot\frac{1}{1-z^{-1}}\cdot K2 =$$

$$= \frac{\left(Kp+\frac{Ki}{1-z^{-1}}\right)\cdot Kee\cdot A\cdot\frac{(1-e^{-T/\tau})z^{-1}}{(1-e^{-T/\tau}z^{-1})}}{1+\left(Kp+\frac{Ki}{1-z^{-1}}\right)\cdot Kconv\cdot Kee\cdot A\cdot\frac{(1-e^{-T/\tau})z^{-1}}{(1-e^{-T/\tau}z^{-1})}}\cdot\frac{1}{1-z^{-1}}\cdot K2 \quad <3.10>$$

Desarrollando dicha ecuación y organizando sus términos se obtiene finalmente la siguiente función de transferencia:

$$=\frac{K2(Kp+Ki)KeeA(1-e^{-T/\tau})z^{-1}-K2KpKeeA(1-e^{-T/\tau})z^{-2}}{1+[(Kp+Ki)KconvKeeA(1-e^{-T/\tau})-e^{-T/\tau}-2]z^{-1}+[1+2e^{-T/\tau}-(2Kp+Ki)KconvKeeA(1-e^{-T/\tau})]z^{-2}-[e^{-T/\tau}-KpKconvKeeA(1-e^{-T/\tau})]z^{-3}}$$

<3.11>

Empleando la ToolBox de Identificación del software Matlab, se procede a obtener la estructura resultante. Para ello se ejecutan los siguientes comandos tal y como se muestran a continuación.

- **`Th=arx(datai,[3 2 1])`**

Donde *datai* es el vector que contiene los valores de las muestras de entrada y salida obtenidas de la adquisición, *3* y *2* son los ordenes del denominador y del numerador respectivamente y *1* es el retardo que encontramos entre la entrada y la salida.

- **`Th=sett(Th,T)`**

Permite incluir el periodo de muestreo en el modelo.

- **`[numarx_id,denarx_id]=th2tf(Th)`**

Obtiene la estructura en formato numerador y denominador.

El resultado obtenido tras el procesamiento por Matlab para nuestro sistema esta modelado por la siguiente ecuación:

$$\frac{\overbrace{0.82727\,Z^{-1}}^{N1} - \overbrace{0.050397\,Z^{-2}}^{N2}}{1 - \underbrace{1.1593\,Z^{-1}}_{D1} + \underbrace{0.42958Z^{-2}}_{D2} - \underbrace{0.099384Z^{-3}}_{D3}} \qquad <3.12>$$

3.4.4. Obtención de los parámetros.

Para obtener el valor de los parámetros A y τ empleamos las ecuaciones de cada uno de los términos de la función de transferencia <3.11> y sus valores obtenidos por Matlab <3.12>. De esta forma, según que términos se empleen nos aparecen distintos sistemas de ecuaciones que producen diferentes resultados para los mismos parámetros.

A continuación se muestran algunos ejemplos:

```
%A) termino 1º del numerador(N1) y 1º del denominador(D1)
Taua=-T/log( (N1+Kconv/K2)-2-D1 )
Aa=N1/(K2*Kee*(Kp+Ki)*(1-exp(-T/Taua)))

%B) termino 1º del numerador(N1) y 2º del denominador(D2)
Taub=-T/log( D2/2+(2*Kp+Ki)*Kconv*N1/(2*K2*(Kp+Ki))-1/2 )
Ab=N1/(K2*Kee*(Kp+Ki)*(1-exp(-T/Taub)))

%C) termino 1º del numerador(N1) y 3º del denominador(D3)
Tauc=-T/log( D3+N1*Kp*Kconv/(K2*(Kp+Ki)) )
Ac=N1/(K2*Kee*(Kp+Ki)*(1-exp(-T/Tauc)))
```

Donde:

- Kee = 24/127
- Kconv = 200·(TimeBase+1)·60/(58593.75·2π)
- TimeBase = 8
- T=0.000256*TimeBase (seg.)

En la siguiente tabla se muestran los distintos resultados obtenidos para los términos de la ecuación <3.12> en la tarea de identificación para el conjunto de muestras estudiado, para pruebas con una carga de 56Kg, y con un periodo de muestreo de 100ms.

Elementos	τ	A
N1 y D1	3.0602e-004 +2.1371e-004i	0.0097 + 0.0000i
N1 y D2	2.2520e-004 +5.6159e-004i	0.0077 + 0.0000i
N1 y D3	**8.9126e-004**	**0.0109**
N2 y D1	3.5947e-005 +6.4991e-004i	5.6894e-004 +1.4720e-019i
N2 y D2	2.2459e-004 +5.6217e-004i	8.1486e-004 -5.8134e-020i
N2 y D3	**8.8750e-004**	**0.0012**

Tabla 3.2. Resultados de identificación para un periodo de 100ms.

A continuación se muestran los resultados obtenidos para los términos de la ecuación <3.12> para pruebas con una carga de 56Kg, y con un periodo de muestreo de 150ms.

Elementos	τ	A
N1 y D1	-0.0090	-0.0912
N1 y D2	2.6449e-004 +5.1645e-004i	0.0193 - 0.0000i
N1 y D3	**0.0011**	**0.0274**
N2 y D1	7.4793e-005 +6.4320e-004i	0.0162 - 0.0000i
N2 y D2	2.6480e-004 +5.1601e-004i	0.0229 - 0.0000i
N2 y D3	**0.0011**	**0.0324**

Tabla 3.3. Resultados de identificación para un periodo de 150ms.

En las tablas se ha marcado en negrita los resultados válidos de las distintas posibles soluciones de identificación para cada una de las 2 pruebas mostradas, tras desecharse valores irreales del sistema.

Tal y como se observa, los resultados válidos obtenidos son semejantes para las dos soluciones extraídas de los datos, en los dos experimentos mostrados.

Estos resultados permitirán realizar simulaciones realistas de los sistemas de control de posición diseñados para el seguimiento de la trayectoria que se presentan más adelante en esta memoria. En el proyecto, sin embargo, no se ha

llegado a incluir este estudio de diseño basado en ajuste por simulación de los controladores de posición diseñados, pero se deja este trabajo como línea abierta para trabajos futuros, con el conocimiento de las ventajas que puede reportar esta metodología de diseño.

4. Generación y Control de Trayectorias.

En este capítulo se explica el desglose de tareas que se realizan para conseguir que un objeto móvil, en este caso la silla de ruedas, sea capaz de circular de forma totalmente autónoma desde un punto origen de un edificio a otro punto destino.

4.1. Introducción.

Primeramente se ha de establecer una ruta, es decir, los lugares por los que ha de transitar la silla para llegar de la mejor forma al lugar deseado, quedando determinados las salas, pasillos o ascensores por los que se debe transitar.

A continuación se han de generar las trayectorias que se deben seguir para circular por cada una de las salas. Estas se deben ajustar a las particularidades de la zona del edificio donde se vayan a localizar, ya que al no tener todas las zonas o lugares las mismas características, el empleo de trayectorias constantes produciría movimientos no óptimos e incluso incómodos para el usuario.

Una vez determinadas las trayectorias, la silla ha de seguirlas lo más fielmente posible. Es tarea del controlador conseguir dicho objetivo. Para ello se empleará técnicas de "deadreckoning" en las que la estimación de la posición actual de un objeto se realiza mediante cálculos a partir de ciertas variables como la posición inicial, el rumbo seguido y la velocidad durante un periodo de tiempo

determinado, además del modelo cinemático de la silla. Conocida la posición es posible actuar de diferente forma sobre los motores de la silla y conseguir así alcanzar la posición deseada en cada instante de tiempo.

4.2. Obtención de ruta.

La primera tarea que se ha de desarrollar para la navegación de un móvil en cualquier entorno, es encontrar la ruta que se ha de realizar.

Como se comentó en la introducción de esta memoria, el presente TFC es continuación del TFM [01], en el que se propone un servicio de generación de rutas accesibles para edificios o entornos cerrados, que especifica y obtiene en caso necesario la descripción del edificio en términos de estructura, accesibilidad y conectividad entre los distintos orígenes y destinos posibles, e incluye algoritmos necesarios para la búsqueda de la ruta más adecuada en cada caso. Es por esto, que se usarán los resultados de este TFM como punto de partida de las tareas de navegación, quedando totalmente resuelta en él la primera de ellas: la de obtención de la ruta.

En el mencionado trabajo [01], se explican los procesos y algoritmos para la generación de las rutas deseadas, por lo que en este se realiza únicamente una pequeña mención a los procesos implementados en la generación de rutas:

1. Almacenar y codificar de manera off-line en ficheros XML la información relativa al edificio, su descripción geométrica así como la información referente a las posibles condiciones de movilidad.

2. Generar los Grafos de Descripción Métrica (MDG):
   2.1. Se generan los grafos que implican a las salas, ascensores y escaleras, a partir del centroide de la sala y el punto central de la puerta por la que se tiene acceso a ella.

   2.2. Se generan los grafos de los pasillos, compuestos por el conjunto menor de puntos que recorren el centro de estos.

2.3. Obtención de los puntos y segmentos de interconexión entre los dos tipos de grafos anteriores, de modo que se obtenga el MDG completo.

3. Generar las rutas.

A partir de los puntos de origen y destino, así como de las condiciones de movilidad deseadas elegidos por el usuario a través de la aplicación desarrollada, y cuya interfaz de usuario se muestra en la figura 4.1, se generan todas las posibles rutas que los unen en el MDG obtenido a priori para después poder establecer cual de ellas es la que forma el camino mínimo o más adecuado.

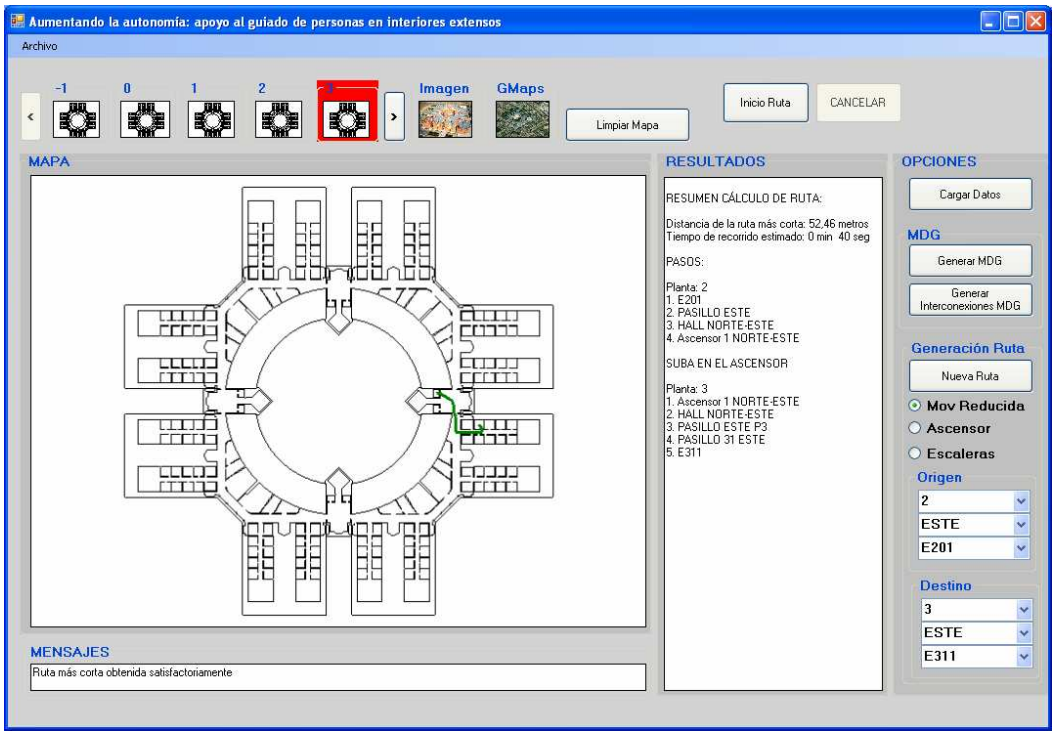


Figura 4.1. Interfaz de usuario de la aplicación desarrollada.

Tras realizar los tres procesos descritos se obtiene como resultado una sucesión de puntos en un plano, llamados nodos que forman la ruta que debe realizar la silla.

Este resultado es devuelto a través de un fichero XML llamado route.xml. En él, cada uno de los nodos que forman la ruta generada está representado por el elemento “point”, cuyos atributos “x” e “y” representan las respectivas coordenadas del plano. Cada conjunto de nodos pertenecientes a la misma planta

se encuentran incluidos dentro de una etiqueta “Floor” que a través del atributo “number” indica la planta a la que corresponde.

La figura 4.2 muestra un ejemplo de una ruta generada dentro de una misma planta del edificio de la Escuela Politécnia y su correspondiente fichero route.xml. Además, en azul se representa la situación del origen de coordenadas en el plano.

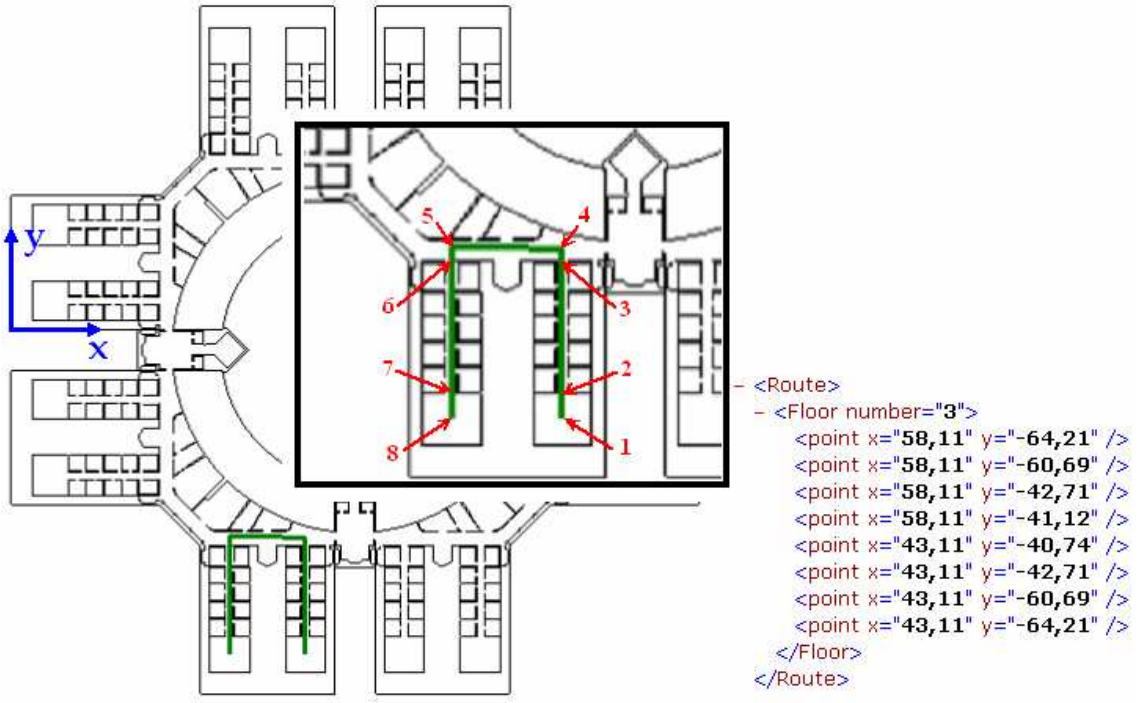


Figura 4.2. Creación de la ruta y su fichero route.xml.

En el ejemplo de la figura 4.2 se incluye también una vista ampliada de la ruta, donde se detalla la posición de cada uno de los nodos que genera el algoritmo de obtención de ruta y que corresponden a:

- Los centroides de las salas origen y destino (nodos 1 y 8).
- Los centros de las puertas por las que se ha de pasar (nodos 2, 3, 6 y 7).
- Los nodos pertenecientes a los pasillos por los que debe circular y que se encuentran en la perpendicular de las puertas (nodos 4 y 5).

En el ejemplo descrito se observa que no se han generado nodos a lo largo de los pasillos entre los pares de nodos 2-3 y 6-7, ya que coinciden con los puntos de la recta que une los puntos medios de las puertas existentes en los dos extremos de los pasillos.

Para los casos en los que la ruta generada implica un cambio de planta, el fichero route.xml que se obtiene es similar al mostrado en el ejemplo de la figura 4.3, donde el origen seleccionado es exactamente el mismo que el del anterior ejemplo, pero en este caso el destino es la misma sala de la planta inferior.

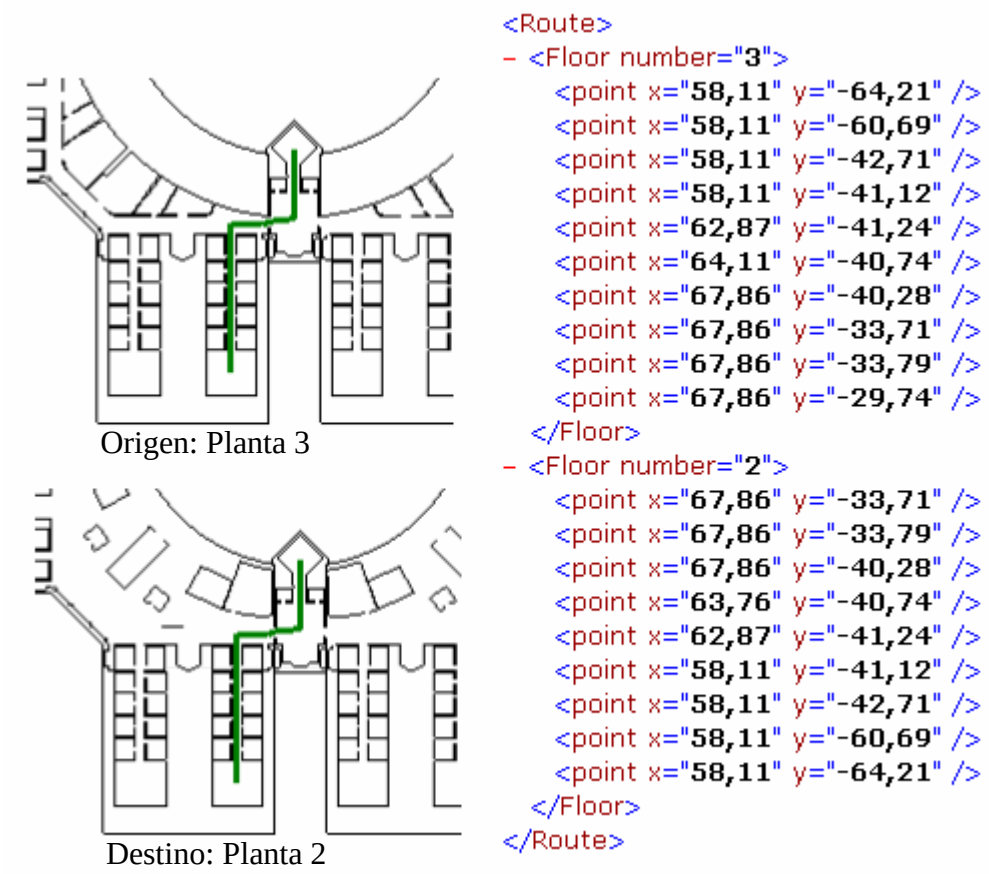


Figura 4.3. Creación de una ruta entre dos plantas y su fichero route.xml.

Como puede verse, los tramos de la ruta pertenecientes a cada una de las plantas, están descritos en el fichero route.xml en diferentes conjuntos de nodos, englobados por las etiquetas “Floor” que identifican la planta del edificio a la que corresponden.

4.3. Generación de trayectorias.

4.3.1. Descripción.

Una vez establecida la ruta de nodos optima a seguir para ir de un origen a un destino, la siguiente cuestión a afrontar es la generación de las trayectorias que se van a emplear para seguirla.

Como ya se comentó anteriormente, la ruta indica el camino que tiene que realizar la silla para llegar desde un nodo a otro, por tanto es necesario definir el tipo de movimientos que la van a formar.

En los siguientes apartados primeramente se realiza un análisis de distintas soluciones por las que se puede optar para la generación de las trayectorias, de modo que se pueda hacer uso de aquella que mejor se adapte a este trabajo.

Más adelante, se detallan los diferentes pasos a realizar para procesar los datos obtenidos del generador de rutas de manera que se obtengan de ellos las distintas trayectorias que componen una ruta.

4.3.2. Elección de trayectorias.

Es necesario, por tanto, en primer lugar elegir el tipo de trayectoria a emplear para recorrer cada uno de los tramos.

Como se verá a continuación, optar por cada uno de los tipos está condicionado por diferentes factores, y la elección de un tipo u otro marcará la posterior implementación del controlador que se debe emplear para seguirlas de manera autónoma.

En la actualidad son conocidos diferentes algoritmos para la generación de trayectorias que unen dos o más puntos. Algunos de los más empleados en la navegación robótica debido a sus buenos resultados son los generadores de las curvas clotoides y "splines".

La primera de ellas, mostrada en la figura 4.4 izquierda, tiene como principal característica la variación de la curvatura de giro en la trayectoria de forma constante y proporcional a la longitud del arco recorrido. Estas trayectorias son muy útiles para conseguir continuidad en los cambios de dirección. La mayor desventaja que tiene la obtención de dicha curva es el elevado tiempo de cálculo que requiere, siendo necesario emplear algoritmos que ayuden a reducirlo.

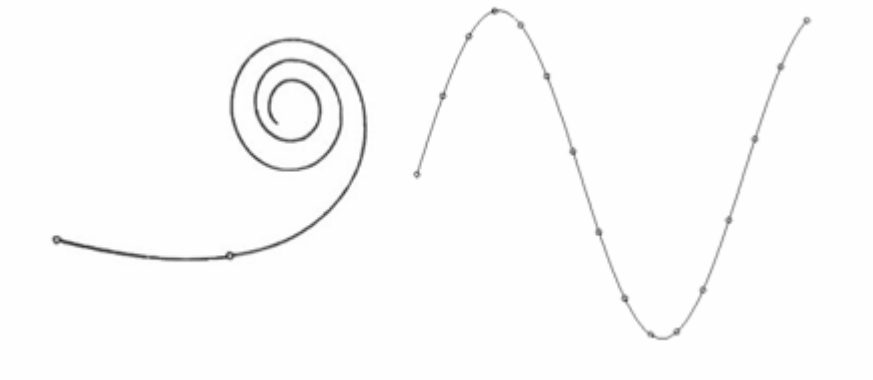


Figura 4.4. Curvas clotoides y splines.

Por otro lado, las curvas “splines”, también empleadas para tareas de interpolación de datos, permiten la unión de varios puntos mediante una curva más o menos suave asegurando su continuidad. Un ejemplo de esta se muestra en la figura 4.4 derecha. El algoritmo que la genera emplea polinomios de grado variable, para obtener la curva que contenga a todos los puntos de la curva a generar. Es común el empleo de distintos polinomios de tercer grado para obtener una sucesión de curvas que unan los nodos por parejas, haciendo coincidir en estos polinomios las pendientes de cada una de las curvas con la de la sucesiva, para así conseguir la continuidad de la trayectoria con el menor numero de oscilaciones posibles.

Otra de las principales ventajas que tiene el empleo de estas curvas es la facilidad que otorgan a la inclusión o modificación de los puntos que forman la ruta a recorrer, algo que podría darse en caso de aparecer obstáculos en medio del camino y que el móvil deba salvar.

El empleo de estos tipos de curvas está muy extendido en las tareas de guiado de móviles, pero para el presente trabajo se ha optado por el uso de trayectorias más simples, recurriendo a trayectorias en línea recta para recorrer los tramos rectos y curvas simples de radio constante cuando sea preciso realizar giros.

Los motivos de elegir este tipo de trayectorias detalladas anteriormente son:

- Los algoritmos necesarios para generar este tipo de trayectorias son mucho más sencillos que los necesarios para las clotoides y “splines”.

- Las características de los edificios hacen que los nodos de las rutas sean fáciles de recorrer mediante trayectorias rectas y arcos de circunferencia de radio constante.

- Desde el comienzo, el proyecto ha sido concebido con la idea de recorrer rutas inalterables por los objetos que puedan aparecer. De lo contrario seria preciso el empleo de elementos sensores en el móvil que puedan detectarlos.

- En función del grado del polinomio empleado para generar las “splines”, se producen oscilaciones más o menos amplias a la hora de recorrer tramos rectos. Esto hace que la circulación por pasillos pueda no resultar cómoda.

- Las curvas simples, al hacer tender a cero su radio, permiten al móvil hacer giros sobre su misma posición. Al tratarse de una silla de ruedas con dos ruedas motrices formando una estructura de tracción diferencial, resulta algo muy útil para el objetivo de la aplicación.

Por todo lo anterior se ha decidido que para la navegación por el tipo de entornos de interés en este TFC, emplear trayectorias rectas y curvas de radio constante queda adecuadamente justificado. De este modo, las trayectorias a seguir, obtenidas a partir de una ruta dada serán similares a las mostradas en el ejemplo de la figura 4.5, en donde se representa en rojo una ruta definida por sus nodos, y en azul las trayectorias que se han de emplear para recorrerla.

