Universidad Zaragoza
1542

Trabajo de Fin de Grado

Geo-Pointing mediante unidad Pan-Tilt aplicado al movimiento y las comunicaciones de un vehículo robótico con Matlab.

Geo-Pointing using a Pan-Tilt unit applied to the movement and communications of a robotic vehicle with Matlab.

Autor:

Javier Franco Ramírez

Director:

José Luis Villarroel Salcedo

ESCUELA DE INGENIERÍA Y ARQUITECTURA

AÑO 2024

AGRADECIMIENTOS

En agradecimiento de todas las personas que me han apoyado en el trascurso de mi carrera estudiantil, entre ellos, profesores ejemplares que me han animado a seguir siempre hacia delante y orientadores que han sido capaces de guiarme por el buen camino; añadir también a mi familia, que siempre han estado allí para cualquier cosa que he podido necesitar y me han aconsejado como mejor han podido, agradezco la educación que tanto instituciones como personas cercanas a mí me han brindado, todo ello para colmar con eventos como el que hoy se describe.

Sinceramente,

Javier Franco Ramírez.

ÍNDICE

Capítulo 1: Introducción y Objetivos.

Todos somos conocedores de la inmensa utilidad de los componentes robóticos y el impacto en la vida de las personas [1] y [2]; es por ello que estamos aquí para emprender junto al Departamento de Informática e Ingeniería de Sistemas de La Universidad de Zaragoza el desarrollo de una aplicación informática cuya función principal es la localización y apuntado de un objetivo concreto en el espacio tridimensional, el cual queda englobado en un proyecto mucho más ambicioso en el que se propone realizar la comunicación entre una "base de operaciones" y un vehículo robótico móvil, este está diseñado para realizar labores de expedición que podrían llegar a descomprometer y facilitar la labor humana.

La comunicación en estos entornos es crucial, puesto que toda la información acerca del manejo del vehículo y, por el contrario, la que se recibe de este, debe ser correctamente procesada y mantenida en el tiempo a lo largo de todo el ejercicio, lo que conlleva la elaboración de una tarea en tiempo real que implica que jamás deba perderse el canal de conexión entre ambos base y vehículo, algo que podría resultaría fatal; en este caso y como primera etapa del proyecto, se propone lograrlo mediante un láser PWM.

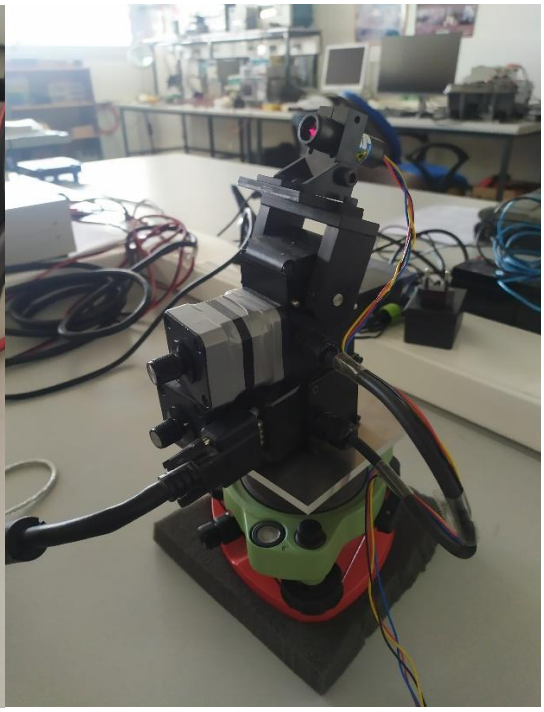
Lo que se pretende con este proyecto es el desarrollo de una aplicación que dote a una o varias unidades de dos ejes tipo *"PAN-TILT"* modelo *"PTU-D46"* de la capacidad de localizar y apuntar a cualquier objetivo en el espacio tridimensional y de seguir su trayectoria en tiempo real; a esta se le comunicarán las órdenes y consignas desde el ordenador mediante su propio controlador con las directrices expuestas en el Manual de Usuario de la máquina [3]. Para la realización de ensayos, se ha montado sobre el soporte de esta un pequeño láser PWM de poca potencia (5 mW aproximadamente) y cada componente del sistema se alimenta mediante una fuente de alimentación estándar de corriente continua.

Para corroborar su correcto funcionamiento, se ha implementado un algoritmo que registra en tiempo real la posición y orientación de dicha referencia, y para verificar la exactitud del seguimiento, también se ha propuesto un entorno simulado.

Además de lo expuesto, para mantener a la unidad fija en su posición inicial y para mantener su eje vertical perpendicular al suelo en todo momento, se ancla mecánicamente a una base nivelante, y damos uso a un trípode para evitar vibraciones indeseables que pueden comprometer la precisión del láser, además de elevar a la unidad a la altura en la que podremos proceder con los experimentos prácticos.

En la Figura 1 se muestran varias fotografías que ilustran los diferentes componentes del sistema completo.

Controlador de la unidad.



Unidad "PAN-TILT" PTU-D46

Fuente de alimentación.

Fig. 1 Componentes del sistema completo.

El contenido de la memoria se ha dividido en varios capítulos, cada uno de ellos se corresponde con las distintas fases e hitos del proyecto, y están ordenados de acuerdo con la continuidad temporal en la que fueron elaborados.

La parte fundamental del trabajo está ubicada en los capítulos 2: *"Realización del seguimiento"* y 3: *"Esquema de funcionamiento".* Después se llevan a cabo los ensayos y experimentos de simulación en el capítulo 4: *"Realización de los ensayos"* y para finalizar, en el capítulo 5: "Posibles mejoras y modificaciones", se exponen detalladamente aquellas pautas, ideas o posibles modificaciones que podrían mejorar la aplicación, pero que no se ha podido llegar a implementar.

Capítulo 2: Realización del seguimiento.

El primer paso para llevar a cabo el sistema de seguimiento es conocer la geometría de la unidad Pan-Tilt, de tal manera que, conociendo la disposición de sus ejes en el espacio, podamos localizarlos en un sistema absoluto de coordenadas con el que empezar a trabajar.

2.1 Geometría de la unidad Pan-Tilt.

La referencia absoluta se define como el punto ortogonal correspondiente a una de las cuatro esquinas en la base de la unidad, esto es, el triedro principal sobre el que se construirán todas las demás localizaciones y orientaciones en el espacio.