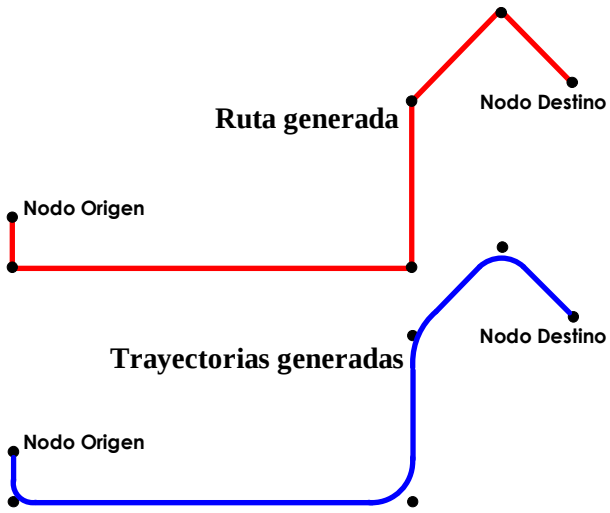


Figura 4.5. Ejemplo de trayectorias a emplear.

En la figura 4.6 se representa en color azul el recorrido realizado por la silla mediante el uso de las trayectorias elegidas y en rojo aparecen dos posibles trayectorias que se pueden dar si se emplean curvas “spline”. Como puede observarse, la trayectoria en azul resultará la más cómoda para el usuario.

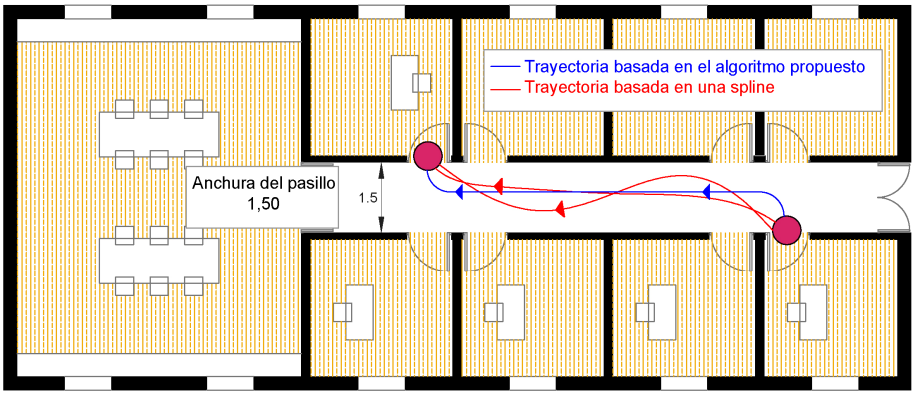


Figura 4.6. Distintas trayectorias a seguir para un mismo recorrido.

Existiría también la posibilidad de realizar únicamente desplazamientos en línea recta con la silla y rotaciones sobre si misma en cada uno de los nodos que impliquen un cambio de dirección, pero esta posibilidad queda totalmente desechada al tratarse de movimientos poco naturales, que han de producir constantes paradas y arranques, y por tanto notables deslizamientos de las ruedas que llevarían a cometer un importante error en la obtención de la posición deseada.

Ya en trabajos anteriores en esta línea [02], se optó por este tipo de desplazamientos debido sobre todo a la búsqueda de movimientos lo mas naturales posibles y semejantes a los que realizaría una persona. Aunque pueda parecer algo muy secundario, se debe recordar que se trata de facilitar el desplazamiento a una persona, y no de guiar una simple maquina como pudiera ser una carretilla elevadora en un almacén.

Además, este tipo de trayectorias hacen que los movimientos de la silla sean más fáciles de predecir por el resto de individuos presentes en el entorno, facilitando en gran medida la integración de la silla en el transito del resto de personas.

4.3.3. Pretratado.

Antes de proceder a generar las trayectorias a partir de la ruta definida por el correspondiente generador en el fichero route.xml, es necesario realizar un pretratado de la misma consistente en:

1. Eliminar los nodos muy próximos entre sí.
2. Introducir nodos extra, útiles para atravesar las puertas.
3. Corregir errores.

Dichas tareas se han resuelto mediante la creación de un algoritmo cuyo comportamiento se ha resumido en el diagrama de flujo de la figura 4.7. El resultado de aplicar dicho algoritmo es la creación de un nuevo fichero llamado nodospuertas.xml similar al fichero original route.xml, pero en el que los nodos no precisos desaparecen y se incluyen, en caso de ser necesarios, otros nuevos a los que se denominan como nodos de aproximación a puertas y que se emplearan para facilitar el cruce de la silla a través de ellas.

4.3.3.1. Eliminación de nodos muy próximos.

En algunas ocasiones puede darse el caso de que el generador de rutas proporcione una sucesión de nodos muy próximos (poco distantes) y que algunos de ellos puedan no ser necesarios, e incluso compliquen la tarea de seguimiento de la trayectoria. En estos casos, eliminar dichos nodos optimiza la generación de las trayectorias permitiendo que estas sean más naturales, y evitando curvas innecesarias o desaceleraciones no deseadas.

Para poder eliminar un nodo de la ruta éste ha de cumplir las siguientes condiciones:

- No debe ser ni el nodo primero ni último de la ruta, en cada una de las plantas del edificio por las que esta discurra, pues el primer nodo siempre ha de coincidir con la posición de la silla al inicio de la ruta, y el último determina el lugar al que se desea llegar.

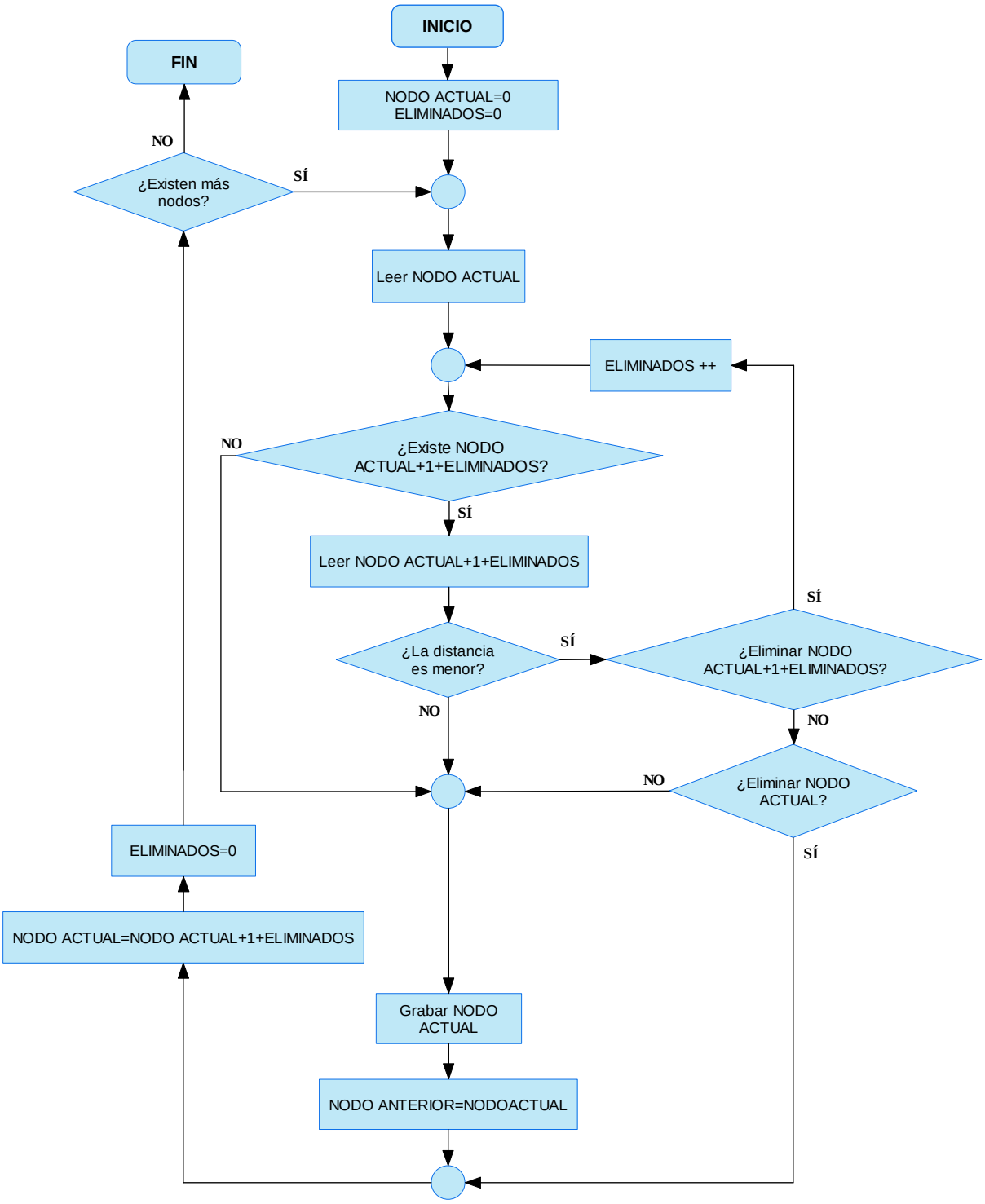


Figura 4.7. Diagrama del algoritmo del pretratado del fichero route.xml.

- Debe encontrarse dentro de una distancia máxima con respecto del resto de nodos de la ruta. Este parámetro se ha fijado a 0,7m pues es ligeramente inferior a la distancia mínima que se ha encontrado entre dos nodos indispensables en el entorno de prueba testeado en el proyecto. En caso de tratarse de nodos alrededor de los coincidentes a las puertas de unión entre pasillos y halls de ascensores, esta distancia se aumenta a 0,9m.

- No debe de ser el nodo de ninguna puerta o ascensor. Para evaluar esta última condición, se realiza una búsqueda del posible nodo a eliminar en el fichero Graphs.xml con el que cuenta la aplicación generadora de rutas, ya que en él se encuentra la información que determina la tipología de cada uno de los nodos.

4.3.3.2. Inserción de nodos de aproximación a puertas.

La tarea de cruzar una puerta con la silla de ruedas, puede requerir mayor precisión en el seguimiento de la trayectoria que la cruza que en el de cualquier otra trayectoria, y cumplir la condición de realizar el cruce con una trayectoria perpendicular a la orientación de la puerta para evitar la colisión con esta.

En la figura 4.8 aparece un ejemplo de lo comentado, donde una ruta pasa por una puerta de manera perpendicular a esta y a través de su punto medio (nodo N3). Por ello no es necesario realizar ninguna intervención sobre los nodos, ya que esta es la manera óptima de cruzarla.

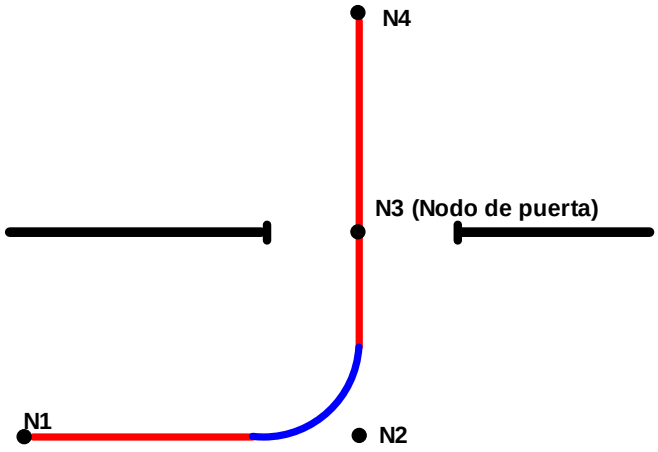


Figura 4.8. Trayectorias optimas a través de una puerta.

Pero puede ser que esto no suceda así. Como se muestra en este otro ejemplo de la figura 4.9, en este caso el nodo destino N4 no se encuentra situado en la perpendicular a la puerta que pasa por su nodo (N3). Como puede apreciarse en la imagen, al tener que realizar un giro, la trayectoria curva se desplaza del centro de la puerta alejándose más o menos de este según fuese la posición del nodo N4, y si la puerta no es lo suficientemente ancha la silla chocará con ella, en este caso con el lado derecho.

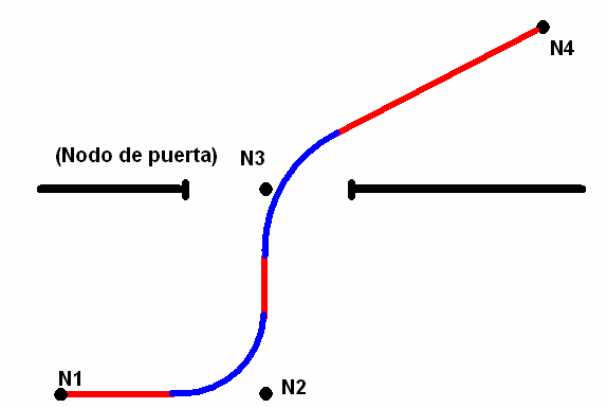


Figura 4.9. Trayectorias a través de puertas sin nodos de aproximación.

Para solventar este posible problema, se insertan en la ruta los denominados nodos de aproximación a puertas, tal como muestra la figura 4.10. Al añadir los dos nuevos nodos, la trayectoria que se genera asociada a estos nodos fuerza a atravesar la puerta de forma perpendicular a su orientación y pasando por el centro esta.

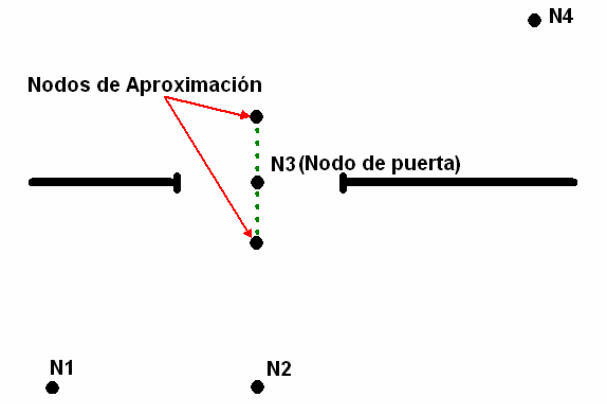


Figura 4.10. Inserción de los nodos de aproximación a puerta.

La figura 4.11 muestra las trayectorias que se generan tras incluir en el ejemplo anterior los nodos de aproximación a puertas. Como puede verse, ahora la ruta pasa por el centro de la puerta y totalmente perpendicular a su orientación en el momento de cruzarla, por lo que no se producirán colisiones con esta.

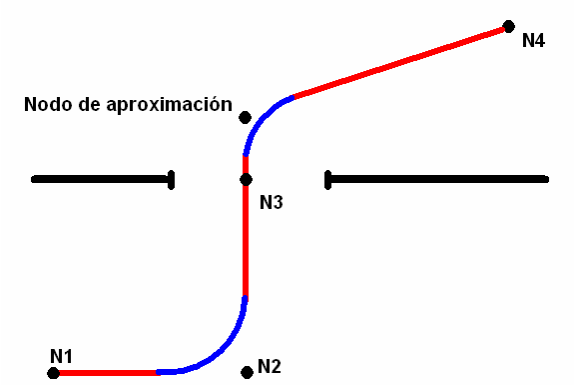


Figura 4.11. Trayectorias resultantes a través de puertas.

Como puede verse también en la figura 4.11, en este caso tan solo se introduce uno de los dos nodos de aproximación, ya que al encontrarse el nodo N2 en la perpendicular a la puerta que pasa por su centro, no es necesario hacer uso del nodo auxiliar entre los nodos N2 y N3, por lo que no es añadido a la ruta.

4.3.3.3. Corrección de errores.

Se detectó que las coordenadas del mapa a partir del que se generan las rutas están desplazadas medio metro hacia el centro del edificio en los nodos pertenecientes a los cuatro bloques de comunicaciones que unen cada una de las zonas del edificio.

Para resolver el problema se les añadió el error detectado a los nodos con error, de modo que los desplazamientos a través de estas zonas se realicen con normalidad.

Este tipo de errores de codificación de la posición de los nodos en el mapa deberían ser corregidos en el propio mapa. Sin embargo, suele ser habitual que no se detecten hasta que se ejecuta la generación de trayectorias y su seguimiento, por lo que será habitual usar el algoritmo de modificación de nodos en la exportación de esta aplicación a cualquier otro entorno.

4.3.4. Generación de trayectorias.

Como se comentaba en puntos anteriores, la elección del tipo de trayectorias realizada va a dividir la ruta en dos tipos de trayectorias: rectas y curvas simples de radio constante.

Para saber cuando se ha de emplear cada una de estas dos trayectorias y como, se realiza un análisis secuencial con todos los tramos que forman la ruta tal como se detalla en el diagrama de la figura 4.13. Los nodos son leídos del fichero nodospuertas.xml generado mediante los procesos descritos en el apartado anterior y los resultados serán grabados en un nuevo fichero XML llamado trayectorias.xml.

Si se toma como ejemplo la ruta de la figura 4.12, se comienza el análisis con los tres primeros nodos N1, N2, N3, comprobando si se encuentran alineados o no. Como en este caso lo están, el tramo N1-N2 se realiza mediante una trayectoria recta. Continuando el análisis, ahora se toman los nodos N2, N3, N4, y como ahora no se encuentran alineados, en el nodo N3 es necesario realizar un giro, por lo que el tramo N2-N3 se subdivide en un tramo recto y el comienzo de una curva y el tramo N3-N4 se subdivide también en un tramo que será el final de la curva y otro tramo recto.

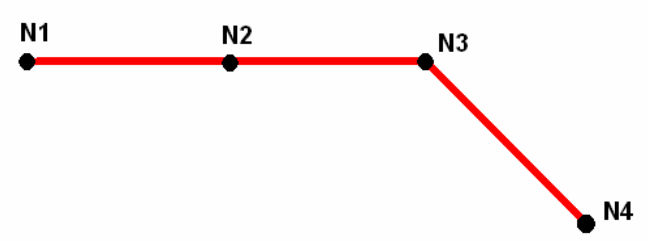


Figura 4.12. Ejemplo de ruta creada.

4.3.4.1. Obtención de las trayectorias rectas.

Las trayectorias rectas están definidas por los propios nodos que definen el tramo. Si continuamos con el ejemplo de la figura 4.12, las coordenadas de los nodos N1 y N2 son las que definen la recta que se ha de seguir para recorrer ese tramo.

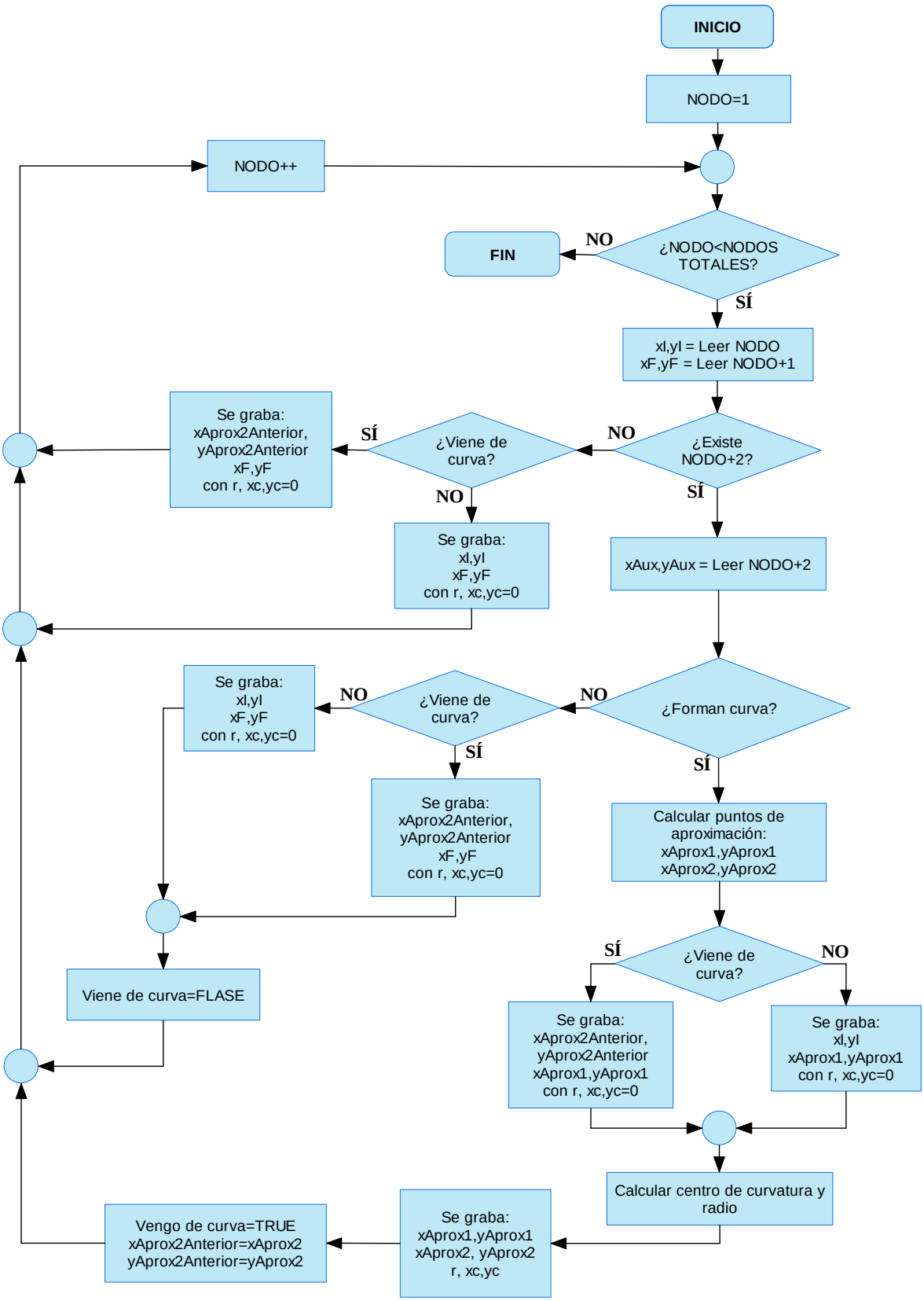


Figura 4.13. Diagrama para la generación de las trayectorias.

4.3.4.2. Obtención de las trayectorias curvas.

Este tipo de trayectorias están determinadas por el ángulo a girar que forman las dos rectas especificadas por los tres nodos en análisis y por lo que se ha denominado como distancia de aproximación a curva, que es la distancia que existirá entre los nodos de inicio y fin de la curva buscada y el nodo central, que para el ejemplo usado de la figura 4.12 es N3.

Una vez realizados los procesos de generación, la curva queda totalmente definida mediante las coordenadas del nodo inicial y final, su radio y las coordenadas de su centro de curvatura. En la figura 4.14 se muestra el desarrollo de los distintos procesos.

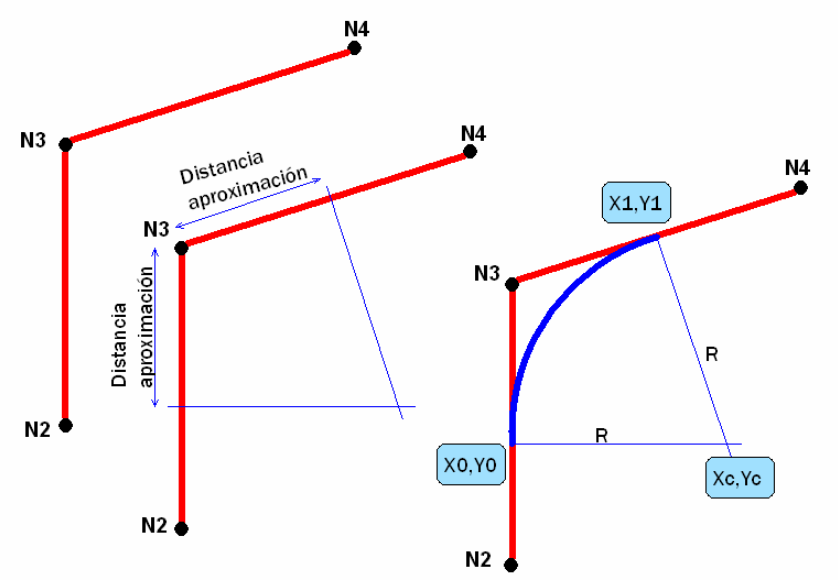


Figura 4.14. Creación de trayectorias curvas.

Para obtener estos datos, se comienza hallando los puntos de inicio y fin de la curva a generar, que serán los puntos que se encuentran en cada una de las dos rectas separados del nodo central la denominada distancia de aproximación a curva, siendo la misma para ambos tramos ya que se desea que el radio de la curva sea constante.

La ecuación <4.1> representa la ecuación de la circunferencia donde todos sus puntos distan del nodo central la distancia de aproximación, mientras que <4.2> es la ecuación de una recta que pasa por el nodo central y de pendiente **m**

calculada en <4.3> a partir del nodo central y el punto de aproximación para esa recta.

$$d_{Aprx}{}^2 = \left(x_{Nc} - x_{Aprx}\right)^2 + \left(y_{Nc} - y_{Aprx}\right)^2 \qquad <4.1>$$

$$y_{Aprx} = y_{Nc} + m\left(x_{Aprx} - x_{Nc}\right) \qquad <4.2>$$

$$m = \frac{y_{Nc} - y_{Aprx}}{x_{Nc} - x_{Aprx}} \qquad <4.3>$$

Resolviendo el sistema formado por las tres ecuaciones y tras resolver las posibles indeterminaciones que se pueden dar con algunas pendientes concretas, se obtiene el punto de la recta buscado. Si se aplica de igual manera para las dos rectas que forman los tramos, ya se dispone de los nodos de inicio (x0,y0) y fin (x1,y1) de la curva a obtener.

La distancia de aproximación será siempre la mitad de la longitud del menor de los dos tramos bajo análisis, pues de esta manera se garantiza que en caso de que el anterior o posterior tramo estos, formase una curva con algunos de los dos tramos, la curva se pueda realizar de la misma forma que la actual.

También se impone la condición de que la distancia de aproximación a curva no supere nunca un valor, que para este trabajo se ha fijado a 1m, ya que de lo contrario, si los dos tramos fuesen muy largos, la curva en su punto medio se alejaría demasiado del nodo central y podría darse algún problema con el entorno en el que se encontrase la silla.

Hallado el punto inicial y el final, se traza en ellos una perpendicular a cada uno de los tramos y el punto donde estas se crucen será el centro de circunferencia buscado.

El cálculo de la perpendicular al primer tramo en el punto inicial se realiza a través de las ecuaciones <4.4> y <4.5>. La primera representa la recta que une el punto inicial (x0,y0) y cualquier otro punto de la perpendicular que se busca, y la

segunda ecuación es la forma de obtener la pendiente de la perpendicular a partir de la pendiente **m** del tramo.

$$y - y_0 = mp(x - x_0) \qquad <4.4>$$

$$mp = \frac{-1}{m} \qquad <4.5>$$

Resolviendo de nuevo el sistema formado por las dos ecuaciones se obtiene <4.6>, donde al dar un valor cualquiera para **x** o **y**, se obtiene su correspondiente. De este modo, aplicándose a los dos tramos en los puntos de inicio y fin de la curva, se obtienen las ecuaciones punto a punto de las dos perpendiculares.

$$y = \frac{(x_0 - x_{Nc})}{(y_0 - y_{Nc})}(x_0 - x) + y_0 \qquad <4.6>$$

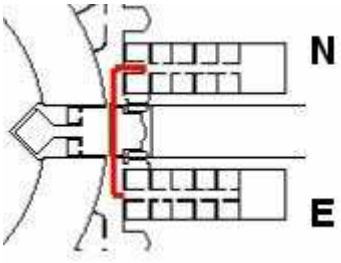
Ya solo queda hallar el punto de intersección entre las dos perpendiculares a los tramos calculadas y se habrá obtenido el centro de curvatura (xc,yc). Para ello tan solo se ha de resolver el sistema formado por las dos ecuaciones de las dos perpendiculares.

Por último, el radio de curvatura es fácil de obtener como la distancia desde el centro de curvatura a cualquiera de los puntos de inicio o fin de la curva definida.

4.3.5. Descripción de los datos resultantes.

En este apartado se muestra y detalla mediante un ejemplo la evolución que sufren los datos que se manejan en el proceso de generación de trayectorias.

La figura 4.15 presenta como ejemplo una ruta obtenida a través de la aplicación generadora de rutas. En dicha aplicación se ha seleccionado como origen de la ruta el comienzo del pasillo 1 de la tercera planta en la zona Este, y como punto de destino, la puerta de los baños del pasillo 4 contiguo al anterior y que se encuentra en la zona Norte.




```
- <Route>
  - <Floor number="3">
      <point x="106,46" y="-13,9" />
      <point x="104,87" y="-13,9" />
      <point x="104,99" y="-9,14" />
      <point x="105,05" y="-8,26" />
      <point x="104,49" y="-7,9" />
      <point x="104,49" y="-0,06" />
      <point x="105,03" y="0,84" />
      <point x="105,04" y="5,6" />
      <point x="106,5" y="5,6" />
      <point x="109,97" y="5,6" />
    </Floor>
  </Route>
```

Figura 4.15. Creación de ruta y su fichero route.xml.

Los datos devueltos por la aplicación mediante el fichero route.xml pertenecen a distintos nodos de la ruta marcada en el plano. Entre otros, estos puntos son el origen y destino, las puertas que se atraviesan, los puntos medios de los pasillos y la unión entre los distintos pasillos.

Tras el pretratado, el fichero nodospuertas.xml obtenido es el presentado en la figura 4.16, en el que se observa que:

- Los nodos cuarto (105,05/-8,26) y séptimo (105,03/0,84) han desaparecido.
- Aparece un nuevo nodo (104,99/0,44) como nodo de aproximación a puerta para salir del bloque de comunicaciones.
- Se produce un desplazamiento de medio metro en los nodos de las puertas que dan acceso a las zonas Este y Norte, debido a la corrección realizada.

```
- <Route>
  - <Floor number="3">
      <point x="106,46" y="-13,9" />
      <point x="104,87" y="-13,9" />
      <point x="104,99" y="-9,14" />
      <point x="104,99" y="-7,9" />
      <point x="104,99" y="-0,06" />
      <point x="104,99" y="0,44" />
      <point x="105,04" y="5,6" />
      <point x="106,5" y="5,6" />
      <point x="109,97" y="5,6" />
    </Floor>
  </Route>
```

Figura 4.16. Contenido del fichero nodospuertas.xml.

A continuación se presenta el fichero trayectorias.xml en la figura 4.17, obtenido tras el procesado para la generación de las distintas trayectorias necesarias para recorrer la ruta.

```
- <Route>
  - <Floor number="3">
      <tramo x1="106,46" y1="-13,9" x2="105,665" y2="-13,9" r="0" xc="0" yc="0" />
      <tramo x1="105,665" y1="-13,9" x2="104,89" y2="-13,10525" r="0,7752094" xc="105,665" yc="-13,12479" />
      <tramo x1="104,89" y1="-13,10525" x2="104,9744" y2="-9,759817" r="0" xc="0" yc="0" />
      <tramo x1="104,9744" y1="-9,759817" x2="104,99" y2="-8,52" r="49,1963" xc="55,7937" yc="-8,52" />
      <tramo x1="104,99" y1="-8,52" x2="104,99" y2="-7,9" r="0" xc="0" yc="0" />
      <tramo x1="104,99" y1="-7,9" x2="104,99" y2="-0,06" r="0" xc="0" yc="0" />
      <tramo x1="104,99" y1="-0,06" x2="104,99" y2="0,19" r="0" xc="0" yc="0" />
      <tramo x1="104,99" y1="0,19" x2="104,9924" y2="0,6903631" r="51,6367" xc="156,6267" yc="0,19" />
      <tramo x1="104,9924" y1="0,6903631" x2="105,0329" y2="4,870168" r="0" xc="0" yc="0" />
      <tramo x1="105,0329" y1="4,870168" x2="105,77" y2="5,6" r="0,7371104" xc="105,77" yc="4,863025" />
      <tramo x1="105,77" y1="5,6" x2="106,5" y2="5,6" r="0" xc="0" yc="0" />
      <tramo x1="106,5" y1="5,6" x2="109,97" y2="5,6" r="0" xc="0" yc="0" />
    </Floor>
  </Route>
```

Figura 4.17. Fichero trayectorias.xml.

Cada una de las entradas del fichero trayectorias.xml con la etiqueta “tramo” pertenece a cada uno de los tramos que conformaran el itinerario a seguir, y el significado de sus distintos atributos es el detallado en la tabla 4.1.

Atributo	**Significado**
x1/y1	Coordenadas del nodo origen del tramo.
x2/y2	Coordenadas del nodo destino del tramo.
r	Radio de la curva a seguir.
xc/yc	Coordenadas del centro de curvatura de la curva a seguir.

Tabla 4.1. Tabla descriptiva del contenido del fichero trayectorias.xml.

Se observa que para los tramos rectos, tanto el valor del radio como las coordenadas del centro de circunferencia son nulos mientras que en las trayectorias curvas si existen valores para dichos atributos. Esta será la característica a emplear para distinguir más adelante en las tareas de seguimiento entre un tipo de trayectoria y otro, ya que por la posición del origen de coordenadas en el plano, no es posible que se den trayectorias con radio y centro de curvatura ambos nulos.

Finalmente se aúnan todos los tramos rectos sucesivos, de manera que tras un tramo recto, el siguiente sea siempre una trayectoria curva. De este modo se consiguen velocidades más uniformes, ya que de lo contrario la silla

desaceleraría al aproximarse a un nodo intermedio entre dos rectas para después volver a acelerar para alcanzar el siguiente nodo. De esta forma se obtienen un último fichero llamado trayectorias2.xml el cual se muestra a en la figura 4.18.

```
- <Route>
  - <Floor number="3">
      <tramo x1="106,46" y1="-13,9" x2="105,665" y2="-13,9" r="0" xc="0" yc="0" />
      <tramo x1="105,665" y1="-13,9" x2="104,89" y2="-13,10525" r="0,7752094" xc="105,665" yc="-13,12479" />
      <tramo x1="104,89" y1="-13,10525" x2="104,9744" y2="-9,759817" r="0" xc="0" yc="0" />
      <tramo x1="104,9744" y1="-9,759817" x2="104,99" y2="-8,52" r="49,1963" xc="55,7937" yc="-8,52" />
      <tramo x1="104,99" y1="-8,52" x2="104,99" y2="0,19" r="0" xc="0" yc="0" />
      <tramo x1="104,99" y1="0,19" x2="104,9924" y2="0,6903631" r="51,6367" xc="156,6267" yc="0,19" />
      <tramo x1="104,9924" y1="0,6903631" x2="105,0329" y2="4,870168" r="0" xc="0" yc="0" />
      <tramo x1="105,0329" y1="4,870168" x2="105,77" y2="5,6" r="0,7371104" xc="105,77" yc="4,863025" />
      <tramo x1="105,77" y1="5,6" x2="109,97" y2="5,6" r="0" xc="0" yc="0" />
    </Floor>
  </Route>
```

Figura 4.18. Fichero trayectorias2.xml.

El motivo de dividir los resultados de cada uno de los procesos en distintos ficheros es que facilita en gran medida los algoritmos a desarrollar para la generación de las trayectorias finales a emplear. Además, el mantener los resultados de los distintos procesos resulta de gran ayuda para las tareas de depuración, pues la resolución de posibles problemas se vuelve mucho más sencilla gracias al análisis de estos ficheros.

4.4. Control de movimientos.

4.4.1. Descripción.

Una vez obtenidas todas las trayectorias mediante las que se desea recorrer la ruta establecida, la siguiente tarea es conseguir que la silla de ruedas las siga de manera autónoma y precisa, permitiéndose un error que esté dentro de unos márgenes comprensibles. Como ya se detallará de manera más concreta en el capítulo de resultados y conclusiones, errores de pocos centímetros carecen de importancia teniendo en cuenta los desplazamientos realizados.

Para llevar a cabo el presente proyecto se ha hecho uso de un sistema de posicionamiento relativo, ya que se emplean técnicas de “deadreckoning” para calcular la posición en la que se encuentra la silla, gracias a los encoderes instalados en los ejes de sus motores. Esto quiere decir que la posición real de nuestro móvil en cada instante, será estimada a partir de la posición de origen y los movimientos que realice en el transcurso del tiempo.

4.4.2. Modelo del lazo de control.

Debido a la arquitectura completa de la plataforma hardware empleada, a la hora de realizar el control de la silla se distinguen dos sistemas. Uno de ellos es el bajo nivel, formado por los motores, los encoders y la tarjeta de potencia AX3500, la cual realiza las labores de lectura de encoders y excitación y control de los motores de manera que la respuesta de estos sea lo más fiel posible a las consignas que reciben, es decir, su velocidades angulares. Este se comunica a través del Bus-CAN con el otro sistema, el alto nivel, donde se desarrollan los procesos para realizar el control de posición y generar las consignas para cada motor (ω_D, ω_I) que se han de enviar al bajo nivel.

En la figura 4.19 se representa el lazo de control implementado para realizar las tareas de seguimiento, donde se pone de manifiesto lo comentado en el párrafo anterior y donde se muestra cada una de las tareas que intervienen en las labores de control. Centrándonos en el control de posición, pues el bajo nivel ya fue descrito con detallen en el tercer capítulo, donde se realizó su identificación, a continuación se explica el flujo del lazo de control y en los sucesivos apartados se detallará cada uno de los bloques que lo forman.

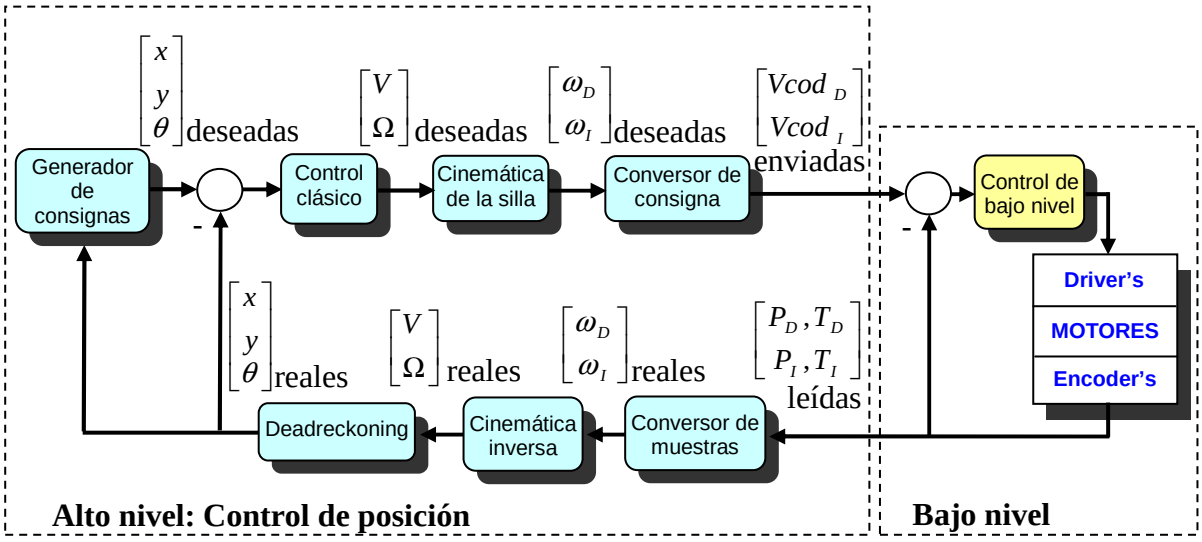


Figura 4.19. Diagrama del lazo de control implementado.

Es necesario remarcar que dado que estamos trabajando con un sistema digital programado, como sistema discreto que es, los instantes de tiempo también serán discretos y estarán dados por un determinado periodo de muestreo, el cual se explicará más adelante.

El análisis y descripción del lazo de control, según el orden en que se realizan las principales tareas de control es el siguiente:

a) *La posición real (xr,yr,θr).* Para poder obtener el error de posición a partir del cual se implementa todo el lazo de control, es necesario contar con la posición real y la consigna del móvil en cada período. La posición real es el primer parámetro que necesita el controlador para continuar con sus procesos y es obtenida mediante un sistema de posicionamiento relativo, haciendo uso de técnicas de "deadreckoning" a partir de los datos generados por los encoders ópticos acoplados a los motores y del modelo cinemático de la silla.

b) *La consigna (xd,yd,θd).* A partir de los parámetros característicos de las trayectorias a seguir, proporcionados por el generador de trayectorias, el controlador debe extraer cuál es la consigna de posición adecuada para cada instante de muestreo. Tal y como se observa en la figura 4.19, esta consigna depende de la posición real del móvil, de modo que según se mueva este, se irán generando las consignas de posición deseadas para cada instante, como es habitual en todos los sistemas de navegación.

c) *El error y el controlador.* Comparando las variables anteriores se obtiene el error de posición en los términos adecuados, es decir error de distancia y error de orientación (ed,eθ), para aplicar después el algoritmo de control que permita obtener las consignas de velocidad para las ruedas del móvil para el bajo nivel. Como se verá más adelante, se puede observar que se han implementado dos controladores de muy distinto tipo, uno para la velocidad de avance de la silla (V) y otro para la velocidad de giro de esta (Ω), por lo que estas son las dos consignas que el controlador va a generar como salidas en cada periodo de muestreo y que serán explicadas en apartados posteriores.

d) *Cinemática de la silla.* Tras generar el controlador las consignas de velocidades de la silla (V,Ω), se necesita obtener las consignas que se van a enviar al bajo nivel, las cuales son las velocidades de cada una de las ruedas (ω_D,ω_I). Estas, van a ser obtenidas haciendo uso de las distintas ecuaciones que detallan la cinemática de nuestra silla de ruedas, y que describe los movimientos que realiza la silla a partir de los distintos giros que se dan en cada una de sus ruedas. De similar manera, se puede emplear la cinemática inversa para que, a partir de las velocidades reales de las ruedas obtenidas gracias a los encoders, se obtenga las velocidades reales de avance y giro de la silla (V y Ω reales).

e) *Deadreckoning.* Es el último proceso que cierra el lazo de control, mediante el cual se va a calcular la posición real alcanzada a partir de las velocidades V y Ω reales durante el último periodo de muestreo y la que fue la última posición calculada en el intervalo anterior.

El siguiente diagrama de la figura 4.20 representa la implementación del lazo de control de posición desarrollado. Cada iteración comienza por el desbordamiento de un timer, el cual genera el periodo de muestreo. Su activación, así como la inicialización de las variables necesarias para el seguimiento de todas las trayectorias generadas, se produce en el momento de solicitarse el inicio del seguimiento del recorrido, tal como se muestra en la figura 4.21, donde también aparecen los procesos que se dan ante una orden de pausar el seguimiento de una trayectoria o de continuar con él.

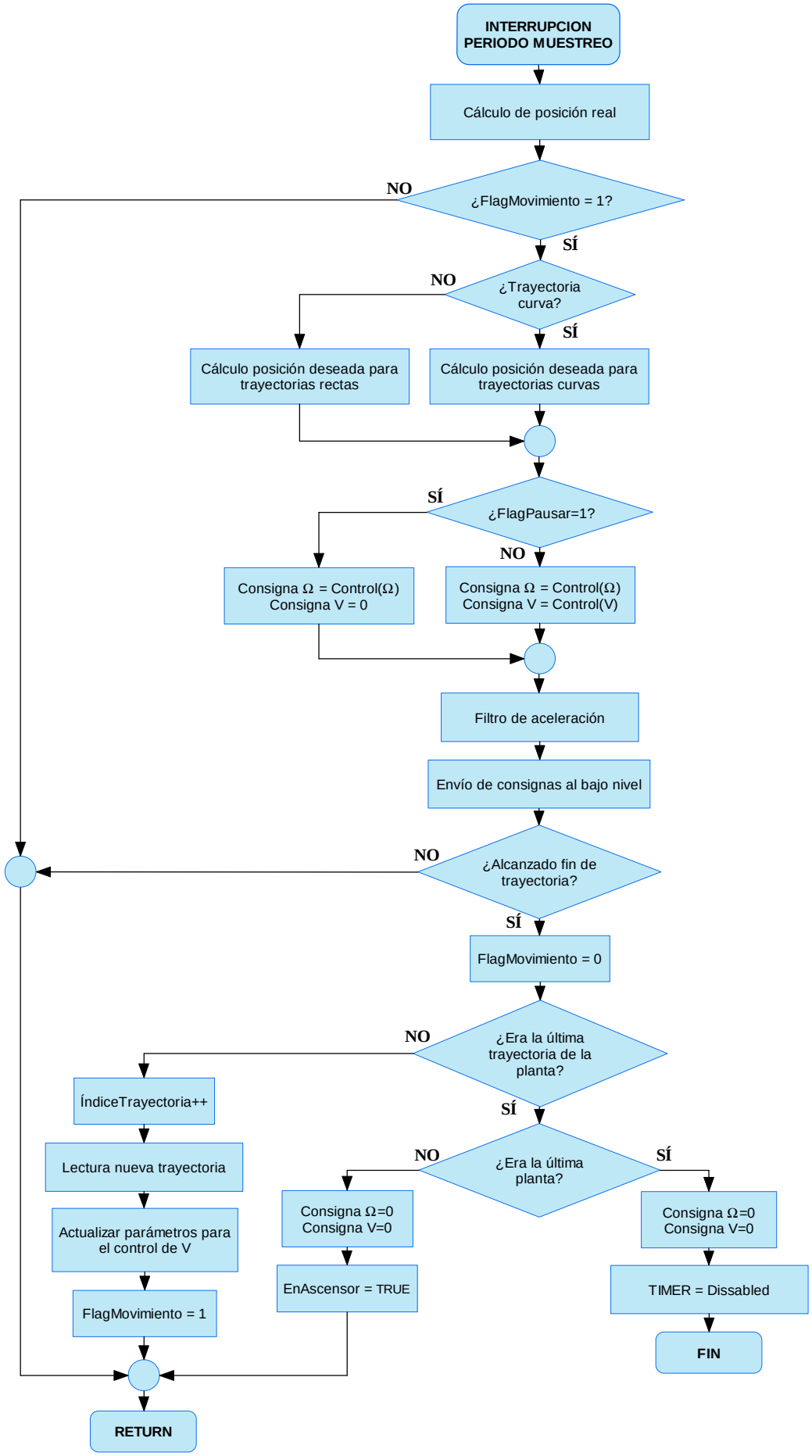


Figura 4.20. Tareas del lazo de control.

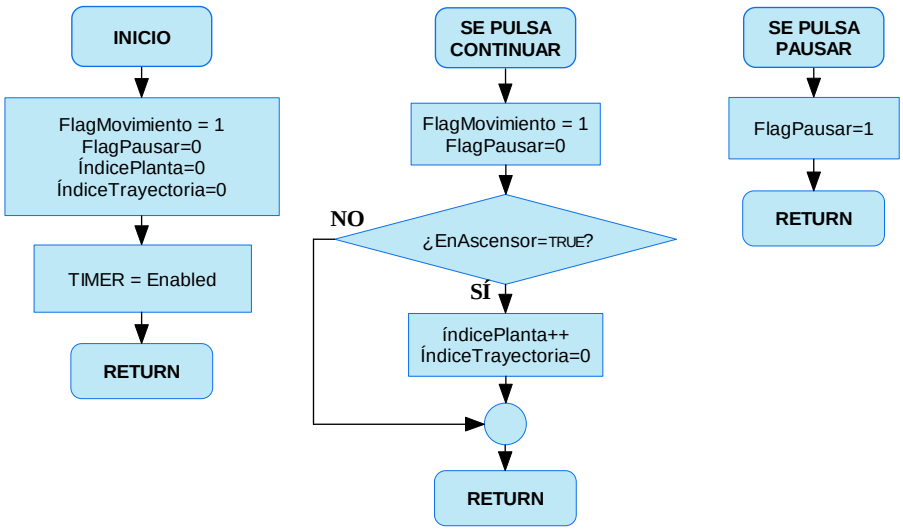


Figura 4.21. Tareas en la activación y pausa del seguimiento de trayectorias.

4.4.3. Elección del periodo de muestreo.

La elección de este periodo de muestreo es muy importante, pues para que el modelo discreto responda del mismo modo que el continuo, este debe ser al menos el doble de rápido que la respuesta de la planta. Asignarle el valor apropiado está condicionado por distintos aspectos además de la propia dinámica del sistema, la cual necesita de un periodo lo suficiente pequeño como para reconstruir el comportamiento del sistema igual que el modelo continuo.

- Es totalmente necesario que durante el intervalo de tiempo que dura el periodo de muestreo, nuestro sistema sea capaz de ejecutar todos los procesos necesarios para las tareas de control.
- No se puede superar la velocidad del bajo nivel, pues de lo contrario la actuación sobre los motores no podrá responder a las consignas del controlador.

Por un lado, en trabajos ya realizados anteriormente con la silla [06], se realiza un estudio sobre el límite de dicho tiempo, observando que hasta los 200ms, las diferencias entre las respuestas del modelo continuo y discreto son inapreciables, por lo que este puede ser el tiempo máximo a emplear como periodo.

Por otro, tenemos impuesta como condición la velocidad de comunicaciones con el bajo nivel. Dado que el intercambio de mensajes con datos procedentes de los

encoders entre la placa de potencia y el PC a través del Bus-CAN se realiza de forma periódica, fijada en 100ms, este será el periodo mínimo que se pueda usar.