En la parte izquierda de la figura 2 se describe el triedro que se corresponde a la referencia absoluta (ABS), y en la parte derecha, el movimiento de ambos ejes de rotación que posee la unidad (θ_{PAN} y θ_{TILT}).

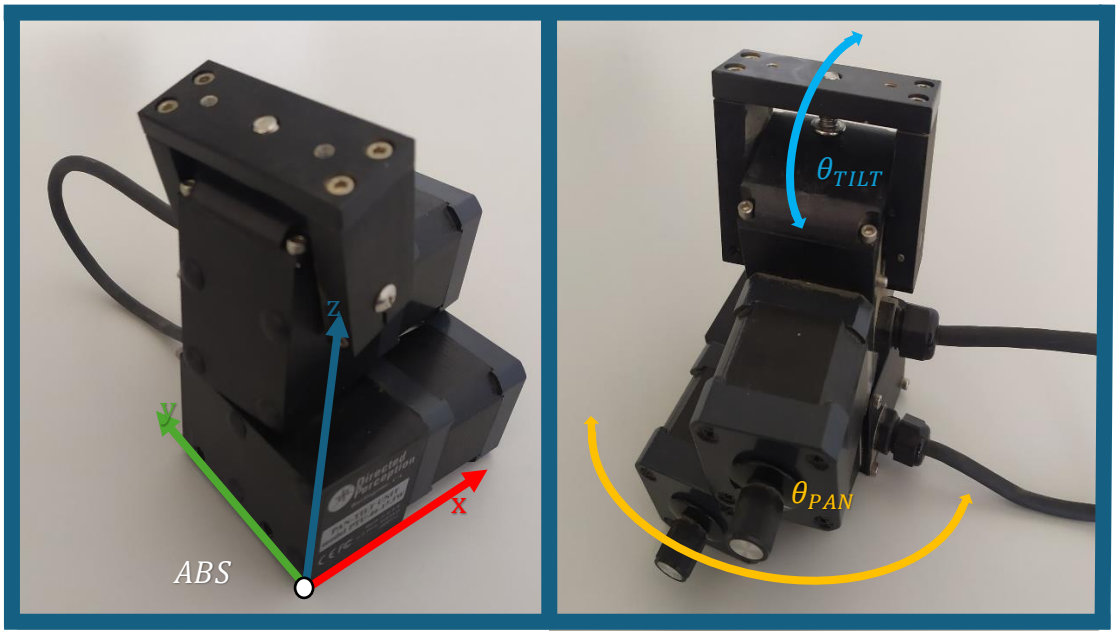


Fig. 2 Representación visual referencia absoluta y ejes de la unidad

Todo lo que necesitamos conocer para ubicar el eje horizontal o θ_{PAN}, son las distancias X_{PAN} e Y_{PAN} de dicho eje con respecto a la referencia absoluta, y en el caso del eje θ_{TILT}, ya que está montado sobre el eje horizontal θ_{PAN}, nos basta con la distancia vertical Z_{TILT}, que se corresponde con la distancia desde este segundo eje hasta el suelo, tal y como se aprecia en la figura 3.

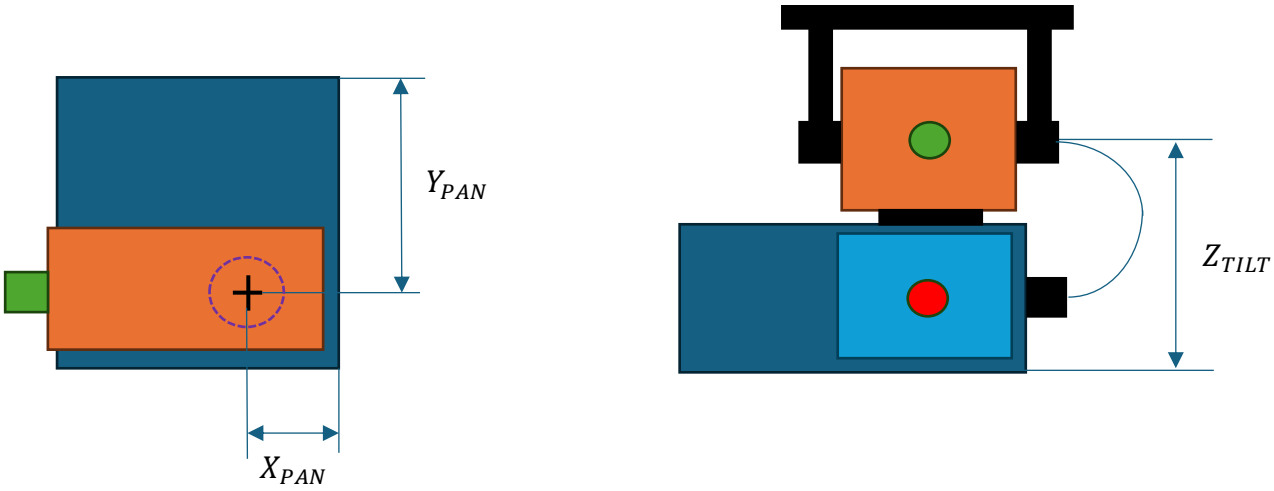


Fig. 3 Posiciones relativas de los ejes respecto al orígen de coordenadas

En la tabla 1 se detallan el valor numérico de las cotas para las referencias:

X_{PAN}	21.59 mm
Y_{PAN}	45.65 mm
Z_{TILT}	91.694 mm

Tabla 1 cotas de referencia para los dos ejes respecto al triedro principal

A la altura Z_{TILT} hay que añadirle la altura de la base nivelante (*86 mm*) además de la del trípode, en el caso de que sean usados durante los experimentos prácticos.

2.2 Cálculo de las consignas para los ejes.

Una vez que hemos localizado nuestros dos ejes en el espacio, lo siguiente es obtener una consigna para cada uno de ellos y que entre ambos el láser montado apunte a un lugar designado, para el caso inicial, este punto objetivo es estático.

- *Eje horizontal (θ_{PAN}):* para encontrar el ángulo necesario para orientar horizontalmente la máquina, tenemos que encontrar las coordenadas relativas del punto designado con respecto a la referencia del eje PAN:

$$X_{REL} = X_{DESTINO} - X_{PAN}$$
$$Y_{REL} = Y_{DESTINO} - Y_{PAN}$$

Y con estas dos distancias resolvemos el problema geométrico:

$$\theta_{PAN} = atan\left(\frac{Y_{REL}}{X_{REL}}\right) - \Delta\theta_{PAN,0}$$

Siendo $\Delta\theta_{PAN,0}$ el offset inicial del ángulo θ_{PAN}, que para la unidad sobre la que hemos trabajado es 0 radianes.