Tomando como márgenes de tiempo los valores comentados, se fija el periodo de muestreo en el mínimo de ellos, 100ms, intervalo de tiempo en el que se puede realizar todo el procesado implementado y obtener un sistema más preciso que con periodos superiores.

4.4.4. Modelo de la silla de ruedas.

Para poder implementar un cierto control del movimiento del móvil sobre el que se realiza el sistema de navegación, va a ser necesario tener un modelo matemático del mismo.

Este modelo va a ser necesario no solo para analizar el control de posición que se diseña sino también para poder obtener la posición real del móvil si se desean emplear técnicas de "deadreckoning".

En varios de los trabajos analizados previamente [02] se presenta el modelo de la silla de ruedas con las entradas y salidas que se aprecian en la figura 4.22, donde las entradas del modelo son la velocidad de avance (V) y la velocidad de giro (Ω) de la silla.

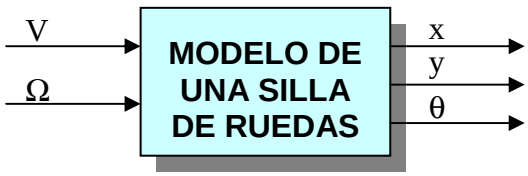


Figura 4.22. Entradas y salidas del modelo de la silla.

Teniendo en cuenta que la dirección de avance de la silla de ruedas es siempre la del eje longitudinal de la silla, y que el centro de giro de cualquier móvil se encuentra en la intersección de las perpendiculares a todas sus ruedas, se conoce como velocidad de avance de la silla de ruedas (V) a la velocidad con que se mueve el punto de intersección de estos dos ejes, tal y como se aprecia en la figura 4.23.

La velocidad de giro de la silla (Ω), indica por otro lado la variación de la orientación de la silla en un incremento de tiempo.

Las variables de salida del modelo elegidas informan de la posición de la silla en respuesta a estas consignas de velocidad. Esta posición vendrá definida por tres variables:

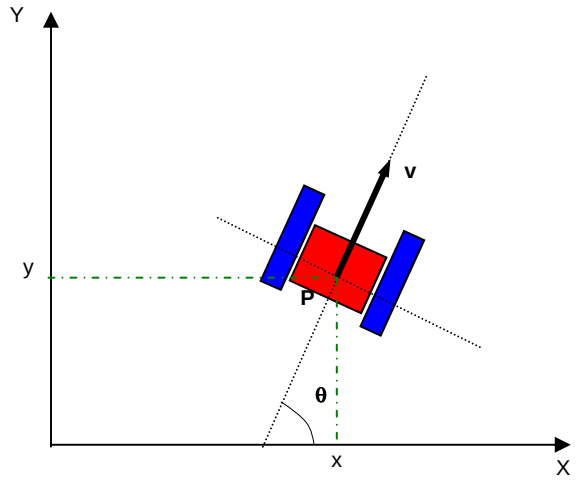


Figura 4.23. Representación de las variables de entrada y salida del modelo.

a) Por un lado el par (x,y) informa de las coordenadas cartesianas del centro de movimiento de la silla, en un sistema de coordenadas definido en el espacio en el que se desenvuelve la silla.

b) Por otro lado, la orientación (θ) informa del ángulo formado por el eje de movimiento de la silla (eje longitudinal) y el semieje positivo de abscisas del sistema de coordenadas elegido, en sentido antihorario.

A partir de la definición de las entradas y salidas, es inmediato obtener las ecuaciones del modelo de la silla, que serán las siguientes:

$$\dot{\theta} = \Omega \quad \text{[rad/s]} \quad \text{<4.7>}$$

$$\dot{x} = V[n] \cdot \cos\theta \quad \text{[m/s]} \quad \text{<4.8>}$$

$$\dot{y} = V[n] \cdot sen\,\theta \quad \text{[m/s]} \quad \text{<4.9>}$$

Por otro lado, si bien las consignas obtenidas por el controlador de posición en la mayor parte de los sistema de este tipo son V y Ω, los controladores de los motores de la silla de ruedas requieren como consignas, las velocidades angulares de cada una de las dos ruedas motrices (ω_D para la rueda derecha y ω_I para la rueda izquierda).

Las ecuaciones que describen la relación de las consignas con estas nuevas variables son lo que se conoce como cinemática de la silla, y pueden ser obtenidas fácilmente a través de los parámetros de la silla, tal como se muestra en la figura 4.24. Son las ecuaciones <4.10> y <4.11>.

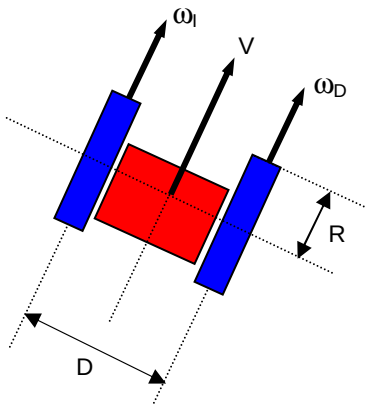


Figura 4.24. Diagrama de las relaciones cinemáticas de la silla de ruedas.

$$\omega_D = \frac{1}{R}\cdot\left(V + \Omega\frac{D}{2}\right) \qquad [rad/s] \qquad <4.10>$$

$$\omega_I = \frac{1}{R}\cdot\left(V - \Omega\frac{D}{2}\right) \qquad [rad/s] \qquad <4.11>$$

Las relaciones inversas (V y Ω en función de ω_D y ω_I) constituyen lo que se denomina como cinemática inversa.

A la hora de trabajar con el móvil, el sistema que realmente se va a presentar es el que se muestra en la figura 4.25. Al emplear un sistema de realimentación de posición en base a encoders ópticos incrementales, las salidas que proporciona la silla de ruedas son las velocidades angulares de las ruedas (ω_D y ω_I), a partir de las cuales el sistema de "deadreckoning" ha de obtener las variables de posición (x, y, θ). Para ello se basa en integrar la posición recorrida por la silla en cada periodo de ejecución del algoritmo de control.

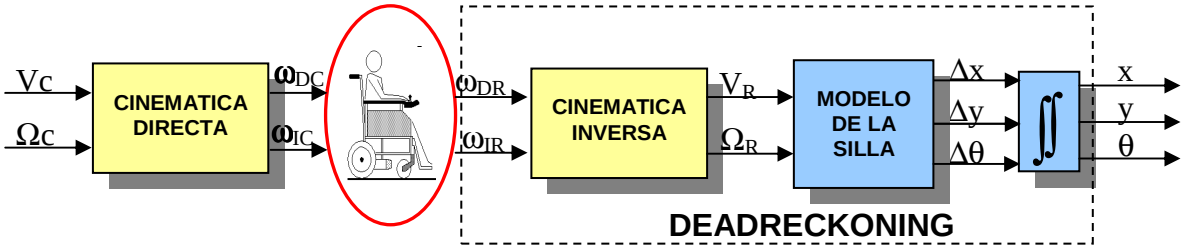


Figura 4.25. Diagrama explicativo del sistema real de realimentación.
Leyenda de subíndices:
C: Consigna, R: Valor real.

En el proceso de "deadreckoning" se puede obtener la ubicación real de la silla de ruedas en cada instante de muestreo conociendo la posición anterior de la misma (x[n-1], y[n-1], θ[n-1]) y mediante el modelo discretizado de esta, tal como se muestra a continuación (<4.12>, <4.13>, <4.14>).

$$\theta[n] = \theta[n-1] + \Omega[n] \cdot Ts \qquad [rad] \qquad <4.12>$$

$$x[n] = x[n-1] + V[n] \cdot \cos(\theta[n]) \cdot Ts \qquad [m] \qquad <4.13>$$

$$y[n] = y[n-1] + V[n] \cdot sen(\theta[n]) \cdot Ts \qquad [m] \qquad <4.14>$$

En estas ecuaciones queda de manifiesto el importante papel que juega el periodo de muestreo elegido. Como ya se explicó en el apartado anterior 4.4.3, su elección deber garantizar que el comportamiento de nuestro sistema discreto sea similar al que tendría el modelo de un sistema continuo, de modo que las diferencias entre sus respuestas sean inapreciables.

Además queda en este punto de manifiesto el problema que presenta el uso de un posicionador relativo como el de "deadreckoning". Debido a su constitución, la silla de ruedas se comporta como un sistema no holonómico, pues el estudio independiente de cada una de las partes móviles no basta para obtener el movimiento absoluto del sistema. Es por ello que para conocer la posición real del móvil es necesario integrar el movimiento realizado durante el último periodo con el acumulado durante todo el trayecto, con lo que los errores de posicionamiento cometidos en el proceso de integración pueden introducir errores de posición graves, pues son acumulativos.

Debido a esto en muchos casos se añaden estimadores que, a partir de la observación del movimiento de las ruedas, permitan obtener la posición real del sistema sin errores. En este proyecto tampoco se ha incluido un elemento de este tipo, por lo que en el controlador diseñado se tendrá especial cuidado en evitar aceleraciones bruscas, que son las situaciones más propensas para que se cometa un error de integración en el proceso de "deadreckoning".

4.4.5. Generador de consignas.

Como ya se comentó antes, el proceso de control del seguimiento de las trayectorias está basado en la corrección del error existente entre la posición que se desea tener en cada momento y la posición real en la que se encuentra la silla, por lo que se deben de obtener ambos parámetros en cada periodo de muestreo.

El generador de consignas es el encargado de obtener las primeras de dichas posiciones. Según se detalla en el modelado de la silla, la posición de esta en todo momento está definida por tres parámetros que son las dos coordenadas cartesianas que determinan el punto del plano en el que se encuentra (x, y) y su orientación (θ) respecto al eje de abscisas, por tanto este elemento del lazo de control ha de generar cada una de ellas.

La posición deseada corresponderá con algún punto de la trayectoria que se está siguiendo, en concreto con aquel que se halle más próximo a la posición real en la que se encuentre el móvil, y su orientación será aquella que permita el sentido de avance establecido por la trayectoria. Por tanto, las consignas [x, y, θ] deseadas dependerán de los resultados obtenidos por el generador de trayectorias y de la posición real que se tenga a la hora de realizar su obtención.

La manera de obtener cada uno de los parámetros dependerá por tanto del tipo de trayectoria que se desea seguir.

a) Generación de consignas para trayectorias rectas:

   Cuando se trate de una trayectoria recta, las coordenadas (x, y) deseadas serán obtenidas trazando una perpendicular a la recta que define la trayectoria que se quiere seguir, de modo que esta pase por el punto que define la posición real de la silla. El punto donde se cortan la perpendicular trazada y la recta de la trayectoria, definirá las coordenadas de la posición deseada. El parámetro que define la orientación (θ) será igual a la orientación de la propia trayectoria, pues al tratarse de una recta, esta es constante. La figura 4.26 representa lo expuesto.

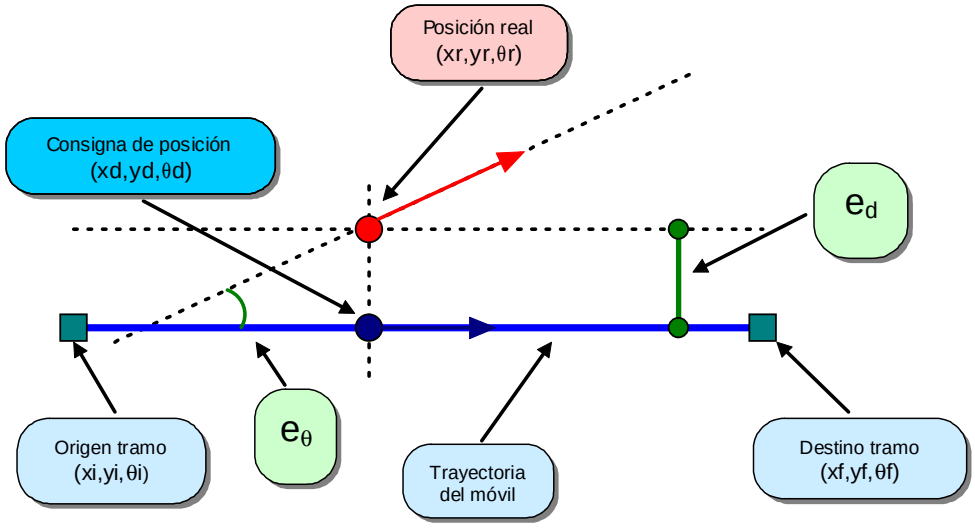


Figura 4.26. Cálculo de la posición deseada en trayectorias rectas.

Las ecuaciones <4.15><4.16><4.17> definen matemáticamente lo expuesto en el párrafo anterior, donde **m** representa la pendiente de la trayectoria y **mp** la de su perpendicular.

$$mp = -1/m \qquad \text{<4.15>}$$

$$x_d = \frac{y_i - m \cdot x_i - y_r + mp \cdot x_r}{mp - m} \qquad \text{<4.16>}$$

$$y_d = m \cdot (x_d - x_i) + y_i \qquad \text{<4.17>}$$

b) Generación de consignas para trayectorias curvas:

Con las trayectorias curvas se actuará de manera parecida, tal como se muestra en la figura 4.27. Para encontrar el punto de la trayectoria que corresponda con la posición deseada, se trazara una recta que una el centro de circunferencia de la trayectoria curva y que pase por el punto de la posición real en la que se encuentra la silla, el lugar donde la recta y la trayectoria se corten será el punto buscado. En este caso la consigna de orientación vendrá dada por la normal a la recta trazada, por lo que siempre será tangente a la trayectoria e cada uno de sus puntos.

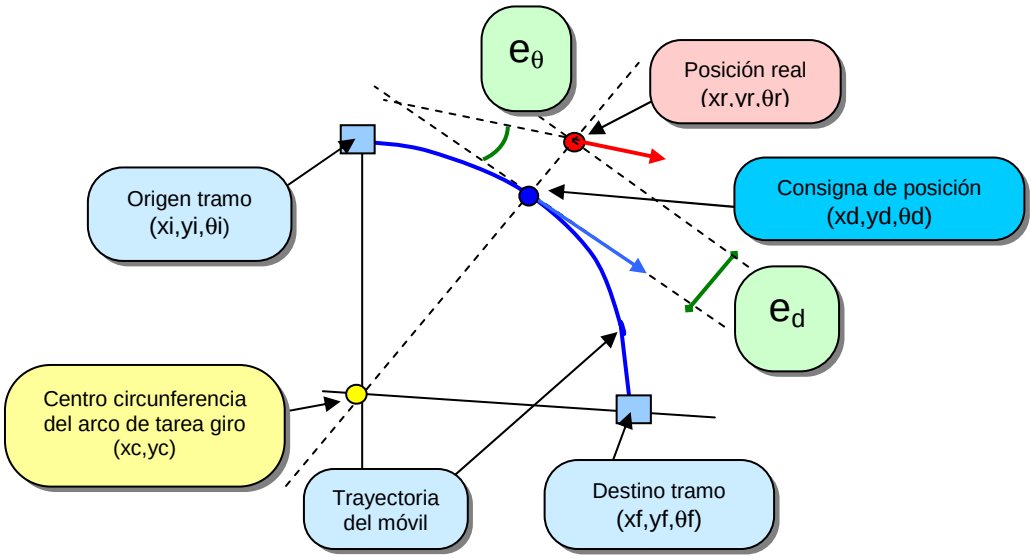


Figura 4.27. Cálculo de la posición deseada en trayectorias curvas.

De igual forma que antes, las ecuaciones <4.18><4.19><4.20> mostradas, detallan la manera de obtener matemáticamente los tres parámetros que determinan la posición deseada cuando el seguimiento pertenece a una trayectoria curva.

$$y_d = y_c + \frac{R}{\sqrt{\left(\frac{x_r - x_c}{y_r - y_c}\right)^2 + 1}} \qquad <4.18>$$

$$x_d = x_c + \frac{x_r - x_c}{y_r - y_c}(y_d - y_c) \qquad <4.19>$$

$$\theta_d = \arctan\left(\frac{yc - yr}{xc - xr}\right) + \frac{\pi}{2} \qquad <4.20>$$

4.4.6. Control.

Este es el bloque del lazo encargado de actuar para que, a partir de las entradas que recibe, genere las respuestas para la silla en forma de velocidad de avance (V) y velocidad de giro (Ω) y poder así realizar el seguimiento de cada una de las trayectorias que forman la ruta.

El controlador que se ha implementado toma en cada periodo de muestreo las consignas de posición [x, y, θ] deseadas, cuya obtención era explicada en el apartado anterior, y las compara con la posición real de la silla definida también con los mismos parámetros, coordenadas y orientación, de los cuales se detallará más adelante su manera de obtenerlos según se sucedan los procesos del lazo de control creado.

Al comparar las posiciones reales y deseadas, se obtiene como resultado dos parámetros que son la diferencia o error de desplazamiento (ed) y el error de orientación (eθ), que como se mostraba en la figura 4.19 serán los empleados por el controlador para generar las consignas V y Ω deseadas. En las figuras 4.26 y 4.27 empleadas para describir la obtención de la posición deseada, pueden verse también la definición gráfica de ambos errores para los dos tipos de trayectorias empleadas, mientras que las siguientes ecuaciones <4.21> y <4.22> muestran la manera matemática de obtenerlos.

Error de distancia: $$e_d = \sqrt{(x_r - x_d)^2 + (y_r - y_d)^2}$$ <4.21>

Error de orientación: $$e_\theta = \theta_d - \theta_r$$ <4.22>

El primero de ellos, el error de desplazamiento, se define como la distancia que separa la posición real de la silla y la posición deseada.

El otro, el error de orientación, se define como la diferencia entre la orientación real de la silla y la orientación de la trayectoria en la posición deseada.

Una vez definidas las entradas del controlador, a continuación se detalla la acción de este para obtener en cada instante de muestreo sus salidas V y Ω. Para facilitar tanto el diseño como el posterior ajuste del controlador a desarrollar, se opta por crear dos controladores independientes para cada una de las salidas a obtener, de manera que resulten más sencillas todas las tareas. La opción de generar ambas salidas mediante un solo controlador resulta más compleja, y para los resultados buscados en el presente trabajo no aportaría ningún beneficio significativo. Ya en trabajos anteriormente realizados con un móvil similar se optó por esta opción, [02], del cual se han extraído los fundamentos para ser aplicados en la generación de los actuales algoritmos de control.

4.4.6.1. Control de la velocidad de avance (V).

La variable de velocidad de avance (V) es la que permite a la silla avanzar por la trayectoria diseñada por el generador de trayectorias. El controlador de V ha de encargarse de que ésta sea lo más adecuada posible a las características del tramo a recorrer, haciendo que la conducción sea confortable para el usuario.

Por su sencillez y por el hecho de existir tan solo dos tipos de trayectorias, se ha optado por realizar un control de la velocidad de avance basado en la generación de un perfil trapezoidal para las trayectorias rectas y mantener una velocidad constante acorde con el radio en las trayectorias curvas. La figura 4.28 muestra lo comentado.

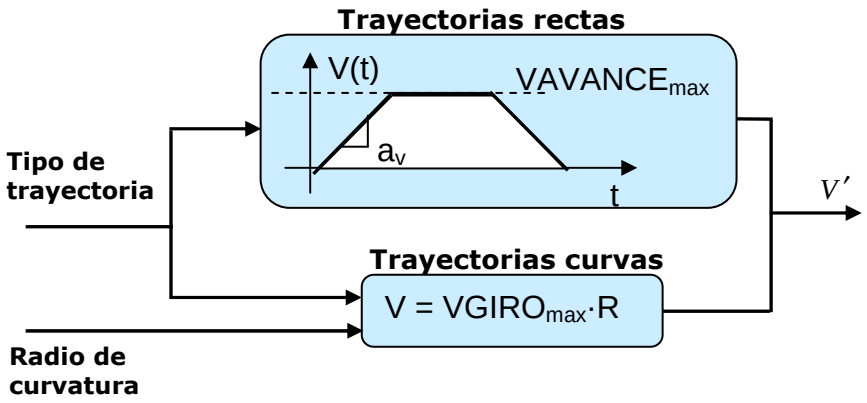


Figura 4.28. Diagrama del control de la velocidad de avance.

Por tanto, la forma de obtener la velocidad de avance dependerá del tipo de trayectoria a seguir:

a) Para las trayectorias rectas se genera como se comentaba un perfil de velocidad trapezoidal, el cual comienza con un incremento de velocidad hasta alcanzarse una velocidad máxima, manteniéndose ese valor durante el trayecto para finalmente decrementar hasta llegar a un valor final.

- El valor inicial variará dependiendo si se parte o no del reposo, igual que el valor final, que será nulo si se ha de finalizar el seguimiento, o tendrá el valor con el que ha de continuar en la siguiente trayectoria.

De este modo los desplazamientos serán continuos y progresivos si que se produzcan paradas innecesarias.

- El valor de velocidad VAVANCEmax, es la máxima velocidad que se le permite alcanzar a la silla, ya que al ser posible desarrollar velocidades muy altas para la navegación con ella, es necesario limitarla. Por tanto, se toma como máxima velocidad de avance 0,3m/s, velocidad que permite seguimientos con buenos resultados bajo condiciones de seguridad apropiadas.

b) En el caso de las trayectorias curvas se obtendrá para ellas una velocidad proporcional a su radio de curvatura calculado por el generador de trayectorias, de modo que en los giros con radios pequeños la velocidad sea cómoda y segura, mientras que para las curvas amplias se puedan desarrollar velocidades superiores.

- El valor de velocidad VGIROmax es el parámetro fijo del controlador para la velocidad de avance en tramos curvos, siendo su valor fijado a 0,1m/s para que el control sobre la velocidad de giro (Ω) no se vea entorpecido por esta variable.

Como puede observarse, la salida de velocidad de avance que se muestra en la figura 4.28 es mostrada como V´ ya que aún no se trata de la consigna que realmente se obtiene del controlador. Esta, según se representa en la figura 4.29 es tratada antes a través de un filtro de aceleración mediante el cual, además de generarse el perfil trapezoidal para las trayectorias rectas, permite detener o volver a poner en marcha la silla de manera progresiva en caso de ordenarse en cualquier momento una pausa en el seguimiento de las trayectorias.

De este modo, el parámetro de la aceleración de V (av), marca la aceleración o desaceleración que sufre la velocidad de avance en cada periodo de muestreo. Este es fijado a 0,02m/s^2, de modo que no se produzcan movimientos bruscos para garantizar la comodidad y evitar desplazamientos no controlados de las ruedas.

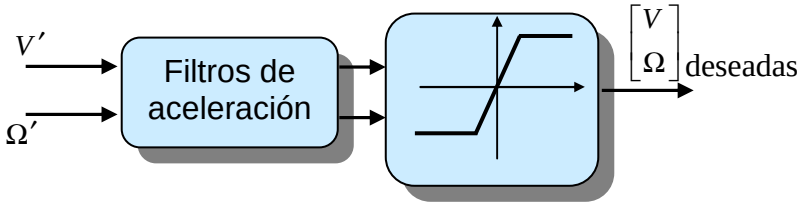


Figura 4.29. Filtro de aceleración y saturador para las consignas deseadas.

4.4.6.2. Control de la velocidad de giro (Ω).

La velocidad de giro (Ω) tiene una función bastante distinta a la planteada para la de avance (V). Esta variable permite a la silla seguir fielmente la forma de la trayectoria a recorrer, minimizando el error de posición que separa la localización del móvil del camino diseñado por el generador de trayectorias.

Para dicha tarea se ha optado por la implementación de un controlador clásico proporcional al error de posición, definido como el existente entre la posición deseada y la posición real para cada intervalo de muestreo.

El empleo de este tipo de control se debe a que la algoritmia a desarrollar es mucho más sencilla que la necesaria para otros tipos de controladores como pudieran ser controladores PID, controladores borrosos, robustos, etc., y los resultados obtenidos con este como se verá más adelante, son totalmente suficientes para realizar la tarea propuesta.

La figura 4.30 muestra el diagrama del controlador creado, en la que como puede observarse realmente se han desarrollado dos controladores totalmente similares, uno para cada tipo de trayectorias a seguir. Así, de manera sencilla se puede conseguir más fácilmente el ajuste de los parámetros, para que su respuesta se adapte mejor a la trayectoria a seguir.

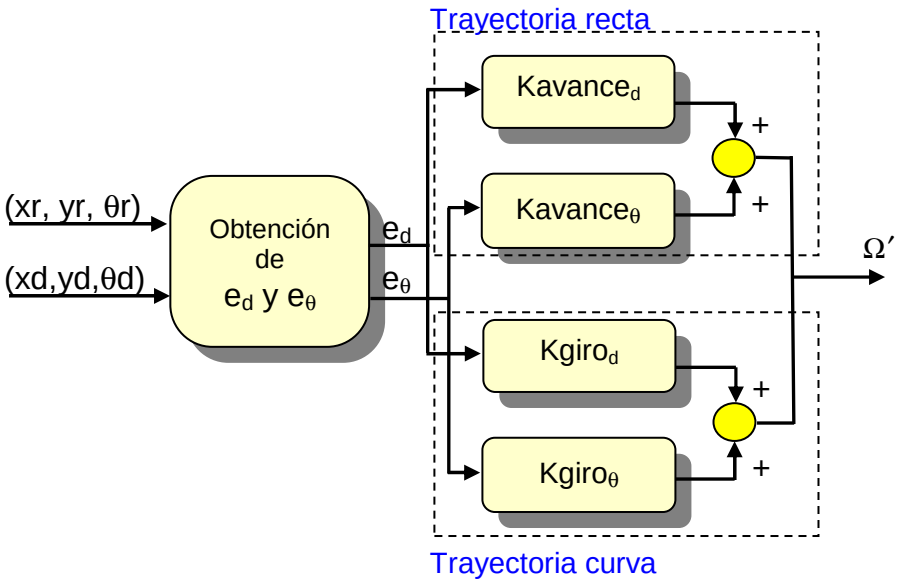


Figura 4.30. Diagrama de control de la velocidad de giro.

Las ecuaciones que describen el comportamiento del controlador son:

- Para trayectorias rectas:

$$\Omega_{avance}[n+1] = Kavance_d \cdot e_d[n] + Kavance_\theta \cdot e_\theta[n] \qquad [rad/s] \quad <4.23>$$

- Para trayectorias curvas:

$$\Omega_{giro}[n+1] = Kgiro_d \cdot e_d[n] + Kgiro_\theta \cdot e_\theta[n] \qquad [rad/s] \quad <4.24>$$

Las entradas a cada uno de los controladores son los errores de posición determinados por ed y eθ, obtenidos tal cual se detallaba al inicio de este apartado, ecuaciones <4.21><4.22>, y la salida, al igual que sucedía en el controlador de la velocidad de avance (V), es representada como Ω´ pues será tratada con otro filtro de aceleración y un saturador antes de obtenerse la consigna Ω al final de los procesos de control, como ya se mostraba en la figura 4.29.

De este modo, mediante el uso del filtro de aceleración con un valor de 0,07rad/s^2 se consiguen eliminar los cambios bruscos de la velocidad de giro, de manera que no se produzcan deslizamientos no controlados en las ruedas de la silla, y

empleando el saturador, se limita el valor máximo de la velocidad de giro para que este no supere nunca 2,62rad/s.

El ajuste de cada uno de los parámetros del controlador se ha realizado de manera experimental. Dada la similitud con el controlador empleado en el trabajo [02], se tomaron valores similares a los usados en este como punto de partida y se han modificando hasta conseguir ajustar los resultados de seguimiento a los deseados. De este modo, partiendo con errores de distancia (ed) constantes de aproximadamente 15cm, y que eran superiores a la distancia establecida para dar por alcanzado el nodo destino (10cm), se ha conseguido reducir dicho error a 7cm.

4.4.7. Cinemática de la silla.

Una vez que se obtiene tras la actuación de los dos controladores los parametros V y Ω, es necesario proporcionárselos al sistema de bajo nivel, donde se comportarán como consignas de velocidad angular de los motores.

Para ello se deben calcular las velocidades angulares deseadas de cada una de las ruedas (ω_I, ω_D) que hacen que la silla mantenga las consignas V y Ω, ya que al tratarse de un sistema no holonómico tal como se comentó en el modelado de la silla y depender el estado total del sistema de la actuación conjunta de las dos ruedas, se necesita obtener las consignas que se han de enviar al bajo nivel para que se actúe sobre cada uno de los motores.

Las ecuaciones a través de las que se obtienen los valores de las velocidades de las ruedas son las siguientes:

$$\omega_D = \frac{1}{R}\cdot\left(V + \Omega\frac{D}{2}\right)\cdot 1000 \qquad [mrad/s] \qquad <4.25>$$

$$\omega_I = \frac{1}{R}\cdot\left(V - \Omega\frac{D}{2}\right)\cdot 1000 \qquad [mrad/s] \qquad <4.26>$$

Donde **R** es el radio de las ruedas motrices de la silla y **D** la distancia de separación entre las dos ruedas motrices.

El bloque de la cinemática inversa es también descrito por estas mismas ecuaciones, pues tan solo hay que resolver el sistema formado por ambas para obtener los valore de las velocidades de avance y giro (V, Ω) reales, obteniéndose las ecuaciones:

$$V[n] = \frac{\omega_D[n] + \omega_I[n]}{2} \cdot R \quad [m/s] \qquad <4.27>$$

$$\Omega[n] = \frac{\omega_D[n] - \omega_I[n]}{D} \cdot R \quad [rad/s] \qquad <4.28>$$

La obtención de V y Ω reales se realizara a partir de estas ecuaciones y de los valores de las ω_I y ω_D reales de las ruedas cuyo proceso de obtención es explicado en los sucesivos apartados.

4.4.8. Comunicación con el bajo nivel.

La comunicación entre el alto nivel donde se implementa el controlador de posición y el bajo nivel donde se encuentran los motores junto con su driver de excitación y control, y sus encoderes, se realiza a través del protocolo Bus-CAN.

Para realizar el envío al bajo nivel de las consignas ω_I y ω_D es necesario codificarlas de manera que la tarjeta de potencia de los motores sepa interpretarlas, ya que los parámetros que ha de recibir son dos valores enteros ($Vcod_I$, $Vcod_D$) comprendidos de 0 a +127 y 0 a -127 según el sentido de giro deseado.

Según la información facilitada por el fabricante de la tarjeta de potencia [08], la relación entre el valor que se debe enviar y la velocidad desarrollada por el motor es la determinada por el ecuación <4.29>, donde **RPM** representa la velocidad del motor en revoluciones por minuto y **PPR** el numero de pulsos por revolución del encoder. **TimeBase** es un parámetro de la tarjeta de potencia para seleccionar el intervalo de tiempo entre dos lecturas del contador de pulsos de los encoders, el cual se mantiene con el mismo valor empleado en los trabajos realizados anteriormente con la silla, que fija este intervalo en 2,048ms.

$$Vcod = \frac{RPM \cdot PPR \cdot (TimeBase + 1)}{58593{,}75} \qquad <4.29>$$

Obtenidas las consignas $Vcod_I$ y $Vcod_D$ codificadas con 8 bits se monta el mensaje con ellas para ser mandado por Bus-CAN al bajo nivel, como se muestra en la tabla 4.2.

IDENTIFICADOR	**DATO A (32bits)**				**DATO B (32bits)**			
0x120							**$Vcod_D$**	**$Vcod_I$**

Tabla 4.2. Mensaje del Bus-CAN para el envío de consigas al bajo nivel.

En el bajo nivel, al recibirse un mensaje con el identificador 0x120, la electrónica asociada a la tarjeta de potencia para conectarla al Bus-CAN sabrá que el contenido de este pertenece a esta dos consignas y las despachará a la tarjeta para que actúe sobre los motores.

Esta misma electrónica es la encargada de solicitar a la tarjeta de potencia los datos referentes a los dos contadores de pulsos de los encoders y las marcas temporales de sus lecturas, para así ponerlos a disposición del alto nivel a trvés del Bus-CAN.

Los mensajes que se han de leer del bajo nivel tendrán la especificación de la tabla 4.3. Sus identificadores serán 0x101 para la rueda derecha y 0x102 para la izquierda y su contenido será el numero de cuentas realizadas en el encoder definidas mediante los 32 últimos bits del mensaje y su marca temporal del instante de la lectura por medio de los 32 primeros bits.

IDENTIFICADOR	**DATO A (32bits)**	**DATO B (32bits)**
0x101	**Tiempo Motor D**	**Contador Encoder D**
0x102	**Tiempo Motor I**	**Contador Encoder I**

Tabla 4.3. Mensajes del Bus-CAN para leer datos de los encoders.

De este modo, tal como se explico en el apartado referente al periodo de muestreo del controlador, este realiza constantes lecturas del Bus-CAN cada 100ms que es el periodo con el que se ponen estos mensajes a disposición, y al leerse un mensaje procedente del bajo nivel, extrae de él cada uno de los parámetros para implementar con ellos la realimentación que cierre el lazo de control.

4.4.9. Conversión de muestras.

Leídos el numero de cuentas y sus marcas temporales procedentes de los encoderes mediante la lectura de los mensajes 0x101 y 0x102, aplicando las ecuaciones <4.30><4.31> se obtienen los valores reales de ω_I y ω_D que tienen las ruedas.

$$\omega_D = \frac{P_D[n] - P_D[n-1]}{T_D[n] - T_D[n-1]} \cdot \frac{2\pi \cdot 1000}{4 \cdot PPR \cdot 32 \cdot Tps} \quad [mrad/s] \quad \text{<4.30>}$$

$$\omega_I = \frac{P_I[n] - P_I[n-1]}{T_I[n] - T_I[n-1]} \cdot \frac{2\pi \cdot 1000}{4 \cdot PPR \cdot 32 \cdot Tps} \quad [mrad/s] \quad \text{<4.31>}$$

Dado que se tratan de encoders incrementales, es necesario comparar la lectura actual **(P[n])** con la anterior **(P[n-1])** para obtener el numero de cuentas realizadas en el ultimo periodo de muestreo. De igual forma se ha de actuar con las marcas temporales **(T[n])** y **(T[n-1])**.

En las anteriores ecuaciones se observa que la constante **PPR** (200 pulsos por revolución del encoder) es multiplicada por 4, ya que se producen cuatro cuentas por cada pulso del encoder debido a los cuatro flancos que se generan en él al disponer de dos canales en cuadratura desfasados 90º.

La constante **TPS** es el intervalo de tiempo que representa cada incremento unitario de las marcas temporales cuyo valor es 96,15·10-6 segundos.

4.4.10. Deadreckoning.

El último paso antes de finalizar todo el proceso del algoritmo para el lazo de control implementado es el cálculo de la posición real en la que se encuentra la silla.

Para ello se realiza la integración del desplazamiento producido durante el último periodo de muestreo y la última posición calculada. A partir de las velocidades angulares (ω_I, ω_D) reales calculadas según el apartado anterior y haciendo uso de las ecuaciones de la cinemática inversa de la silla <4.27> y <4.28>, se obtienen

los valores de la velocidad de avance real y la velocidad de giro real (V y Ω reales).

A continuación se aplican V y Ω reales junto con las ecuaciones <4.32><4.33><4.34> propias de las técnicas de "deadreckoning", donde **Tm[n]**, <4.35>, corresponde al valor medio del incremento de tiempo existente entre dos lecturas medidas en la rueda derecha y la rueda izquierda y obtenido a partir de cada una de las marcas temporales devueltas por el bajo nivel.

$$\theta[n] = \theta[n-1] + \Omega[n] \cdot T_m[n] \qquad [rad] \qquad <4.32>$$

$$x[n] = x[n-1] + V[n] \cdot Cos(\theta[n]) \cdot T_m[n] \qquad [m] \qquad <4.33>$$

$$y[n] = y[n-1] + V[n] \cdot Sen(\theta[n]) \cdot T_m[n] \qquad [m] \qquad <4.34>$$

$$T_m[n] = \frac{T_D[n] + T_I[n]}{2} Tps \qquad [s] \qquad <4.35>$$

De este modo se obtiene la posición real, coordenadas y orientación, en la que se encuentra la silla en cada periodo de muestreo, la cual será comparada con la posición deseada en el siguiente periodo para dar comienzo de nuevo al siguiente ciclo del lazo.

Es importante remarcar de nuevo que, dado que se trata de un sistema de posicionamiento relativo al punto de origen desde donde se comienza el seguimiento de las trayectorias, y que se realiza el cálculo de la posición actual a partir de las posiciones anteriores según las técnicas de "deadreckoning", de la misma manera que se acumula el desplazamiento también se acumularán los errores, por lo que es importante disminuir en lo posible su producción, y tal como se hará mención en el capitulo de trabajos futuros, buscar alternativas para su corrección, ya que es imposibles detectar por el controlador, por ejemplo, el deslizamiento de las ruedas o la existencia de una pequeña diferencia entre ellas.

5. Resultados, conclusiones y trabajos futuros.

Descritas las tareas desarrolladas en este TFC para conseguir que la silla de ruedas siga de manera autónoma una ruta desde un origen hasta un destino seleccionado, a continuación se realiza un análisis de los objetivos alcanzados al finalizar el presente trabajo y se comentan posibles líneas en las que continuar desarrollando para así conseguir nuevas mejoras y avances.

5.1. Presentación de resultados.

En este apartado se presenta una pequeña muestra del seguimiento realizado de algunas trayectorias que conforman distintas rutas dentro del entorno de pruebas bajo el que se ha desarrollado este trabajo, de modo que el lector pueda hacerse una idea de los resultados obtenidos al finalizar el desarrollo de todos los procesos descritos en el proyecto, pues resultaría imposible poner un ejemplo de todas las posibles rutas.

Por ello, se han escogido tres ejemplos con los que se pretende cubrir el amplio abanico de posibles tipos de trayectorias que se pueden presentar.

5.1.1. Ejemplo 1.

El primer ejemplo escogido pertenece a la ruta más sencilla que puede darse, una única recta, por lo que la trayectoria generada para recorrerla también lo será. La

figura 5.1 muestra un fragmento del mapa de la aplicación generadora de rutas al completarse su recorrido, donde se seleccionó como origen un laboratorio situado en el fondo de uno de los pasillos de profesores y como destino el final de dicho pasillo.

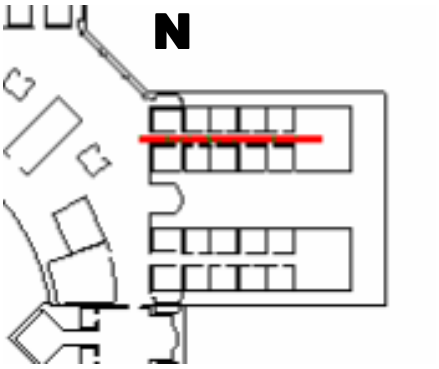


Figura 5.1. Ruta a través de un pasillo recto.

Por otro lado, la figura 5.2 detalla en color azul las posiciones alcanzadas por la silla, frente a las posiciones de la trayectoria deseada que se representan en color rojo. Además, se representan mediante círculos de color verde los nodos generados por la aplicación, pertenecientes a los nodos de origen y destino (situados en ambos extremos) y los nodos de las puertas del laboratorio y del pasillo (situados en el interior de la trayectoria) que ha de cruzar la silla.

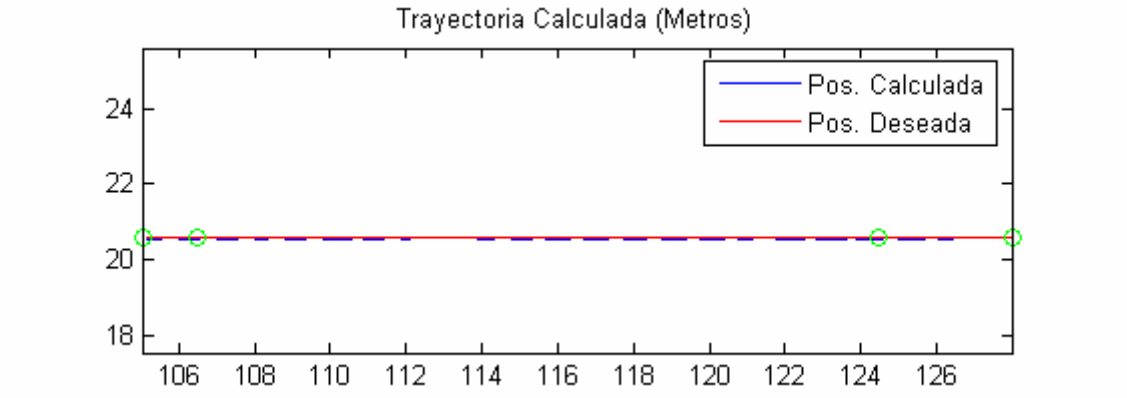


Figura 5.2. Detalle de la trayectoria seguida frente a la deseada.

La velocidad desarrollada durante la navegación es la mostrada en la figura 5.3, donde en el eje de ordenadas aparece el valor de esta dada en m/seg. y en el eje de abscisas se representa el número de muestra. En ella se observa como partiendo del reposo se produce una aceleración hasta alcanzar la velocidad máxima establecida (0,3m/s), y como este valor permanece constante durante toda la trayectoria, a excepción de los intervalos en los que la velocidad cae a

cero debido a que se realizaron tres pausas en mitad del recorrido (intervalos de muestras 90-160 265-265 365-400). Tras volver a ordenar la puesta en marcha de la silla, esta continúa el proceso de seguimiento de la trayectoria, acelerando hasta alcanzar de nuevo la velocidad deseada, y una vez alcanzado el punto de desaceleración por encontrarse próxima al nodo de destino, se vuelve a reducir de manera progresiva la velocidad hasta llegar a los 0,1m/seg. de modo que se llegue suavemente hasta el punto final, deteniéndose por completo al hacerlo y dándose por finalizada la trayectoria.

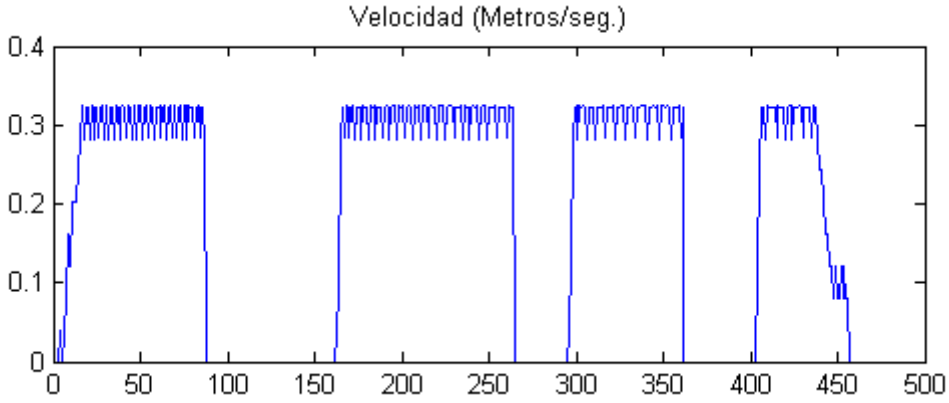


Figura 5.3. Velocidad de navegación desarrollada.

A continuación se presentan las figuras 5.4 y 5.5, donde se muestra los errores de desplazamiento y de orientación (ed, eθ) para cada una de las muestras tomadas y que se han obtenido durante el seguimiento de la trayectoria. Ambos errores, al comienzo de la trayectoria toman un valor nulo ya que la posición inicial de la silla corresponde con la posición deseada al empiece, pero a medida que se avanza, los errores van incrementándose hasta que son corregidos por el controlador.

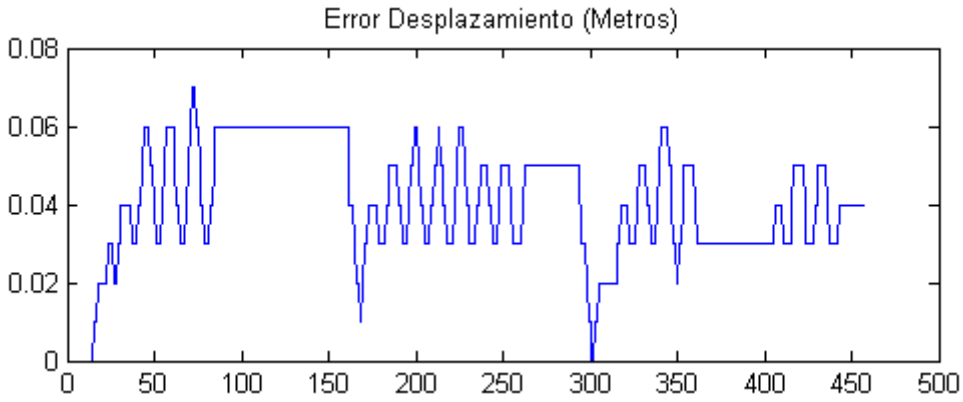


Figura 5.4. Error de desplazamiento producido durante la navegación.

Puede apreciarse que para esta trayectoria se ha logrado que el promedio del error de desplazamiento se encuentre entre los 3cm y 6cm, por lo que al ser inferiores que la distancia fijada para dar por alcanzados los nodos destino de cada trayectoria (10cm), se pueden asumir estas diferencias con las posiciones deseadas.

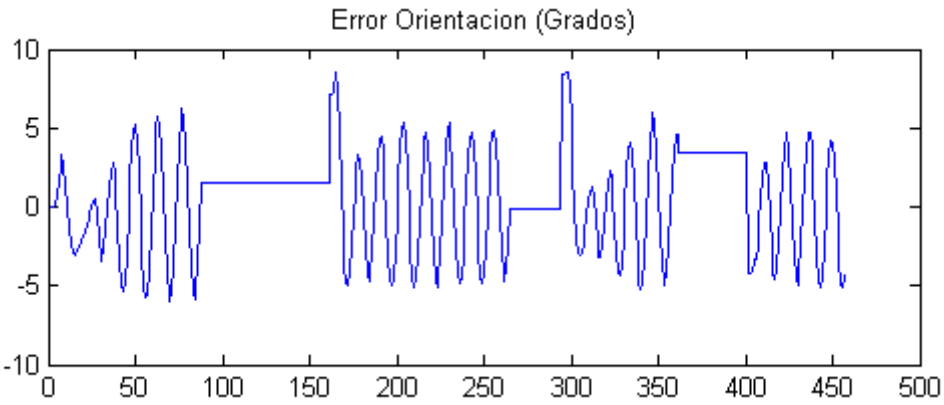


Figura 5.5. Error de orientación producido durante la navegación.

En la gráfica del error de orientación se ve como este varia entorno a los ±6º aproximadamente, por lo que apenas se aprecia el cabeceo de la silla para corregir la posición a lo largo del recorrido.

5.1.2. Ejemplo 2.

Para este segundo ejemplo se ha optado por la ruta que aparece en la figura 5.6, algo más compleja al tener que describirse dos trayectorias curvas en las que se ha de modificar la orientación de la silla 90º en cada giro. Como punto de origen se toma la puerta del primer pasillo de los despachos de profesores en la tercera planta, edificio Sur, y el destino fijado es la puerta de los baños del pasillo contiguo en el edificio Norte, de modo que la silla debe circular a través del bloque de comunicaciones existente entre ambos edificios.

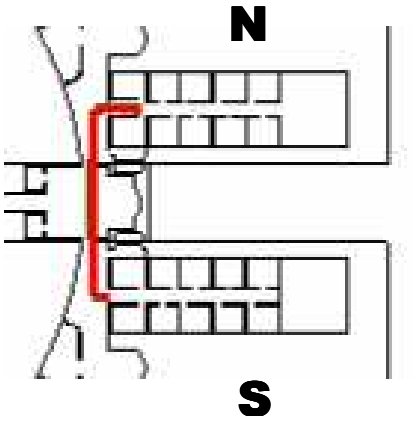


Figura 5.6. Ruta a través del bloque de comunicaciones.

De nuevo la figura 5.7. detalla la trayectoria seguida por la silla junto con la deseada, donde al igual que en el ejemplo anterior también se representan los nodos que conforman la ruta. En este caso se ha incluido una ampliación de cada uno de los tramos curvos para poder observarse con mayor detalle los movimientos realizados en esos intervalos del recorrido, figuras 5.8 y 5.9.

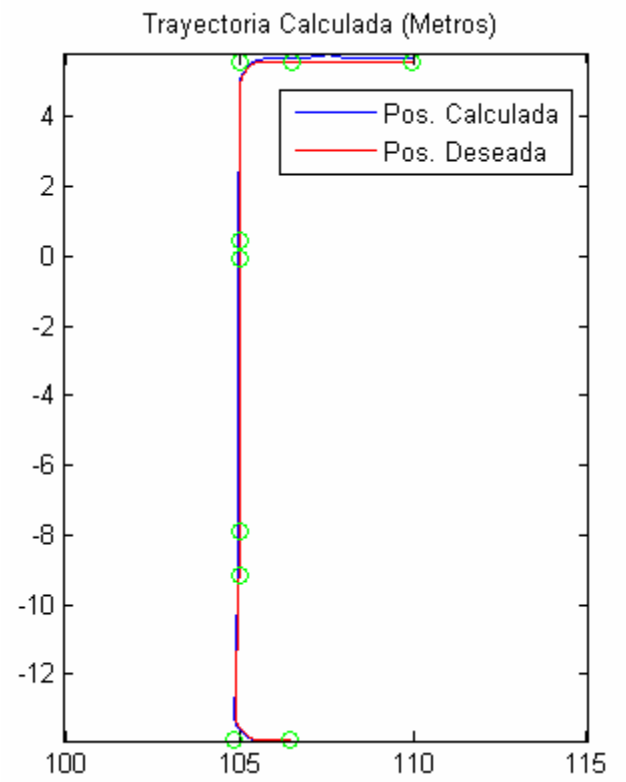


Figura 5.7. Detalle de las trayectorias seguidas frente a las deseadas.