El ángulo resultante de esta operación será aquel que impondremos a la unidad mediante comunicación serie para alcanzar el punto orientando el ángulo θ_{PAN}. En la figura 4 se muestra la resolución del problema de manera visual.

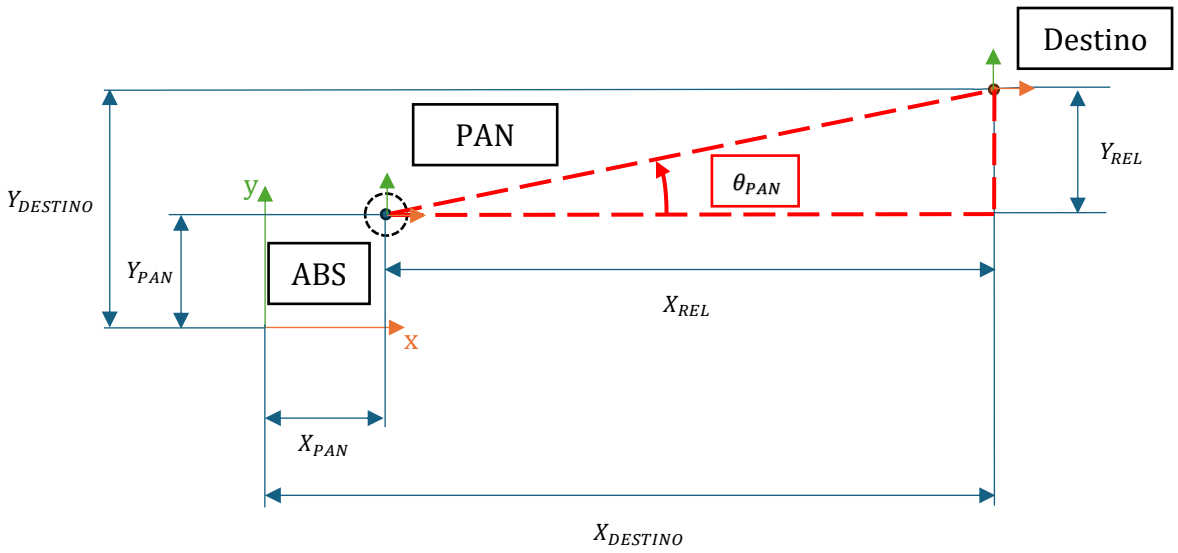


Fig. 4 Resolución geométrica ángulo θ_{PAN}

- *Eje vertical (θ_{TILT}):* una vez que hemos logrado alinear satisfactoriamente el eje de rotación horizontal, el problema de tres dimensiones se convierte en uno de dos, puesto que la profundidad ya ha sido resuelta, ahora solo trabajaremos con el nuevo eje creado por la rotación del eje θ_{PAN} , al que llamaremos "$\hat{X}$".
  La altura del punto designado Z_{ABS}, es la suma de tres componentes: (1) La altura de offset del eje vertical Z_{TILT}, (2) la proyección vertical del soporte Lt_V, y (3) la altura vertical generada por el ángulo que tenemos que calcular θ_{TILT}. Para ayudarnos con los cálculos, definimos dos distancias horizontales, $\hat{X}_{REL}$, $\hat{X}_{ABS}$ y Lt_H a lo largo del nuevo eje "$\hat{X}$" estas cotas se visualizan en el espacio tal y como se muestra en la figura 5.