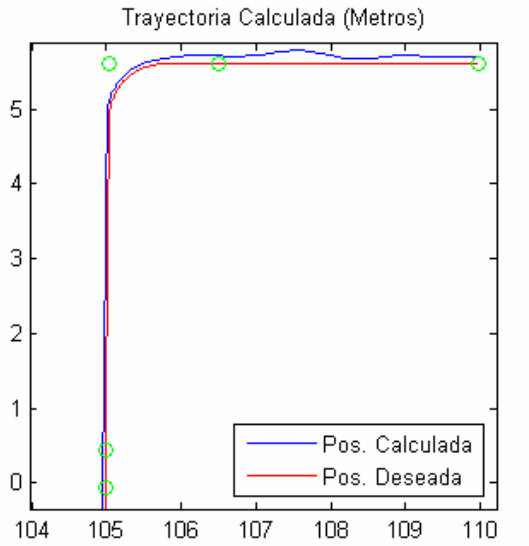


Figura 5.8. Detalle de la segunda curva de 90º.

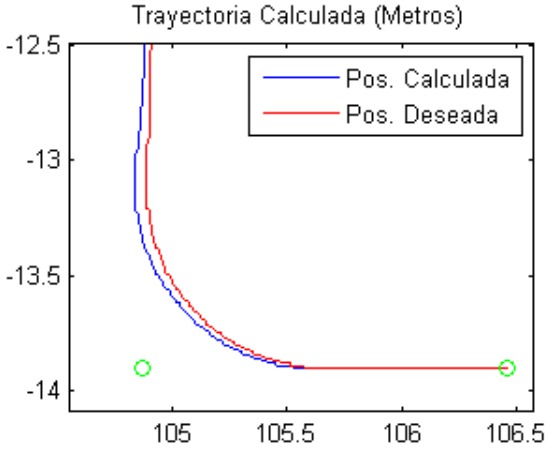


Figura 5.9. Detalle de la primera curva de 90º.

Las velocidades alcanzadas por la silla durante todo el recorrido realizado en este ejemplo son mostradas en la figura 5.10.

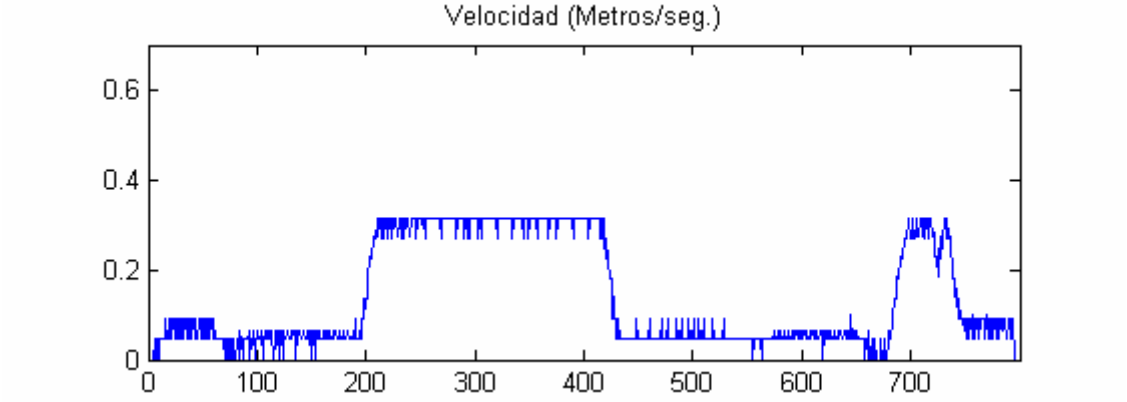


Figura 5.10. Velocidad de navegación desarrollada por la silla.

Al inicio de la trayectoria la silla parte del reposo y acelera, pero al deber realizar la primera curva del recorrido muy pronto, esta no alcanza la velocidad máxima y tan solo llega a una velocidad aproximada de 0,1m/seg. A continuación, se observa una pequeña disminución de la velocidad durante el transcurso de la primera trayectoria curva (intervalo de muestras 65-195) para una vez finalizada, volver a producirse una aceleración que haga conseguir a la silla la velocidad máxima de 0,3m/seg. ya que esta vez al tratarse de un tramo amplio lo permite (intervalo de muestras 195-420). Una vez finalizado el amplio tramo que pasa a través del bloque de comunicaciones y separa las dos trayectorias curvas, la silla disminuye su velocidad hasta conseguir la velocidad adecuada para afrontar la segunda tarea de giro y la mantiene hasta finalizarla (intervalo de muestras 420-680). Alcanzado el último tramo, al tratarse de una recta, se vuelve a incrementar la velocidad obteniéndose de nuevo la máxima, y más tarde la silla desacelera hasta llegar a una velocidad reducida de 0,1m/seg. pues se encuentra próxima al nodo destino (intervalo de muestras 680-790), para finalmente detenerse.

Las figuras 5.11 y 5.12 vuelven a presentar los errores de desplazamiento y orientación (ed, eθ) producidos a lo largo de seguimiento de las distintas trayectorias que forman la ruta del ejemplo.

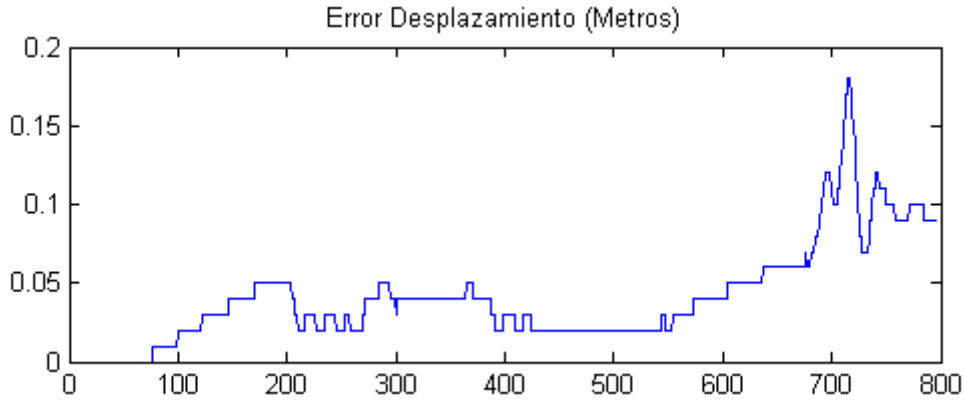


Figura 5.11. Error de desplazamiento producido durante la navegación.

Para el caso del error de desplazamiento se vuelve a ver como inicialmente el error se mantiene nulo o inapreciable, ya que al ser el comienzo del seguimiento, las posiciones real y deseada coinciden, y al realizarse los primeros movimientos a bajas velocidades, este error se mantiene durante más tiempo. Después, al inicio de la primera curva, tal como podía verse en la figura 5.8, el error de desplazamiento aumenta ya que la silla se aleja de su trayectoria deseada produciéndose un error comprendido entre los 2cm y 5cm. Finalmente, al

desarrollar la segunda curva el error vuelve a sufrir un incremento, alcanzándose en este caso valores más altos, 18cm, pero son rápidamente corregidos por el controlador de modo que el error vuelve a ser inferior a 10cm.

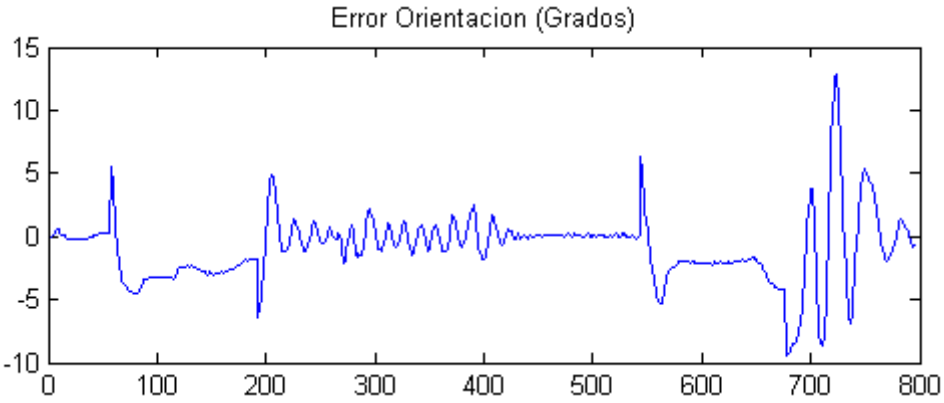


Figura 5.12. Error de orientación producido durante la navegación.

Para el caso del error de orientación, se puede ver que de nuevo se tiene el error comprendido principalmente entre los ±5º, produciéndose valores mayores únicamente en las últimas muestras, correspondientes al seguimiento de la segunda curva y el posterior tramo recto, al igual que sucedía con el error de desplazamiento.

5.1.3. Ejemplo 3.

Para finalizar, se presenta este último ejemplo más completo, formado por trayectorias rectas de distintas longitudes y diferentes curvas que modifican la orientación 45º y 90º tal como se observa en la figura 5.13. Para ello, se seleccionó en la aplicación generadora de rutas la puerta del los baños del pasillo 1, situado en la tercera planta del edificio Este como punto de origen, y como punto de destino de la ruta se escogió la puerta del despacho 314, situado en primer pasillo de la tercera planta del edificio Sur.

De igual modo que en los ejemplos anteriores, a también se presentan las figuras que detallan los desplazamientos realizados por la silla junto con las trayectorias que se desean seguir, donde la figura 5.14 muestra la ruta completa y las figuras 5.15, 5.16 y 5.17 exponen ampliaciones de algunos tramos para poder mostrar con mayor detalle lo sucedido durante el proceso de seguimiento.

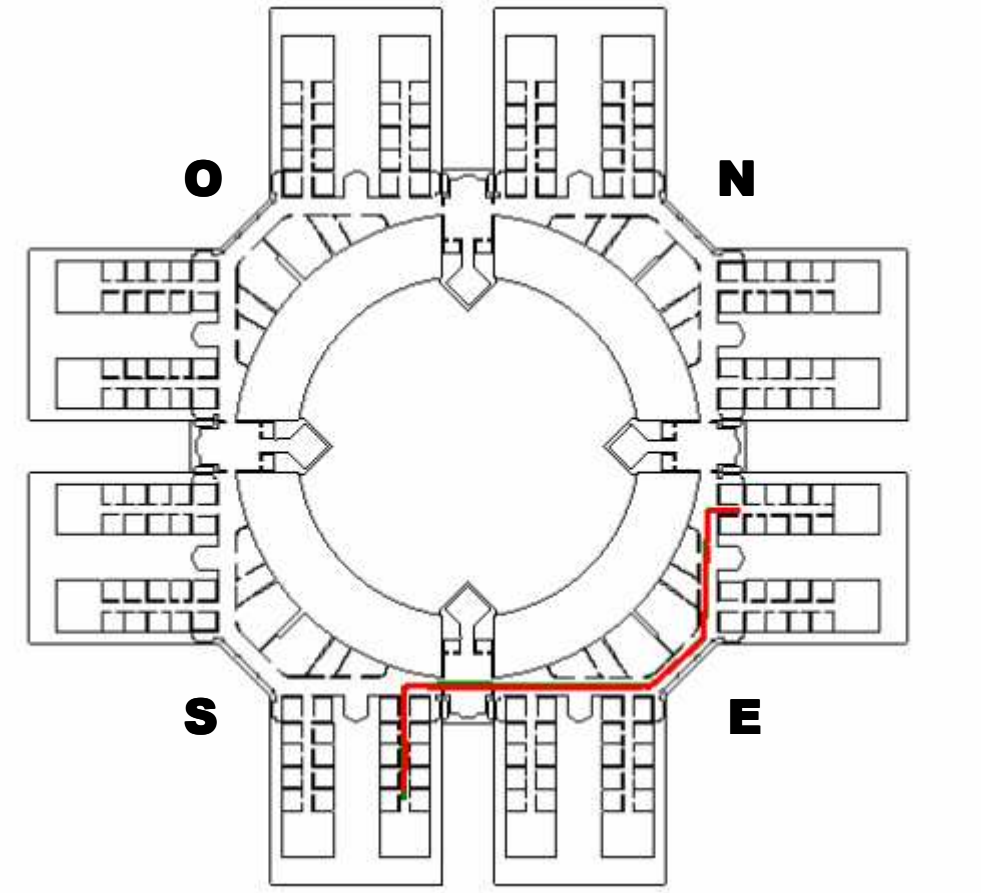


Figura 5.13. Ruta a través de distintas zonas del edificio.

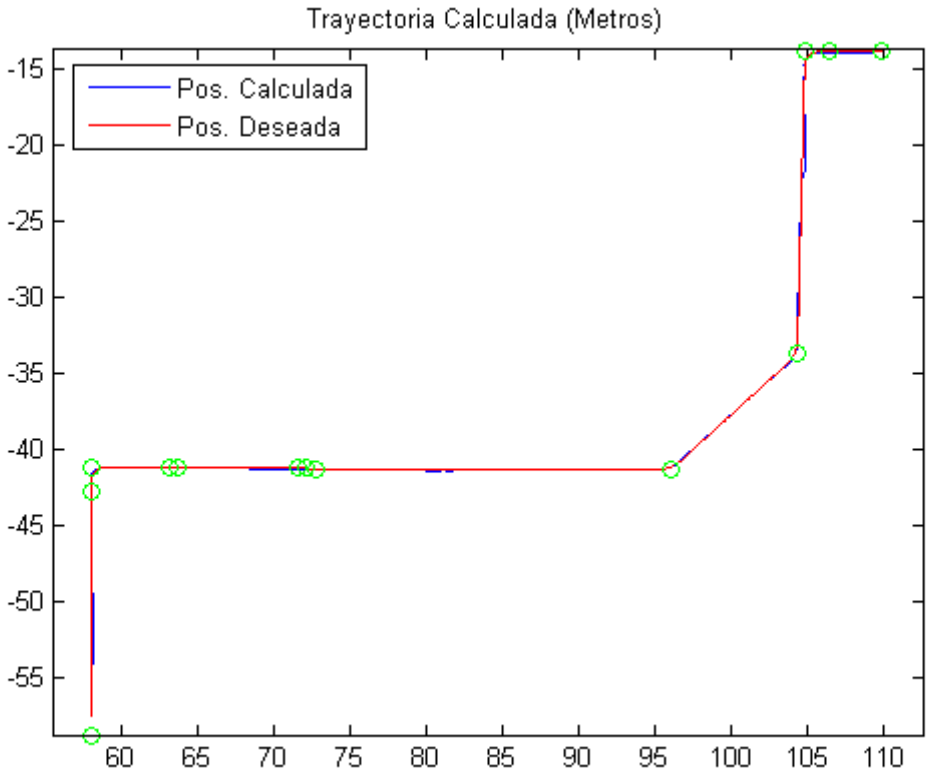


Figura 5.14. Detalle de la trayectoria seguida frente a la deseada.

Además, como puede verse en la figura 5.13 y la figura 5.17, para hacerlo más completo este ejemplo también cuenta con una pausa final ordenada por el usuario que detiene la silla a unos centímetros antes de llegar a su destino.

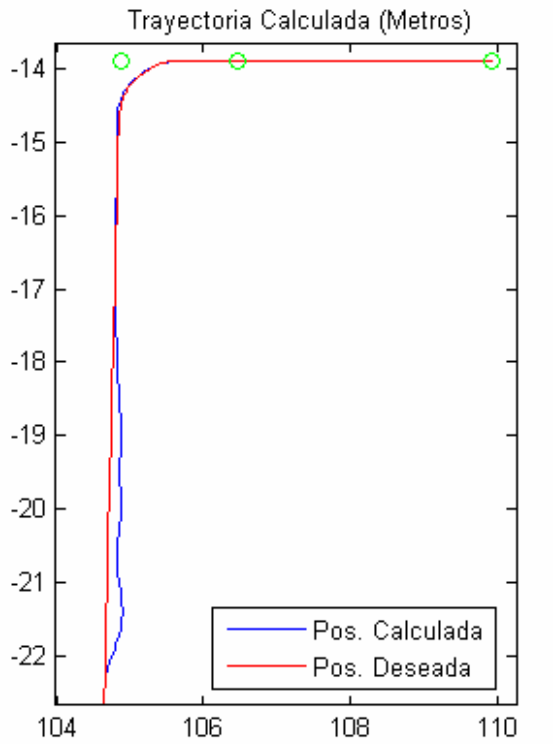


Figura 5.15. Detalle de la primera curva de 90º.

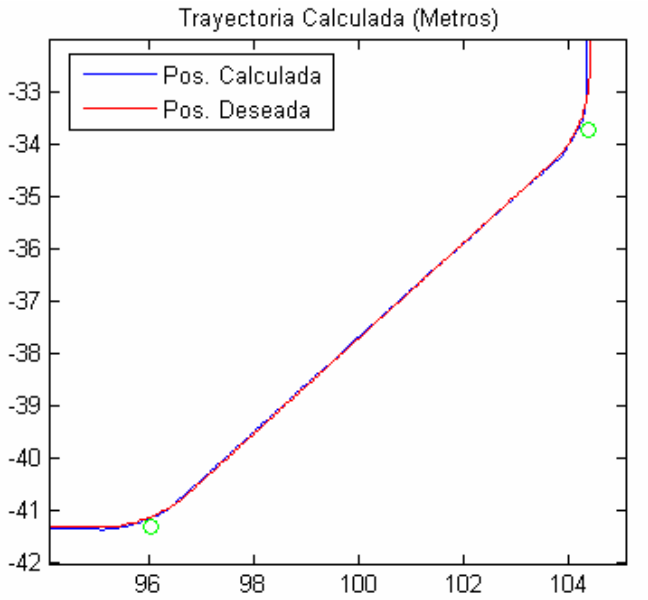


Figura 5.16. Detalle de las curvas de 45º.

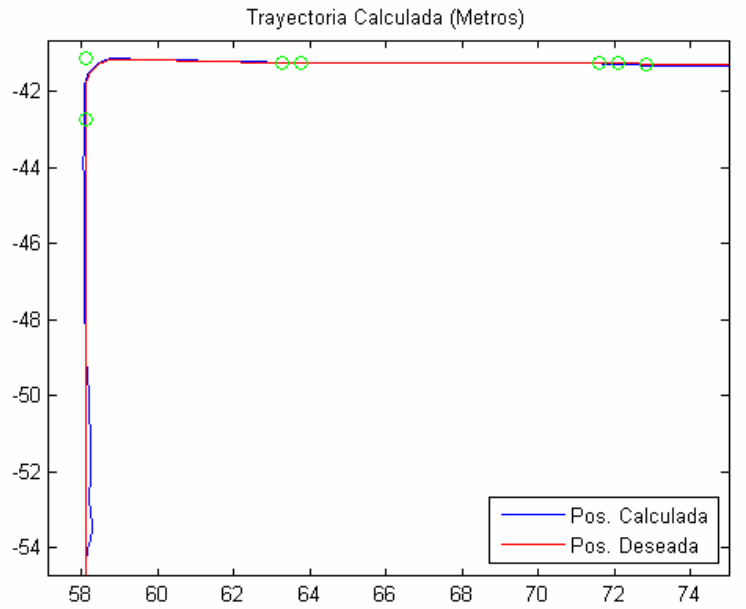


Figura 5.17.Detalle de la segunda curva de 90º.

En la figura 5.18 se puede ver las distintas velocidades alcanzadas por la silla según se ha realizado el seguimiento de las distintas trayectorias de la ruta elegida.

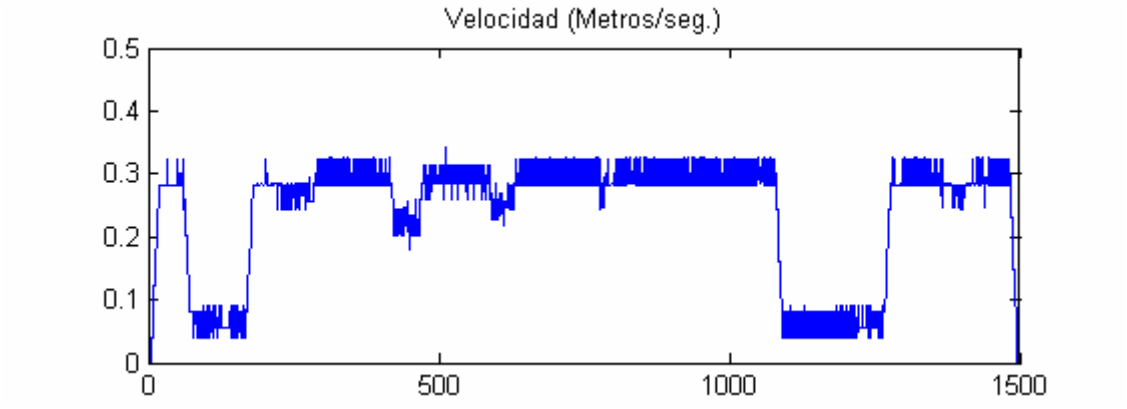


Figura 5.18. Velocidad de navegación desarrollada.

Inicialmente se acelera hasta alcanzar la velocidad máxima durante el primer tramo recto (intervalo de muestras 1-60). A continuación, se reduce la velocidad hasta los 0,05m/seg. para describir la primera trayectoria curva (intervalo de muestras 60-170) y una vez finalizada se vuelve a alcanzar y mantener la velocidad de 0,3m/seg. hasta llegar a la segunda curva (intervalo de muestras 170-420). En este caso (intervalo de muestras 420-470), al tratarse de una curva menos cerrada que la anterior, la velocidad de 0,2m/seg. alcanzada y con la que se describe esta, es superior a la primera ya que como se explico en el tema anterior, la velocidad de avance durante las trayectorias curvas es proporcional a

su radio. Se vuelven a desarrollar los 0,3m/seg. para el tramo que une las dos curvas centrales y después vuelve a disminuir hasta 0,24m/seg. durante la segunda de ellas (intervalo de muestras 590-630). De nuevo, se vuelve a transitar a la velocidad máxima hasta llegar a al a última curva (intervalo de muestras 470-1080), donde la silla reduce su velocidad a 0,6m/seg. (intervalo de muestras 1080-1265). Finalmente, tras terminar la última tarea de giro, la silla vuelve a alcanzar la velocidad máxima hasta que es detenida por el usuario poco antes de llegar a su destino.

Al igual que en los ejemplos anteriores, las graficas que se muestran a continuación en las figuras 5.19 y 5.20 son las pertenecientes a los errores de desplazamiento y orientación (ed,eθ).

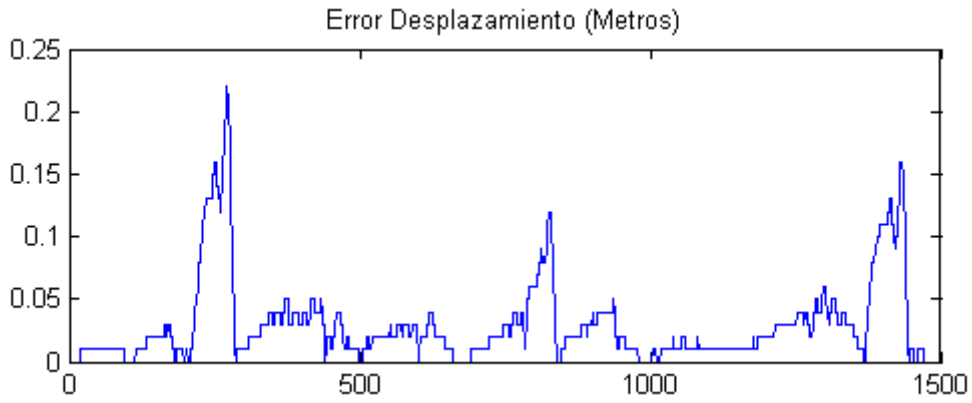


Figura 5.19. Error de desplazamiento producido durante la navegación.

Se puede observar que el error de desplazamiento cometido por la silla durante la mayor parte la ruta no supera los 6cm. Tan solo en tres ocasiones se producen distanciamientos importantes, apareciendo en esta grafica picos superiores al valor mencionado, que vuelven a ser corregidos por medio de la actuación del controlador. En este caso, estos picos se dan durante el segundo, el cuarto y el quinto tramo recto.

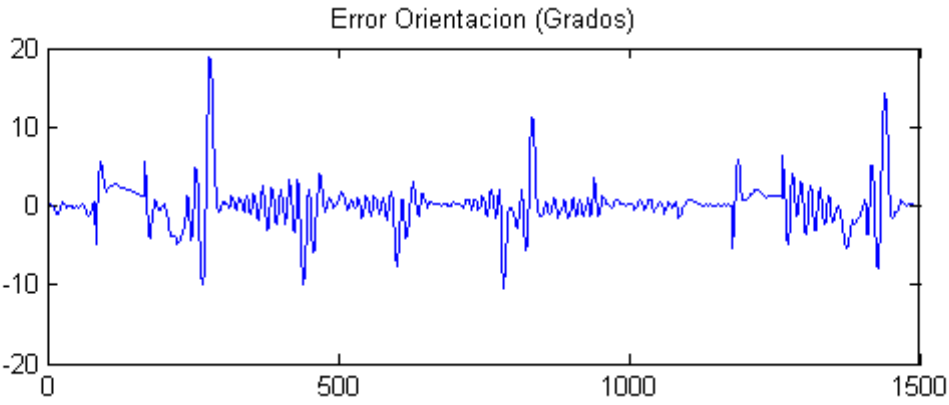


Figura 5.20. Error de orientación producido durante la navegación.

Para el error de la orientación, los resultados son similares, con un valor aproximado de ±6º durante toda la ruta y tres picos más acentuados que coinciden con los producidos para el error de desplazamiento.

5.2. Conclusiones obtenidas.

Tras exponer los ejemplos del apartado anterior en los que se muestran los resultados obtenidos al recorrer distintas rutas, a continuación se presentan las conclusiones que se obtienen a partir de ellos y de las experiencias con la silla al efectuar estos y otros desplazamientos.

Como puede verse en las distintas graficáis, tanto de trayectorias como de los errores producidos, la precisión obtenida para las tareas de seguimiento cumple perfectamente la buscada para este proyecto, pues el error prácticamente constante de aproximadamente 6cm, es totalmente despreciable teniendo en cuenta las trayectorias seguidas por la silla. Si bien es cierto que en ocasiones aparecen errores superiores, estos son corregidos por el controlador implementado, de modo que la silla vuelve a tomar las posiciones deseadas de manera suave.

De este modo, mediante la actuación del controlador y las trayectorias diseñadas, se ha conseguido que los desplazamientos realizados por la silla sean siempre movimientos confortables, algo muy importante para la tarea a realizar. Además, la ausencia de aceleraciones pronunciadas y movimientos bruscos ayuda a disminuir los errores producidos por los deslizamientos de las ruedas.

Estos deslizamientos, así como otros factores como pueden ser pequeñas diferencias entre las ruedas motrices de la silla, o un pequeño error en la posición y orientación de la silla en el inicio de la ruta, producen la aparición de errores acumulativos debido al empleo de las técnicas de "deadreckoning" para la estimación del la posición del móvil en cada periodo de muestreo, los cuales son imposible de detectar por la silla con los medios empleados en el presente proyecto, y que a pesar de ser pequeños, según las trayectorias a realizar pueden convertirse en errores importantes.

Por lo expuesto en el párrafo anterior, queda justificado decir que la navegación, haciendo tan solo uso de la odometría proporcionada por los encoders de la silla, no es practica, ya que será necesaria acompañarla de algún sistema que permita corregir los errores acumulados.

5.3. Trabajos futuros.

Para finalizar el capitulo, se plantean una serie de propuestas a desarrollar para completar lo ya conseguido en este proyecto.

La principal necesidad a abordar es la corrección de los errores acumulativos, pues como se ha mencionado anteriormente, son imposibles de detectar por la silla actualmente. Por tanto, seria necesario dotarla de un medio alternativo que apoye a la odometría y las técnicas de "deadreckoning". Una posible propuesta es el empleo de sistemas de visión capaces de calcular la posición real de la silla a partir del reconocimiento de objetos fijos en el entorno, de forma que el peso de los procesos para el seguimiento de las trayectorias recaiga en la odometría tal como se ha realizado, pues como se ha visto es fácil de llevar a cabo y con buenos resultados, y mediante el uso por ejemplo de un filtro de Kalman, con el que es posible realizar el filtrado, estimación y fusionado de datos, emplear técnicas de visión artificial para realizar la corrección de la posición estimada por la silla.

Además, la visión podría emplearse para dotar al móvil de mayor exactitud en los momentos en que fuese necesaria, como por ejemplo al circular por pasillos o puertas estrechas. También sería muy útil en el desarrollo de las tareas para salir de los ascensores. En el presente trabajo estas no se ha llevado a cabo pues, si

bien se han tenido en cuenta los cambios de plantas a la hora de tratar la ruta y obtener las trayectorias que la forman, así como de dotar a los procesos de seguimiento de un mecanismo para realizar la espera durante el viaje en el ascensor tras detenerse en él de forma automática, se ha dejado para trabajos futuros el implementar las rutinas que permitan a la silla salir de los ascensores y colocarse en la posición correcta para continuar las trayectorias de la nueva planta, pues solo con odometría resulta imposible alcanzar dicha posición sin tener problemas con el entorno por el que se mueve la silla. De cualquier modo, debido a la posición de los nodos de los ascensores suministrados por la aplicación generadora de rutas, los viajes en los ascensores del edificio empleado como entorno de pruebas, quedan descartados, pues por las dimensiones de estos y de la silla, es imposible alcanzar su centroide, que tal como se mencionó en la descripción de la aplicación, es la posición del ascensor suministrada. Será por tanto necesario realizar modificaciones en los nodos de las rutas que precisen viajar en ascensores.

A pesar de disponer la plataforma hardware de un sistema de sensores de ultra sonidos distribuidos por toda su estructura, se señala el empleo de visión para las tareas descritas en los anteriores párrafos, debido a la problemática que conlleva el uso de este tipo de sensado, ya que para su correcto funcionamiento, el objeto a detectar ha de encontrarse dentro de un margen de distancia, y debido a la gran variedad de espacios que se pueden encontrar en edificios como bajo el que se ha desarrollado el proyecto, los hacen poco recomendables. Además, la respuesta de los sensores de ultrasonidos se ve afectada en gran medida por la orientación de las superficies sobre las que se reflejan sus ecos.

Pero por otro lado, al igual que los acelerómetros de los que también dispone la silla, los sensores de ultrasonidos si podrían emplearse para la detección de obstáculos y evitar así posibles colisiones.

Otros desarrollos posibles que continúen y mejoren el trabajo realizado en el presente proyecto pueden ser implementar la interacción de la silla con el edificio por el que transita. Si se dotase a este de redes de comunicación inalámbricas, de la gestión de datos y procesos necesarios, y los medios para actuar sobre los distintos elementos del propio edificio, se alcanzaría una automatización mayor de los desplazamientos, pues se podrían hacer posibles tareas como la llamada y

manejo automático a los ascensores o la apertura de puertas por las que se deba cruzar. Realizar este tipo de operaciones puede resultar difíciles para algunos usuarios si el medio no se encuentra bien adaptados a sus necesidades.

También podría llevarse a cabo el desarrollo de la gestión vía Web de todos los datos propios de los edificios en los que se desee emplear el trabajo aquí realizado, tanto de los ficheros necesarios para obtener las rutas como de aquellos parámetros empleados que son propios del edificio. Un ejemplo puede ser la distancia mínima entre dos nodos muy próximos para que no sean eliminados, descrita en el apartado 4.3.3.1 ya que esta puede variar de un edificio a otro. De este modo, el usuario tan solo necesitara una conexión a Internet para actualizar la información de su aplicación y poder hacer uso de ella inmediatamente. La figura 5.21 hace idea de lo expuesto en este párrafo.

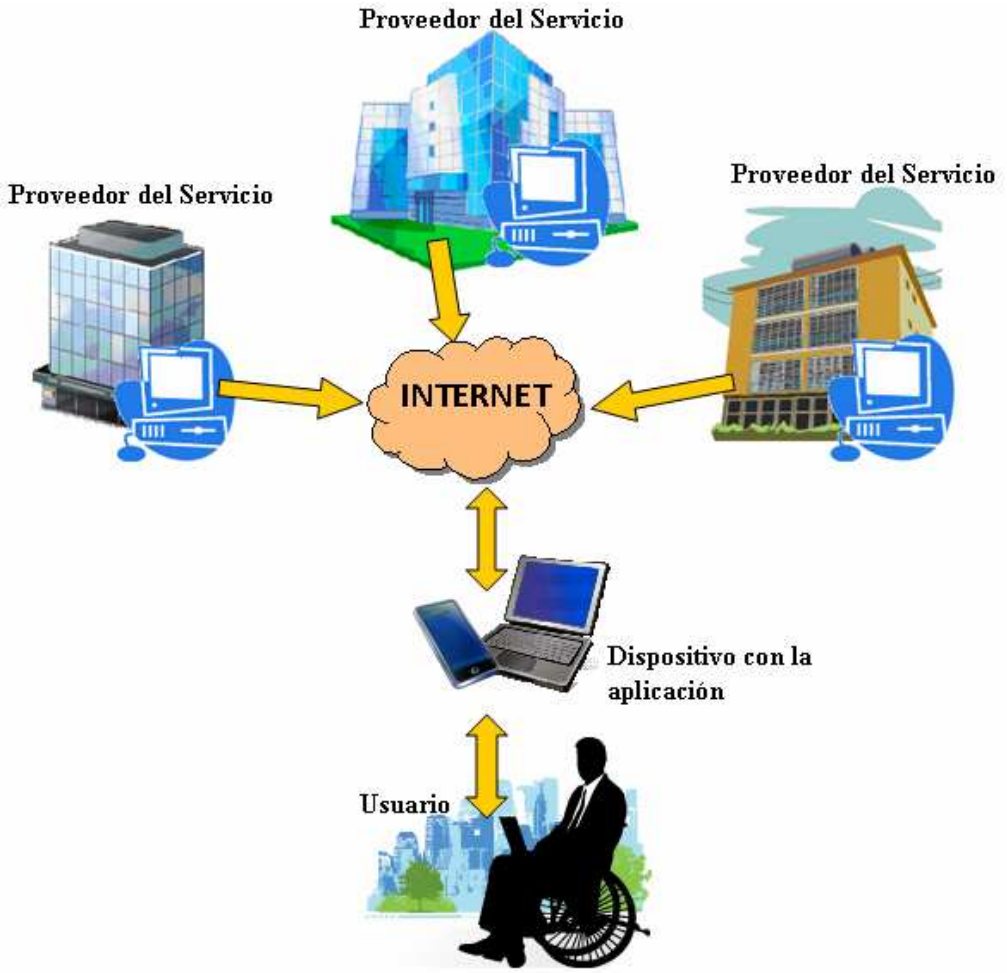


Figura 5.21. Disponibilidad de los datos para la aplicación.

Por último, sería interesante intentar realizar un ajuste más óptimo del controlador de posición. Gracias a la identificación del bajo nivel realizada, se podría conseguir ajustar las constantes del controlador mediante simulación, de modo que los errores producidos sean corregidos a valores menores y más constantes, pues como puede verse principalmente en el primer ejemplo, a pesar de que el error de orientación no toma valores elevados, el controlador no es capaz de conseguir eliminar las oscilaciones casi periódicas que se producen. A pesar de que estas oscilaciones no son apreciables por el usuario de la silla, estas podrían dar lugar a que el móvil se desorientase, por lo que seria adecuado intentar eliminarlas.

III. PLIEGO DE CONDICIONES

En esta sección se detallan las cualidades técnicas de los equipos y herramientas empleadas, así como el software requerido para la realización del proyecto.

1. Equipos Físicos.

- Ordenador portátil Intel Core Duo (desarrollo y tareas de navegación)

Microprocesador	Intel-Core
Velocidad de reloj	1.83GHz
Memoria RAM	2.00GB
Disco duro	120GB
Monitor auxiliar	19 Pulgadas

- Impresora HP LaserJet Pro 200

Modo de impresión	Láser
Velocidad de páginas	14ppm
Buffer de entrada	128Mbytes
Comunicaciones	USB/Ethernet
Resolución	600ppp

- Conversor USB-CAN de LAWICEL.

- Conversor USB-Serie RS232

- Silla de ruedas GARANT del modelo 63E-PRO
  Dos motores del tipo GP76.50Br0.4.
  Frenos (electroimán) del tipo 73 341-05A10 de la marca BINDER.
  Dos encoders IG16-0065 ABI/200/PP de la casa SIKO.
  Batería de alimentación de 24V para el sistema completo.
  Tarjeta de potencia AX3500 de ROBOTEQ para excitar los motores.
  Tarjetas LPC2129 de Embedded Artists para el control de los distintos módulos de la electrónica empleada.
  Ordenador VIA Nehemian a 1,20GHz, con 1GB de RAM y 40GB de disco duro.
  Pantalla táctil.
  Sensores de ultrasonido SRF02.
  Acelerómetro ADXL330.

2. Equipos Lógicos.

- Sistema operativo Windows XP.
- Entornos de programación: Visual Studio 2005/2008.
- Matlab 7.9 y su ToolBox de identificación.
- Drivers para conversor LAWICEL.
- Librería de LAWICEL.
- Procesadores de texto: Microsoft Word 2007 y otros sin formato.
- Editores gráficos: SmartDraw6 y otros.
- Herramientas para comparación de ficheros fuentes: Beyond Compare3.
- Navegador web para visualización de ficheros XML.

IV. PRESUPUESTO

En esta sección se realiza una estima del coste que supone la ejecución del proyecto. Para ello se realizará un estudio dividido en diversos apartados, en los cuales se agrupan los gastos según su origen y finalmente se calcula el total que conlleva su desarrollo.

1. Coste de los materiales.

En este apartado se engloba el precio del uso de los diversos equipos empleados para desarrollar el presente trabajo, describiendo tanto el precio de la parte hardware como el de la parte software. Por último, se hará un pequeño resumen del conjunto de material de oficina utilizado durante la realización del proyecto.

- Recursos hardware:

EQUIPO	PRECIO	UNIDADES	TOTAL
Silla de ruedas	9.000 €	1	9.000 €
Encoder	54,27 €	2	108,54 €
Mecanizado encoder	2.000 €	2	4.000 €
Batería	198 €	1	198 €
Cargador de Batería	135 €	1	135 €
Tarjeta LPC2129	55 €	2	110 €
Tarjeta AX3500	522 €	1	522 €
Conversor USB-CAN	132 €	1	132 €
Conversor USB-RS232	25 €	1	25 €

Tabla IV. 1. Coste de los recursos hardware.

Coste total de los recursos hardware 14.230,54€

• Recursos software:

EQUIPO	COSTE
Ordenador Intel Core Duo 1,83GHz	1.200 €
Impresora HP LaserJet Pro 200	200 €
Visual Studio	615 €
Microsoft Office	350 €
Matlab	5.000 €

Tabla IV. 2. Coste de los recursos software.

Coste total de los recursos software 7.365 €

• Material de oficina:

MATERIAL	COSTE
Material fungible	60 €
Material no fungible	30 €

Tabla IV. 3. Coste del material de oficina.

Coste total del material de oficina 90 €

Sumando cada uno de los totales que han producido los distintos materiales empleados se obtiene el total procedente de estos medios:

RECURSO	COSTE
Recursos de Hardware	+ 14.230,54 €
Recursos de Software	+ 7.365 €
Material de oficina	+ 90 €
Coste total de los materiales	**21.685,54 €**

Tabla IV. 4. Coste total de los materiales.

El coste total de los materiales asciende a VEINTIUN MIL SEISCIENTOS OCHENTA Y CINCO euros con CINCUENTA Y CUATRO céntimos.

2. Coste de la mano de obra.

La realización de este proyecto ha sido llevada a cabo por las siguientes personas:

FUNCION	COSTE / HORA	TOTAL HORAS
Ingeniería	50 €	950 horas
Mecanografiado	12 €	100 horas

Tabla IV. 5. Coste de la mano de obra.

Coste total de la mano de obra 48.700 €

El coste total de la mano de obra asciende a CUARENTA Y OCHO MIL SETECIENTOS euros.

3. Presupuesto de ejecución de material.

Es la suma total de los importes del coste de materiales y de la mano de obra.

RECURSO	COSTE
Coste total del material	+ 21.685,54 €
Coste total de la mano de obra	+ 48.700 €
Coste de ejecución de material	**70.385,54 €**

Tabla IV. 6. Presupuesto de ejecución de material.

El presupuesto total de ejecución del material asciende a SETENTA MIL TRESCIENTOS OCHENTA Y CINCO euros con CINCUENTA Y CUATRO céntimos.

4. Importe de ejecución por contrata.

A continuación se enmarcan los gastos generados por el uso de las instalaciones utilizadas durante la realización del proyecto, cargas fiscales, gastos financieros, tasas administrativas y derivados de las obligaciones de control del proyecto. De

igual forma se incluye el beneficio industrial. A efectos de cálculo se estima este apunte como el 22 % del presupuesto de ejecución material del proyecto.

RECURSO	COSTE
Coste ejecución de material	+ 70.385,54 €
Gastos financieros, beneficios, etc…	+ 22% 15.484,82 €
Coste total de ejecución por contrata	**85.870,36 €**

Tabla IV. 7. Coste total de ejecución por contrata.

El importe de ejecución por contrata asciende a OCHENTA Y CINCO MIL OCHOCIENTOS SETENTA euros con TREINTA Y SEIS céntimos.

5. Honorarios facultativos.

Los honorarios facultativos por la ejecución de este proyecto se determinan de acuerdo a las tarifas de los honorarios de los ingenieros en trabajos particulares vigentes a partir de 1 de septiembre de 1997 dictadas por el Colegio Oficial de Ingenieros de Telecomunicación, y que para el presente año corresponden con los siguientes.

IMPORTE	COEFICIENTE REDUCTOR	PORCENTAJE
Hasta 30.050 €	C = 1	7%
De 30.050 € a 60.101 €	C = 0,9	7%
De 60.101 € a 90.152 €	C = 0,8	7%

Tabla IV. 8. Tarifas honorarios facultativos.

Los derechos de visado se calculan aplicando la siguiente fórmula:

$$0{,}07 \cdot P \cdot C$$

Donde:

P es el presupuesto de ejecución de material.

C es el coeficiente reductor.

Importe derechos de visado:

Total derechos de visado = 0,07 · 85.870,36 · 0,8 = 4.808,74 €

El importe total de los derechos de visado es de CUATRO MIL OCHOCIENTOS OCHO euros con SETENTA Y CUATRO céntimos.

6. Presupuesto total.

El importe del presupuesto total de este proyecto es la suma del presupuesto por contrata y los honorarios facultativos.

CONCEPTO	COSTE
Presupuesto por contrata	+ 85.870,36 €
Honorarios facultativos	+ 4.808,74 €
Subtotal	**90.679,10 €**
I.V.A. (21%)	+ 19.042,61 €
Presupuesto final	**109.721,71 €**

Tabla IV. 9. Presupuesto total.

El importe total de este proyecto asciende a la cantidad de CIENTO NUEVE MIL SETECIENTOS VEINTIUN euros con SETENTA Y UN céntimos.

En Alcalá de Henares, a 9 de Mayo de 2013.

El Ingeniero Técnico de Telecomunicación

Fdo.: Pedro del Moral Peña.

V. MANUAL DE USUARIO

En este capítulo se procede a describir la aplicación desarrollada para llevar a cabo las tareas de navegación a través del edificio. Como ya se ha mencionado en anteriores puntos de la memoria, al ser este proyecto continuación del trabajo [01], en el que se realizaron todas las tareas para la obtención de las rutas, se ha empleado la misma aplicación desarrollada en él, añadiéndose el diseño y procesos correspondientes al actual proyecto.

1. Descripción de la interfaz.

Al arrancar la aplicación **Navegación_interiores** desarrollada en el entorno Visual Studio, aparece su única interfaz gráfica a través de la cual se realizan todas las tareas de navegación. En el momento del inicio el aspecto que presenta es el mostrado en la figura V.1.

Las distintas zonas que aparecen son:

MAPA: Muestra la carga del plano del edificio.

RESULTADOS: Muestra los datos de salida asociados a cada función.

MENSAJES: Muestra breves instrucciones a seguir para realizar una determinad función, además de posibles mensajes de error si éstos se produjeran en algún momento.

En la parte derecha de la ventana se muestran las distintas opciones posibles, como realizar la carga del mapa, el cálculo automático del MGD (Grafo de Descripción Métrica), las opciones para realizar la planificación de una ruta y los botones para iniciar o cancelar el seguimiento de estas.

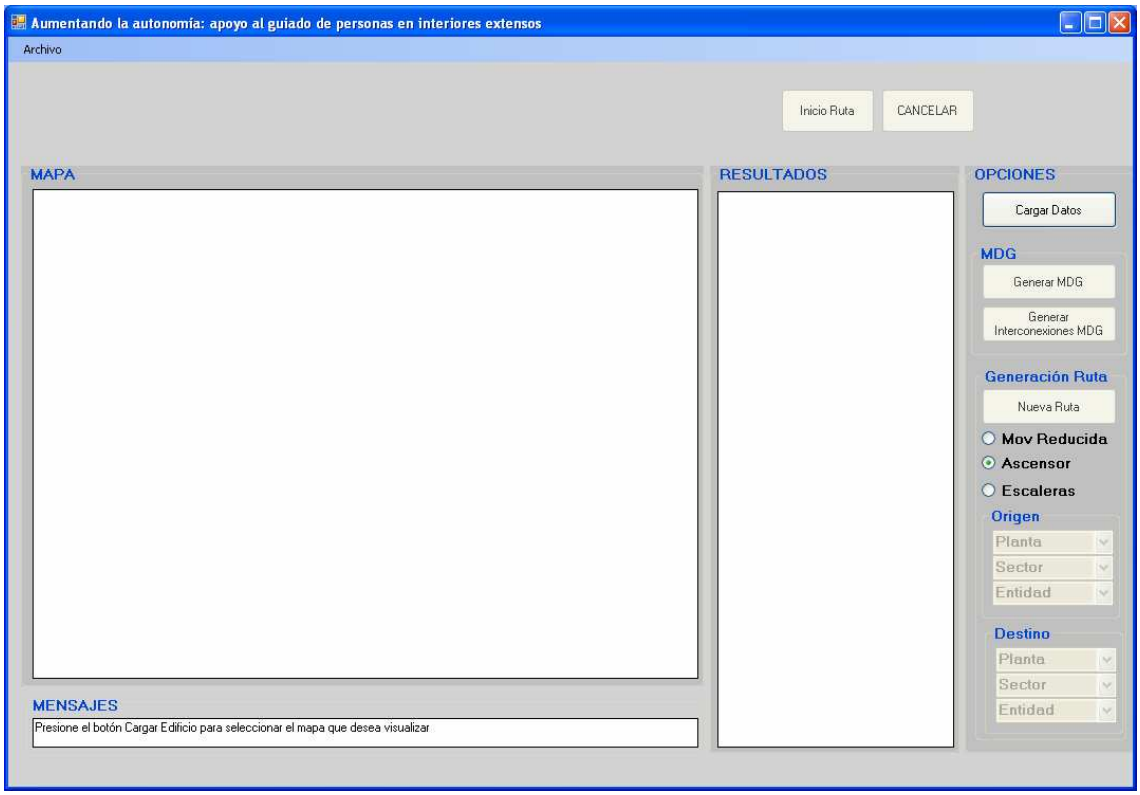


Figura V.1. Pantalla de inicio de la aplicación.

Para trabajar con esta aplicación lo primero que se debe de hacer es cargar la información del edificio por el que se desea transitar. Para ello se ha de pulsar el botón **"Cargar Datos"** y seguidamente se mostrará en el cuadro **MAPA** una imagen del mismo, apareciendo además en la parte superior de la ventana las vistas en miniatura correspondientes a cada una de las plantas del edificio por las que se puede discurrir con la silla mediante esta aplicación. También aparece un enlace a Google Maps en el que se determina su ubicación, y en el cuadro **RESULTADOS** se muestra la información básica del edificio, como su dirección y otros datos como sus coordenadas. Todo esto puede verse en la figura V.2.

Además, se activan los botones de las posibles opciones a realizar. Una vez se ha seleccionado la planta del edificio que se desea visualizar, hay tres opciones:

"Generar MDG": Realiza la representación en el mapa del Grafo de Descripción Métrica, mostrándose cada una de las rutas existentes para la planta seleccionada.

"Generar Interconexiones MDG": Muestra las interconexiones entre las distintas rutas del MDG.

"**Nueva Ruta":** Prepara la aplicación para las tareas de creación y seguimiento de rutas.