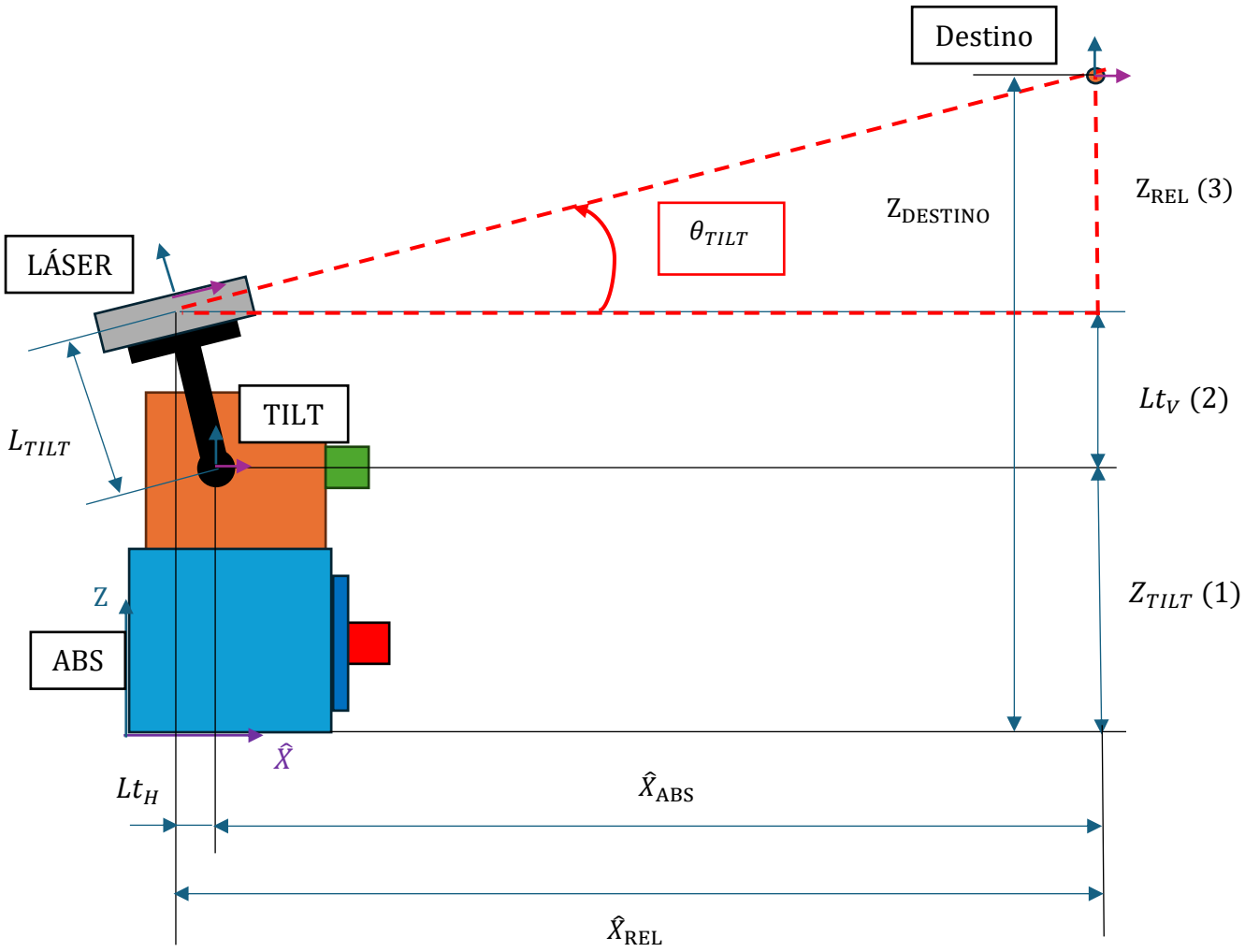


Fig. 5 Resolución geométrica ángulo θ_{TILT}

Debido a que el ángulo que queremos hallar queda implícito en alguna de las cotas que se exponen en la Fig. 5, este no se puede inferir directamente; la manera que hemos elegido para resolver el problema geométrico es mediante un método iterativo, por lo tanto, necesitamos al menos dos maneras de obtener la solución, para así poder contrastar los valores de ambos cálculos y tratar de compensar la diferencia, hasta llegar a converger a una solución final que resulte compatible para sendos cálculos. Los dos puntos de vista desde los que vamos a enfocar la resolución son: la resolución vertical (1) y la horizontal (2).

-(1): La suma de las tres componentes verticales descritas en la Fig. 5 debe igualar a la distancia absoluta del objetivo, reordenando:

$$Lt_V = L_{TILT} * cos(\theta_{TILT})$$

$$Z_{REL} = Z_{ABS} - Lt_V - Z_{TILT}$$

Siendo L_{TILT} la longitud del soporte ($67.116\ mm$).

-(2): El valor $\hat{X}_{ABS}$ es la distancia horizontal entre la referencia DESTINO y la referencia TILT a lo largo de eje $\hat{X}$:

$$\hat{X}_{ABS} = \frac{\hat{X}_{REL}}{\cos(\theta_{PAN})}$$

-$\hat{X}_{REL}$ es la distancia horizontal entre las referencias DESTINO y LÁSER:

$$Lt_H = L_{TILT} * sen(\theta_{TILT})$$

$$\hat{X}_{REL} = \hat{X}_{ABS} + Lt_H$$

-Y con esto podemos volver a resolver Z_{REL}:

$$Z_{REL} = \hat{X}_{REL} * \sin(\theta_{TILT})$$

El cálculo iterativo se encierra en un bucle *"while"* tal y como sigue:

$$while\ (\theta_{TILT} - \theta_{TILT,PREV} > res)$$

Siendo res la resolución angular que para ambos ejes es idéntica ($0.8972\ mRad$).

El ángulo θ_{TILT} resultante tras la convergencia de este método será aquel que impondremos a la unidad mediante comunicación serie para alcanzar el punto orientando el ángulo vertical.

2.3 Método para el seguimiento de un punto cinemático

Una vez que hemos resuelto el problema geométrico del apuntado, con lo que se supone que nuestro láser se orienta con respecto a un punto en el espacio tridimensional, nos centramos en cómo obtener un algoritmo que sea capaz de predecir la nueva posición y orientación de este mismo punto en función de su movimiento, entonces nos preguntamos ¿De qué manera se puede desplazar y/o orientar un punto en el espacio?

Existen bajo nuestro punto de vista dos maneras de cambiar la "pose" en el espacio para un objeto que se mueve "pegado" al suelo, como es el caso de nuestro vehículo robótico:

- Una de ellas es el desplazamiento, en el cual, siempre interviene una cantidad de movimiento. Existen dos casos, en uno de ellos el desplazamiento siempre es hacia delante, es decir, sin cambiar su orientación, sin embargo, en el segundo caso esto último se puede modificar si se altera la orientación con respecto al plano sobre el que sucede el movimiento, y por lo tanto, ya no observamos un movimiento rectilíneo; de aquí obtenemos dos parámetros de movimiento, siendo uno la velocidad lineal y el otro la velocidad angular, ambos se combinan para dar lugar a la variación de posición y orientación del punto respecto al plano que gobierna el desplazamiento.
- En la figura 6 se ejemplifica un caso de movimiento con cambio de orientación sobre un plano cualquiera delimitado por los ejes "$\hat{X}\hat{Y}$" desde la referencia relativa REL.

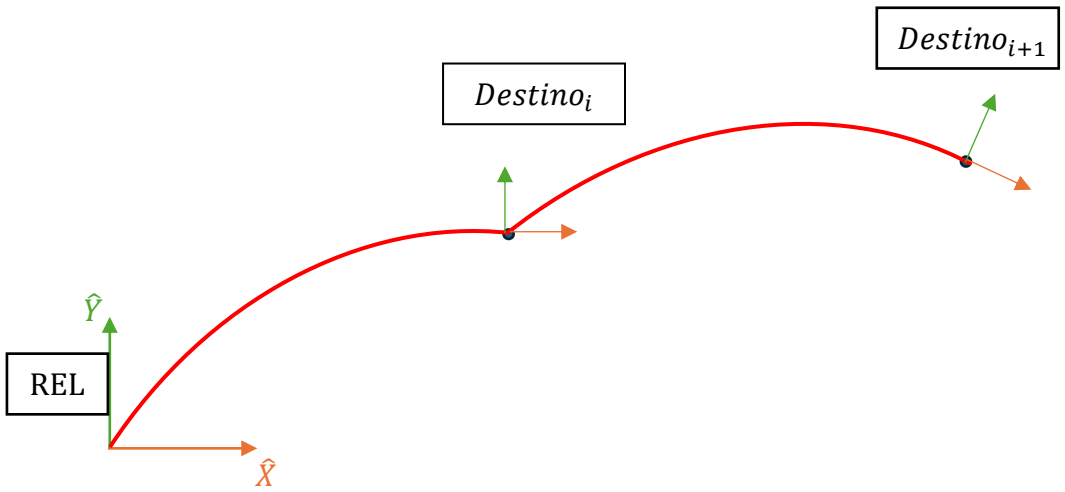


Fig. 6 Movimiento sobre el eje "$\hat{X}\hat{Y}$" con cambio de orientación

- La otra manera supone cambiar intrínsecamente la orientación del plano sobre el que sucede el movimiento con respecto a la referencia absoluta; esto da lugar a nuestro tercer parámetro del movimiento, la inclinación, lo que supone que el propio plano que gobierna el desplazamiento va a cambiar su orientación.
  A continuación, se incluye en la figura 7 un ejemplo de movimiento medido desde la referencia absoluta, pero en este caso alterando el plano sobre el que se lleva a cabo.