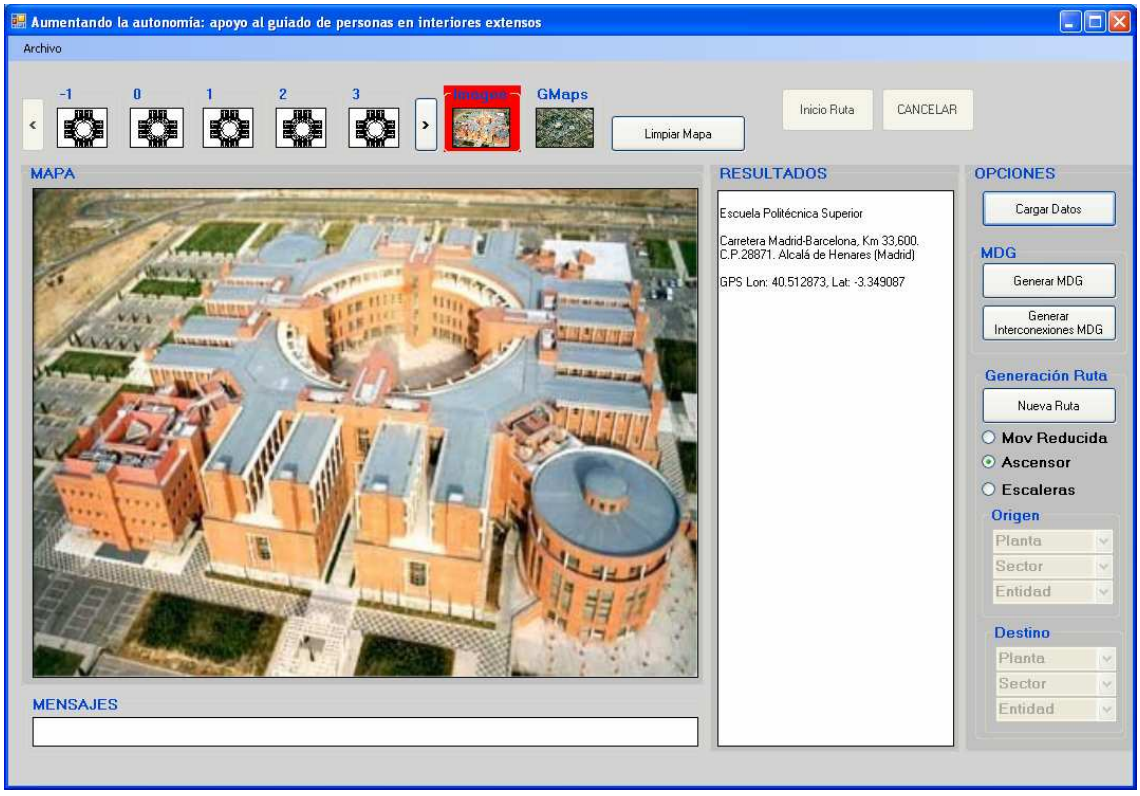


Figura V.2. Aplicación tras la carga del edificio.

2. Obtención y seguimiento de rutas.

Para conseguir que la silla se desplace por el edificio, el primer paso necesario es obtener la ruta deseada. Para ello, una vez cargado el mapa del edificio y seleccionada la planta en la que se encuentra el recinto origen, se pulsa el botón **"Nueva Ruta"**. Sobre el mapa, aparecerán representados por puntos verdes las posibles posiciones del plano que se pueden elegir como orígenes o destinos de la ruta, tal como se puede observar en la figura V.3.

Se debe marcar una de las opciones entre **Movilidad Reducida**, **Ascensor** o **Escaleras**. Según la elección, la ruta obtenida puede variar según por donde deba discurrir para adaptarse a las necesidades del usuario.

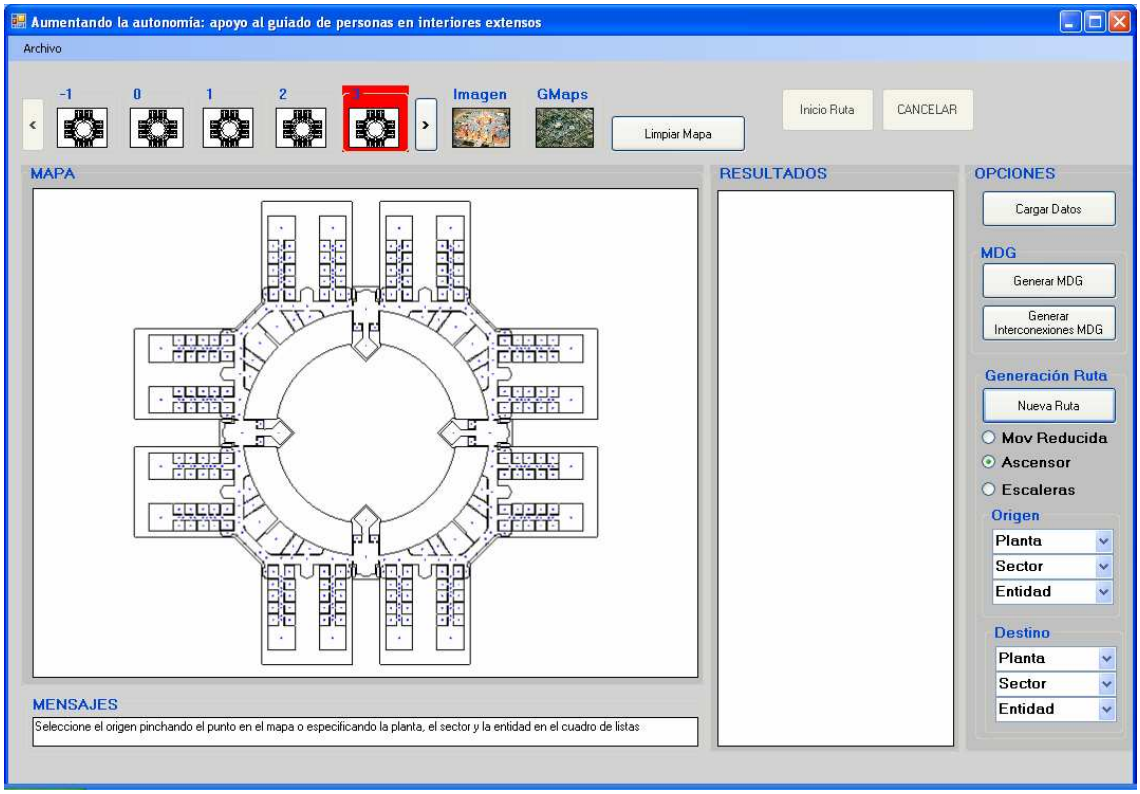


Figura V.3. Selección del origen y destino de ruta.

A continuación, se debe seleccionar el origen y destino de la ruta que se desea crear, para lo que existen dos procedimientos:

- El primero es la selección mediante las listas de desplegables en la parte inferior derecha de la ventana, especificando la planta, el sector y el nombre del edificio.

- El segundo es seleccionarlos mediante el puntero del ratón, clickeando en el plano con el botón izquierdo sobre el punto deseado. Primero se seleccionará el punto origen de la ruta y una vez aparezca en la lista de desplegables, se procede de igual forma con el punto destino de la ruta.

Con cada paso realizado, en el cuadro **MENSAJES** aparecerán las distintas instrucciones a seguir. Una vez establecidos el origen y destino de la ruta a crear, esta es dibujada en verde sobre el mapa tal como se muestra en la figura V.4, y sobre el cuadro de **RESULTADOS** aparece la información de la ruta, como la distancia a recorrer, el tiempo estimado y los puntos por donde discurre.

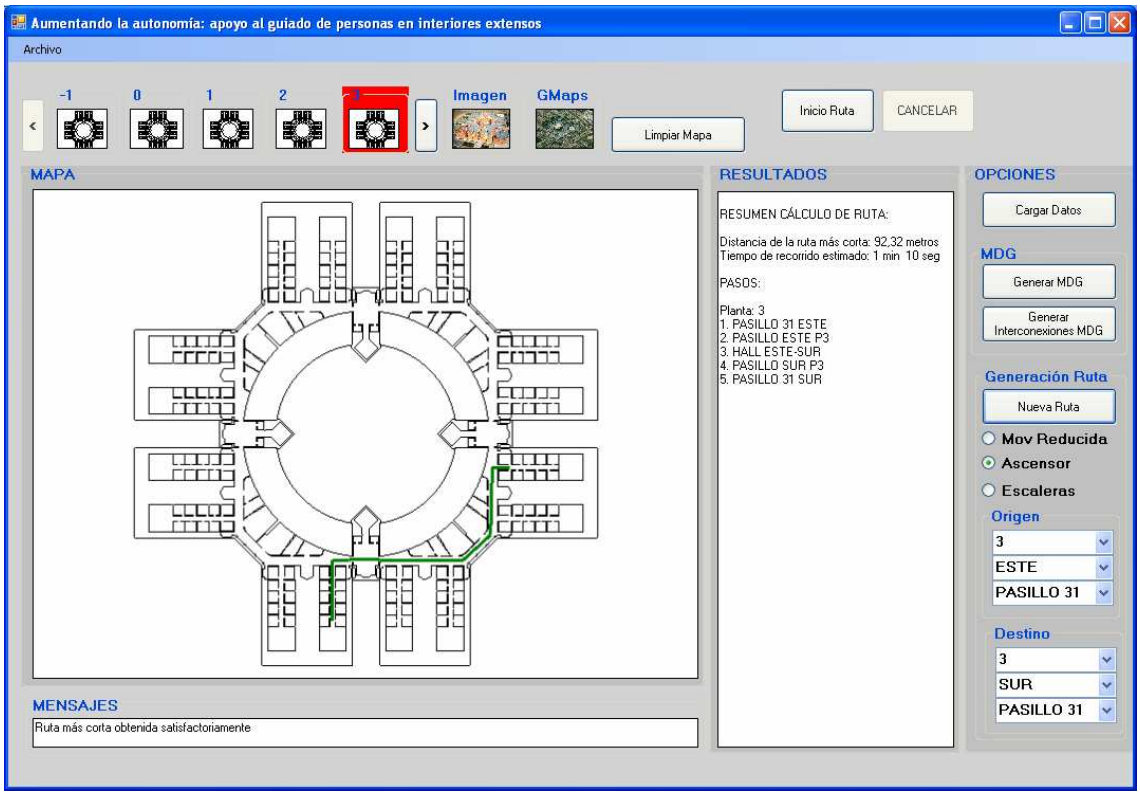


Figura V.4. Obtención de la ruta deseada.

Si se desea crear una nueva ruta, se puede limpiar el mapa con el recorrido anterior pulsando el botón **"Limpiar Mapa"** que se encuentra en la parte superior de la ventana.

Obtenida la ruta se habilita el botón **"Inicio Ruta"**, el cual se pulsará si se desea que la silla comience a seguir la ruta que se ha generado. Para ello es muy importante que la posición real de la silla, tanto coordenadas como orientación, sea lo más exacta posible al punto de origen seleccionado, pues de ello depende el correcto desplazamiento del móvil por los distintos espacios.

Al pulsar dicho botón la silla se pondrá en marcha automáticamente y se habilitará el botón **"Cancelar"**, que permite abortar el seguimiento de la ruta en cualquier momento si es pulsado.

Además, al iniciarse el recorrido de la ruta, el botón **"Inicio Ruta"** pasa a llamarse **"Pausar Ruta"**, de modo que si es pulsado en cualquier momento durante el seguimiento de una ruta, la silla disminuye su velocidad progresivamente hasta quedar totalmente parada, manteniéndose en ese estado hasta que se vuelva a pulsar ese mismo botón que tras la pausa se llamará

“Continuar”. Al hacerlo, la silla emprenderá de nuevo la marcha continuando desde el punto en que se realizó la pausa.

Según discurra la silla por el edificio, como se ve en la figura V.5, su posición será marcada en rojo sobre el mismo plano de la aplicación, y una vez llegue a su destino, se detendrá automáticamente dándose por finalizadas todas las tareas de navegación.

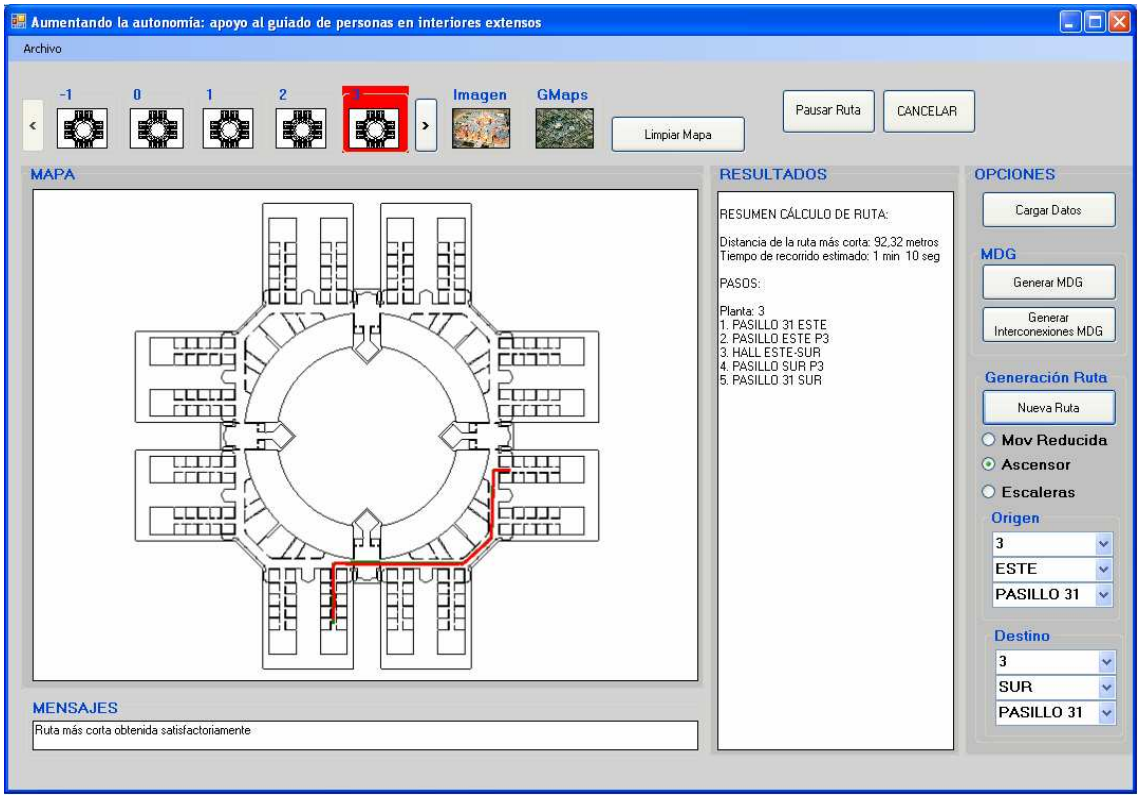


Figura V.5. Seguimiento de la ruta creada.

3. Generación de ficheros para análisis.

Obtenida la ruta deseada y tras realizarse todas las tareas previas a la puesta en marcha de la silla, la aplicación crea un fichero de texto sin formato llamado trayectoria_test.txt en el que se recopila la información con los resultados de los distintos procesos, generada durante todo el seguimiento de la ruta, es decir desde la puesta en marcha hasta la detención final de la silla.

La figura V.6 muestra un ejemplo con los posibles datos que se podría encontrar en este fichero. Este es elaborado en tiempo real según se van obteniendo los

distintos datos de los procesos que se ejecutan durante las tareas de seguimiento de las trayectorias.

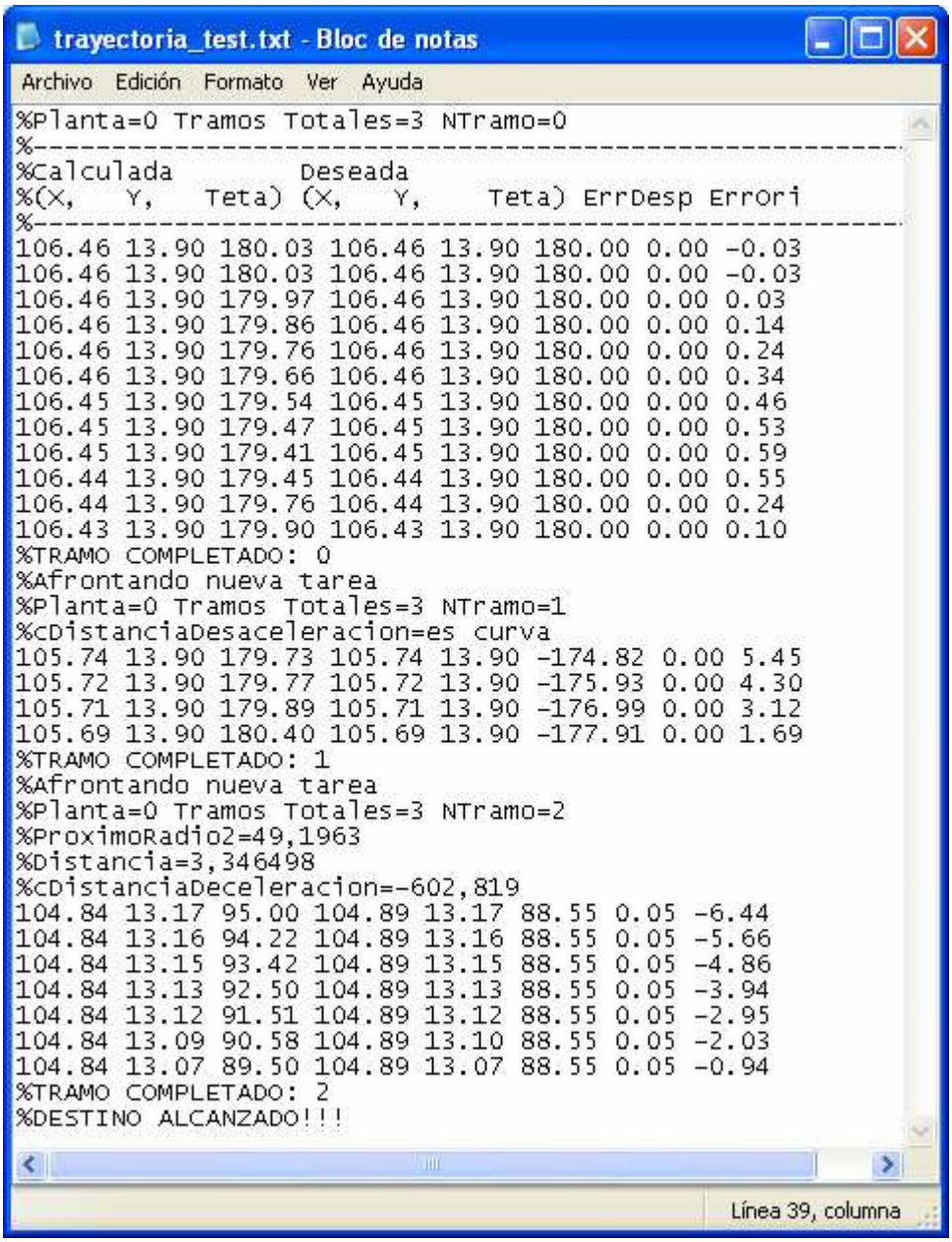


Figura V.6. Ejemplo de la información generada en el seguimiento de una ruta.

Su contenido esta dividido en bloques, uno por cada una de las trayectorias que conforman la ruta, y dentro de estos, cada línea contiene la información obtenida en cada periodo de muestreo. Los datos de cada línea son los que se exponen en la tabla V.1.

Entre cada uno de estos bloques, se almacenan datos de información referente a los procesos que se dan al producirse el cambio de la trayectoria a seguir, como

son el numero del próximo tramo, el tipo que es, su longitud o la distancia de desaceleración. Por último, al alcanzarse el destino de la ruta, el fichero será cerrado con un mensaje que lo indique.

Nº Columna	
1	Coordenada x real calculada por la silla.
2	Coordenada y real calculada por la silla.
3	Orientación real calculada por la silla.
4	Coordenada x deseada de la trayectoria.
5	Coordenada y deseada de la trayectoria.
6	Orientación deseada de la trayectoria.
7	Error de distancia.
8	Error de orientación.

Tabla V.1. Significado para las columnas del fichero trayectorias_test.txt.

Estos datos podrán ser utilizados a posteriori para representar con mayor precisión que la empleada en la aplicación, cada una de las trayectorias seguidas, así como el resto de información que se pueda obtener ellos.

En la realización de este proyecto se ha creado un strip para Matlab llamado TrazaDesplazamiento.m, con el que se representan de manera gráfica la trayectoria real de la silla junto con la que se deseaba seguir. También se obtiene los gráficos con los errores de desplazamiento y de orientación producidos a lo largo de los recorridos y la velocidad desarrollada.

Por otro lado se crean los ficheros PosicionNodos1.txt y PosicionNodos2.txt, dos ficheros también de texto sin formato como los mostrados en la figura V.7, en los que se almacena la posición de los nodos obtenidos en los ficheros route.xml y nodospuertas.xml respectivamente, descritos anteriormente en el capítulo "Generación y Control de Trayectorias". De este modo se tiene su información fácilmente portable, para poder hacerse uso de ella en cualquier aplicación o tarea de análisis, como por ejemplo para representar los nodos en las graficas de las trayectorias.

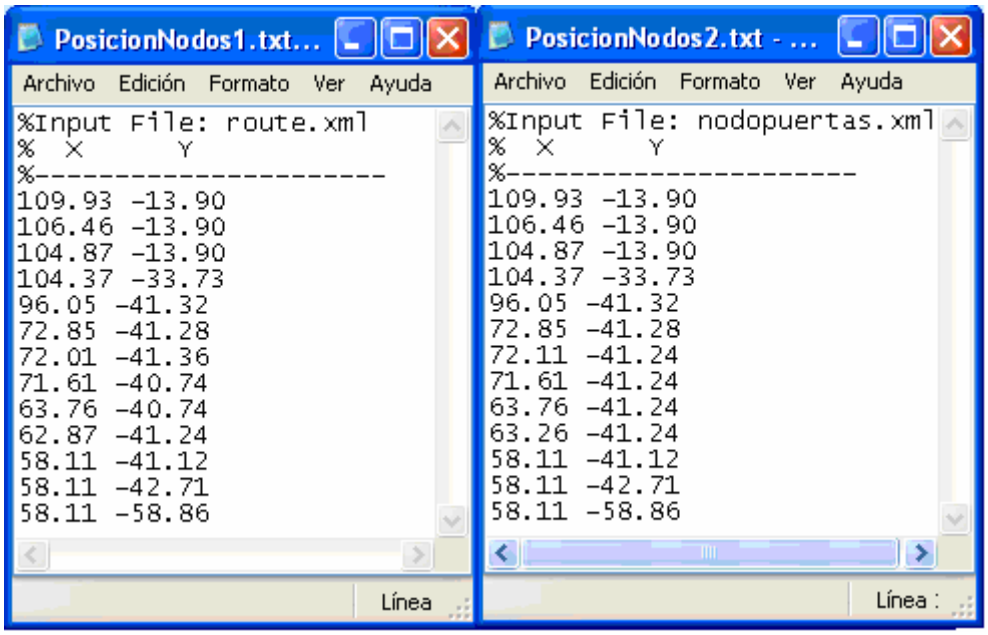


Figura V.7. Ejemplo de los ficheros PosicionNodos1.txt y PosicionNodos2.txt.

Además, debido a que se trata de procesos ejecutados en tiempo real, toda esta información guardada en estos ficheros resulta muy importante para las tareas de desarrollo y depuración, ya que no es posible parar la ejecución de las tareas de seguimiento implementadas para consultar el estado de alguna de las variables empleadas.

De este modo, una vez finalizado un recorrido, es posible analizar el comportamiento realizado por la silla comparándolo con los resultados almacenados los ficheros y poder sacar las conclusiones oportunas.

VI. BIBLIOGRAFIA

1. Proyectos fin de carrera.

[01] David Gualda Gómez.
"Servicio de Navegación para Interiores Extensos"
Escuela Politécnica UAH, Master Universitario en sistemas Electrónicos Avanzados.
2011.

[02] Marta Marrón.
"Navegación Autónoma de una Silla de Ruedas en Interiores Parcialmente Estructurados".
Escuela Politécnica UAH, Ingeniería en Electrónica.
Septiembre 2000.

[03] Juan Bellón Álvarez.
"Guiado de una silla de ruedas mediante joystick y soplido por bus CAN".
Escuela Politécnica UAH, Ingeniería Técnica Industrial en Electrónica Industrial.
Septiembre 2009.

[04] José Basterrechea García.
"Guiado semiautomático de una silla de ruedas comandada por un joystick de cabeza a través de bus CAN".
Escuela Politécnica UAH, Ingeniería Técnica Industrial en Electrónica Industrial.

Septiembre 2009.

[05] María Martínez Blasco.
"Estudio y modelado del sistema dinámico de una silla de ruedas".
Escuela Politécnica UAH, Ingeniería en Electrónica.
Julio 2003.

[06] Eduardo Sebastián Martínez.
"Guiado Semiautomático de una Silla de Ruedas".
Escuela Politécnica UAH. Ingeniería en Electrónica.
Octubre 1999.

2. Otros documentos.

[07] F. Espinosa, M. Mazo, M.E. López, J. Ureña, F.J. Rodríguez y J.J. García.
"Generación de Trayectorias y detección de obstáculos mediante splines en el guiado de robots móviles".
Información Tecnológica. Centro de información tecnológica.
Volumen 9, número 6, intervalos de paginas 125-134.
1998.
http://books.google.es/books?id=jqUlENrmQ6YC&lpg=PA128&ots=KlMAi6VMir&dq=generaci%C3%B3n%20de%20trayectorias%20con%20spline&hl=es&pg=PA125#v=onepage&q=generaci%C3%B3n%20de%20trayectorias%20con%20spline&f=false

[08] Robotep.
"User´s Manual, AX3500 Dual Channel High Power Digital Motor Controler".
www.roboteq.com
2007.