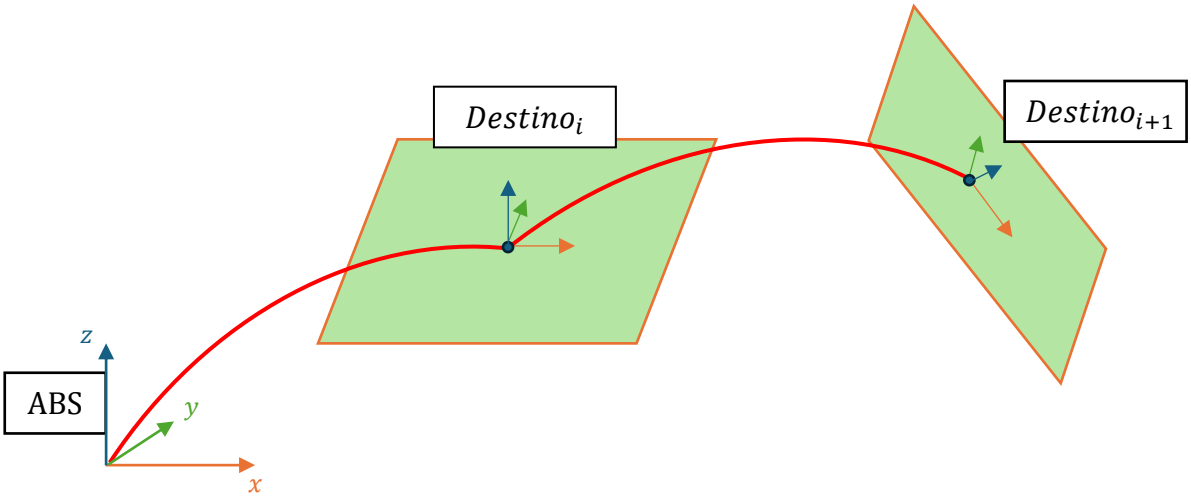


Fig. 7 Cambio en la inclinación donde se describe movimiento planar.

La combinación de los tres parámetros describe el movimiento realizable por un objeto rígido que se desplaza sin despegarse del suelo.

- *Implementación algorítmica:*

Como anteriormente mencionamos, hay dos tipos de desplazamiento, uno de tipo rectilíneo cuando solamente existe velocidad lineal, y para el segundo tipo en el que se incluye la velocidad angular, esto determina una trayectoria tipo arco de circunferencia, tal y como se aprecia en la figura 8, donde representamos ambos tipos de movimiento sobre un plano cualquiera "$\hat{X}\hat{Y}$" desde la referencia relativa REL, siendo v la velocidad lineal y ω la velocidad angular.

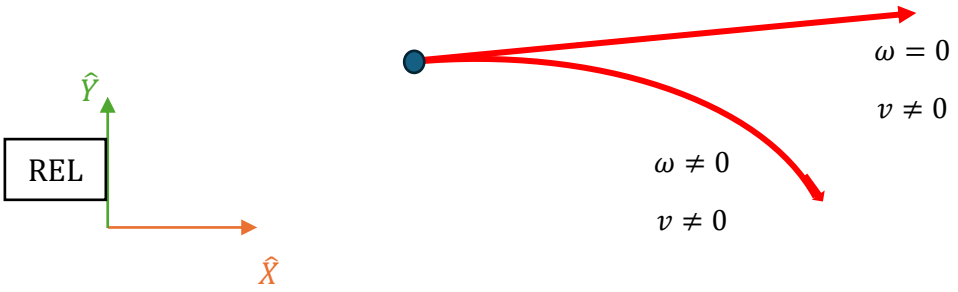


Fig. 8 Diferentes tipos de movimiento sobre un plano cualquiera.

Como se puede observar, el movimiento no puede existir en ausencia de velocidad lineal, pero sí en ausencia de velocidad angular.

El cálculo del desplazamiento sobre un plano cualquiera "$\hat{X}\hat{Y}$" se obtiene de la mediante odometría [3] de la siguiente manera:

- En primer lugar, se crean las variables con las que vamos a operar (Tabla 2) y se ilustra con un ejemplo visual (figura 9).

φ	ángulo girado a lo largo de toda la trayectoria hasta la iteración i.
$d\varphi$	ángulo girado durante el transcurso de la iteración i.
ds	longitud total recorrida durante el transcurso de la iteración i.
dx	Componente $\hat{X}$ de ds.
dy	Componente $\hat{Y}$ de ds.
$T_{CONTROL}$	Periodo de ejecución para cada iteración

Tabla 2 Variables de estudio en un movimiento plano.

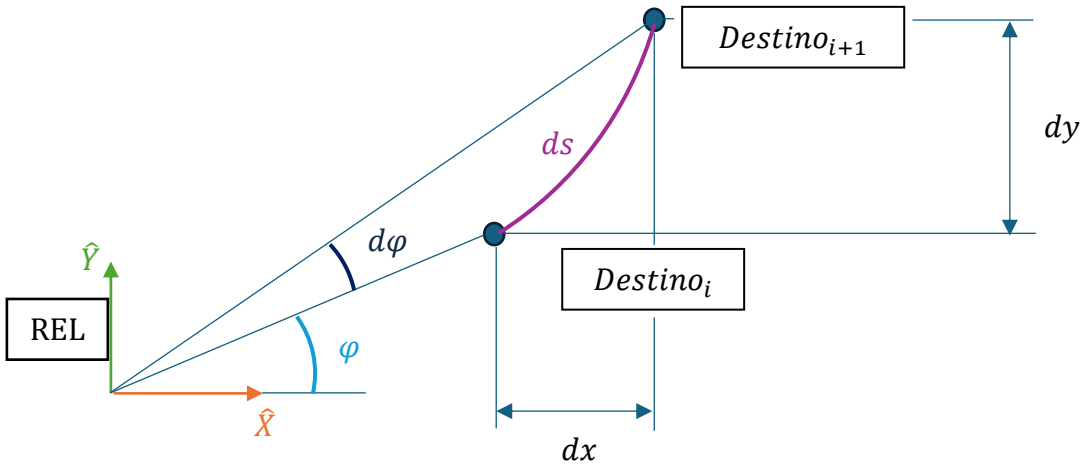


Fig. 9 Ejemplo de movimiento plano.

- Obtenemos $d\varphi$ en el caso de que exista:

$$d\varphi = \omega * T_{CONTROL}$$

- Después se calcula ds en función de si existe velocidad angular ω:

$$\text{Si } \omega = 0 \qquad\qquad \text{Si } \omega \neq 0$$

$$ds = v * T_{CONTROL} \; ; \qquad ds = \frac{v}{\omega} * d\varphi$$

- Y finalmente se descompone ds en las dos componentes horizontales que usaremos en el apartado siguiente, se usa el promedio del ángulo girado $\frac{d\varphi}{2}$ para simplificar el cálculo:

$$dx = ds * \cos\left(\varphi + \frac{d\varphi}{2}\right) \quad ; \quad dy = ds * \sin\left(\varphi + \frac{d\varphi}{2}\right)$$

Este método nos brinda como resultado el desplazamiento del punto sobre el plano. El último paso es componerlo con la inclinación, es decir, el cambio de orientación del plano que consideramos como el "suelo" sobre el que se mueve el vehículo.

Por suerte para nosotros, el robot móvil está dotado con tres giróscopos, uno por cada eje de rotación, los cuales nos aportan la información requerida en formato de "ángulos de Euler", operando con ellos directamente mediante una función del lenguaje MATLAB, (eul2tform).

$$\Delta Destino = [dx\ dy\ 0] * \text{eul2tform(euler)}.$$

Donde $\Delta Destino$ es la variación de posición absoluta de la referencia del vehículo entre dos ejecuciones consecutivas, y Euler es el vector de los tres ángulos de giro de dicha referencia con respecto al origen.

Esto es de extrema importancia, ya que así logramos predecir la posición del vehículo robótico actualizándola constantemente, llevando a cabo un seguimiento preciso de la "pose" actual del objetivo desde el punto de vista de la unidad.

Método para el seguimiento con una trayectoria rectilínea.

Nuestra intención es mantener las comunicaciones en todo momento, de ahí la necesidad de realizar el seguimiento de tal manera que la trayectoria del objetivo y la que describe nuestro puntero láser se aproximen lo máximo posible. Para ello, proponemos que la trayectoria de nuestro puntero láser sea rectilínea por dos motivos, en primer lugar, no podemos modificar la velocidad ni aceleración de los ejes mientras estos están en movimiento, pues esto conlleva demasiado tiempo de procesamiento y segundo, debido a que el tiempo de muestreo y la velocidad que puede alcanzar el vehículo son suficientemente bajos, se pueden interpolar los destinos de dos ejecuciones consecutivas mediante una recta y obtener una buena precisión.

Para ilustrar los cálculos, se expone en la figura 10 un ejemplo común que representa el desplazamiento desde la perspectiva de la unidad, cuando el objetivo

cambia desde la referencia $Destino_i$ hasta la referencia $Destino_{i+1}$ entre dos ejecuciones consecutivas.

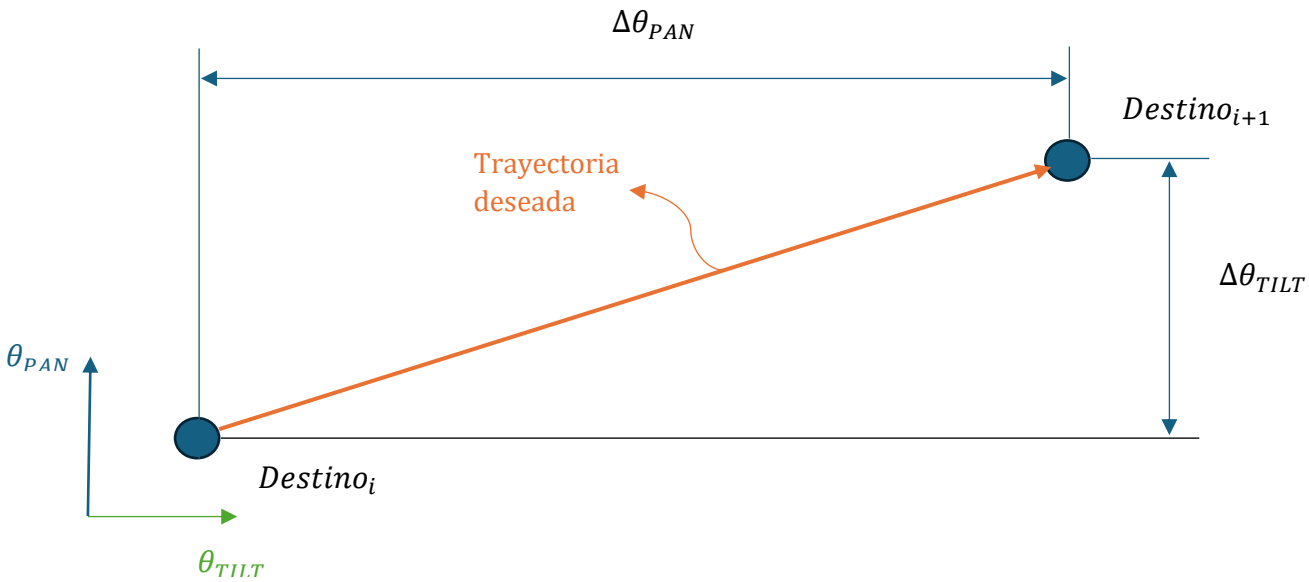


Fig. 10 Ejemplo de trayectoria rectilínea entre dos destinos consecutivos.

La solución para llevarlo a cabo se realiza relacionando la variación en la posición angular que deben recorrer ambos ejes $\Delta\theta_{PAN}$ y $\Delta\theta_{TILT}$. Adaptando las velocidades para que estos alcancen su destino al mismo tiempo, o lo que es lo mismo, que el recorrido lo realicen de manera sincronizada, para ello aplicamos una sencilla regla de tres, donde relacionamos la variación entre las posiciones angulares de destino y las velocidades de consigna entre ambos ejes:

$$\frac{\Delta\theta_{PAN}}{V_{PAN}} = \frac{\Delta\theta_{TILT}}{V_{TILT}}$$

Donde V_{PAN} y V_{TILT} son las velocidades angulares con las que dichos movimientos se llevan a cabo. Dicho esto, lo único que nos queda por definir es el tiempo que se va a invertir en el movimiento y que es idéntico para ambos ejes tal y como se puede deducir de la relación anterior.

Como especificación del proyecto, la comunicación con el vehículo se realiza periódicamente cada 100 ms, con la intención de dejar margen suficiente para otras tareas que se deban llevar a cabo aparte del posicionamiento (envío y procesamiento de información entre la base y el vehículo, por ejemplo). Este tiempo debe ser acotado y reducido considerablemente, ya que esto conlleva gran parte del periodo de control, y no podemos permitir que los ejes no hayan concluido su

movimiento antes de que comience la siguiente iteración, pues esto supondría graves retrasos.

Este apartado queda inconcluso, ya que, sin experimentación previa (*capítulo 4: "Realización de los primeros ensayos"*) no se puede determinar el tiempo que quedará disponible para realizar el movimiento.

Con estos parámetros, se define el movimiento, y se calcularán las diferentes consignas para los ejes en cada iteración del programa.

Capítulo 3. Esquema de funcionamiento.

La aplicación se va a desarrollar en el entorno de MATLAB [4], debido a la sencillez en el manejo de dicha herramienta y al vasto repertorio de funciones y algoritmos que simplifican los cálculos y facilitan la comprensión del programa.

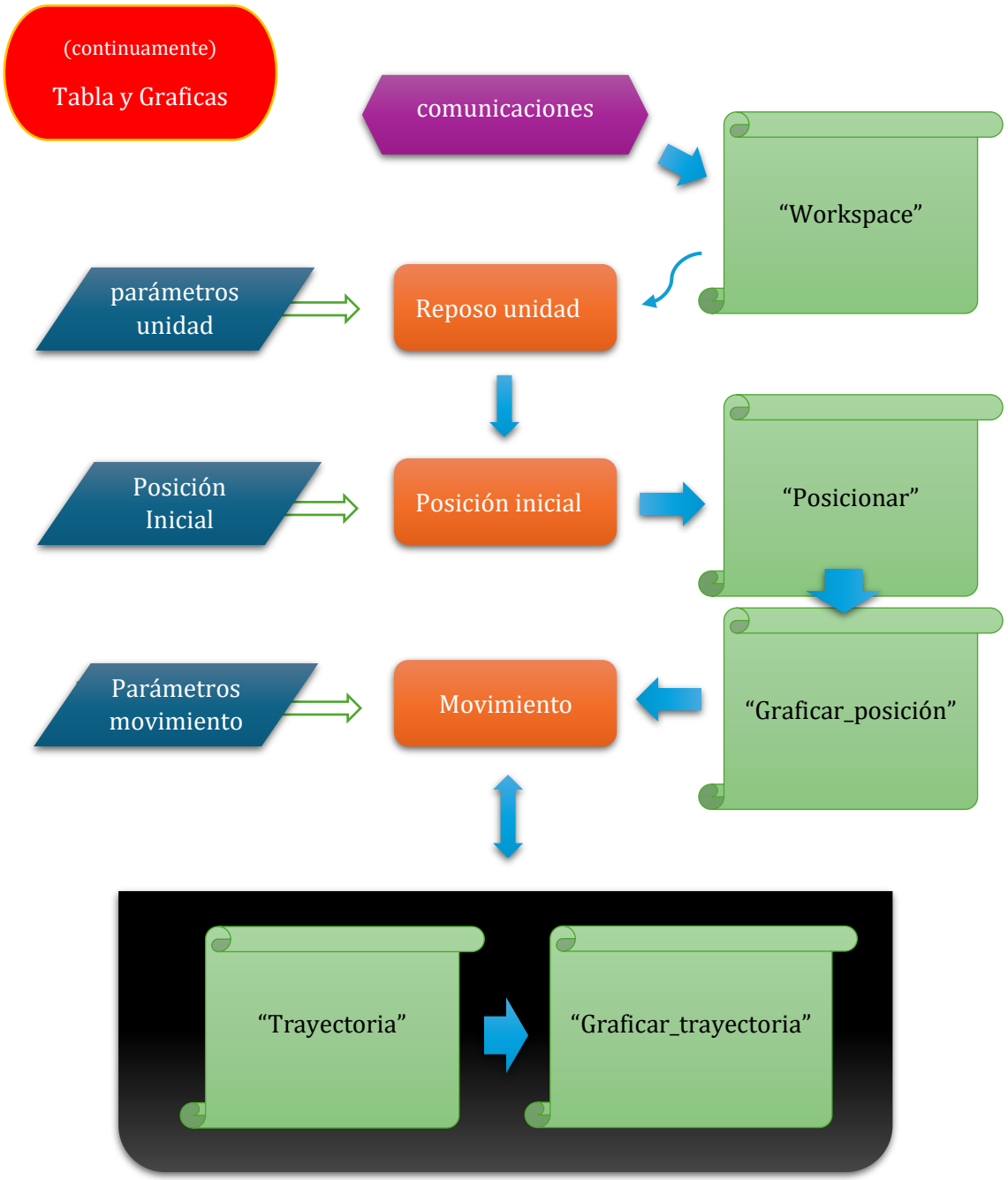
Después de plantear la problemática y la resolución teórica que hemos alcanzado, ha llegado el momento de definir un conjunto de procedimientos que implementan los diseños realizados en el estudio teórico.

Para ello, se construyen varios subprogramas que representan las diferentes vertientes del flujo del programa y que aportan continuidad a todo el proceso. Cada uno de ellos está diseñado para realizar una tarea específica dentro de todo el proceso.

A continuación, se van a nombran y desglosar uno por uno los subprogramas que se han desarrollado:

3.1 “Referencias”

El programa principal, encargado de imponer las consignas que servirán como entradas de los algoritmos de cálculo, el cual recibe como entradas los argumentos de la pose actual del vehículo y sus parámetros de movimiento, también es el encargado de realizar las llamadas a los subprogramas correspondientes para llevar a cabo los procedimientos pertinentes, el proceso que se desarrolla es el siguiente:



Donde:

- morado = inicio del programa.
- naranja = procedimiento.
- azul = datos.
- verde = script.
- rojo = procedimiento alternativo.
- negro = terminación programa

Fig. 11 Esquema de funcionamiento del subprograma "Referencias".

En primer lugar, se definen el canal de comunicaciones entre el equipo que va a ejecutar el programa y la unidad, y todos los parámetros necesarios que dan pie al comienzo de los cálculos, para ello usamos un nuevo script llamado “Workspace”.

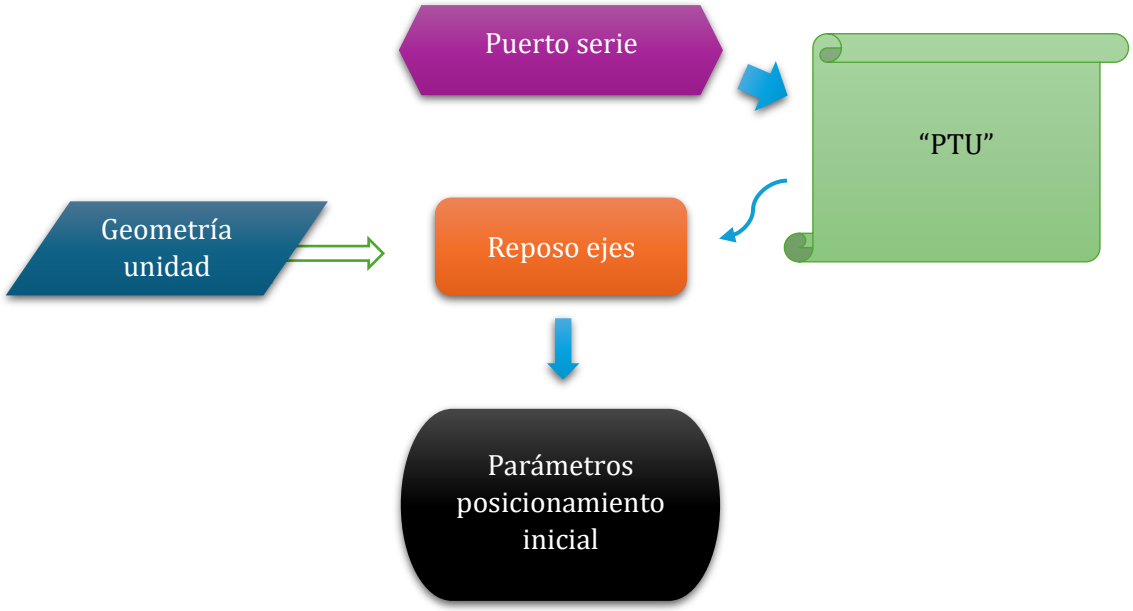
A continuación, posicionamos a la unidad en el estado de reposo, es decir, con ambos ejes en la posición angular cero. La ejecución del programa se detiene hasta que el movimiento haya finalizado.

Después, nos disponemos a localizar el punto inicial del vehículo robótico mediante el posicionamiento de los ejes haciendo una llamada al subprograma *“posicionar”;* que requiere como valores de entrada las coordenadas absolutas del objetivo respecto del origen.

En el momento de cerciorarnos que el paso anterior ha sido llevado a cabo con éxito, damos luz verde a la segunda parte del programa, que consiste en generar destinos para ambos ejes periódicamente discretizando la trayectoria que va a seguir el vehículo robótico en todo momento, e interpolando cada uno de estos puntos mediante una línea recta con la intención de realizar el ajuste más fino posible. Para ello, definimos un tercer script al que denominaremos “trayectoria” y cuyas entradas serán los tres parámetros del movimiento del vehículo.

3.2 “Workspace”

La tarea de este script es abrir el canal de comunicaciones y mandar a la unidad al estado de reposo, además de definir algunos parámetros y constantes iniciales previos a los cálculos. El procedimiento llevado a cabo se muestra a continuación:



Donde:

- morado = inicio del programa.
- naranja = procedimiento.
- azul = datos.
- verde = script.
- negro = terminación programa

Fig. 12 Esquema de funcionamiento del subprograma “Workspace”.

En primer lugar, se define el puerto serie RS-232 [5] mediante el cual vamos a comunicar bidireccionalmente el equipo donde se ejecutará el programa y la unidad; para lo cual, se define un objeto llamado “fso_ptu” que va a representar el estado de la unidad. Esto nos permitirá modificar y obtener los diferentes valores y consignas mediante la construcción de funciones que servirán como intérpretes entre ordenador y unidad; todas estas funciones y valores se detallan en el subprograma “PTU”.

La unidad por defecto trabaja a una velocidad de envío de 9600 bits por segundo. Sin embargo, este valor se puede superar para así agilizar la tarea de envío y recepción de paquetes de datos, de tal manera que configuramos el canal al valor

máximo permitido: 38400 baudios (4 veces más rápido) y llamamos a “PTU” para inicializar la entidad que representa a la máquina.

- InicializarObjetoPTU(fso_ptu)
- CrearPuertoSerie()

Una vez configurado el canal de comunicaciones, asignamos valores a las cotas que se corresponden con las distancias de los dos ejes operativos de la unidad con respecto al origen de coordenadas, de la misma manera que se ha descrito en el apartado 2.1: *“Geometría Unidad PAN-TILT”.* (Nota, una de las unidades tiene una orientación inicial distinta de cero para el eje PAN por defecto).

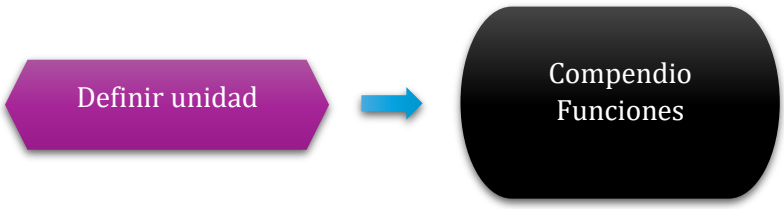
A continuación, asignamos valores de posición, velocidad y aceleración al bloque de la unidad para llevar a cabo el posicionamiento inicial y mandamos a ambos ejes a la posición de reposo, 0 radianes cada uno.

- MandarVelocidades(PAN,TILT, máximas permitidas)
- MandarAceleraciones(PAN,TILT, máximas permitidas)
- MandarPosición(PAN,0)
- MandarPosición(TILT,0)

Para finalizar se inicializan algunas variables que dan pie a los cálculos.

3.3 “PTU”

Encargado de definir la entidad que va a representar a la unidad desde nuestro programa y procesar las funciones para poder interactuar con esta. Tal y como se muestra en el esquema de la figura 13.



Donde:

- morado = inicio del programa.
- negro = terminación programa

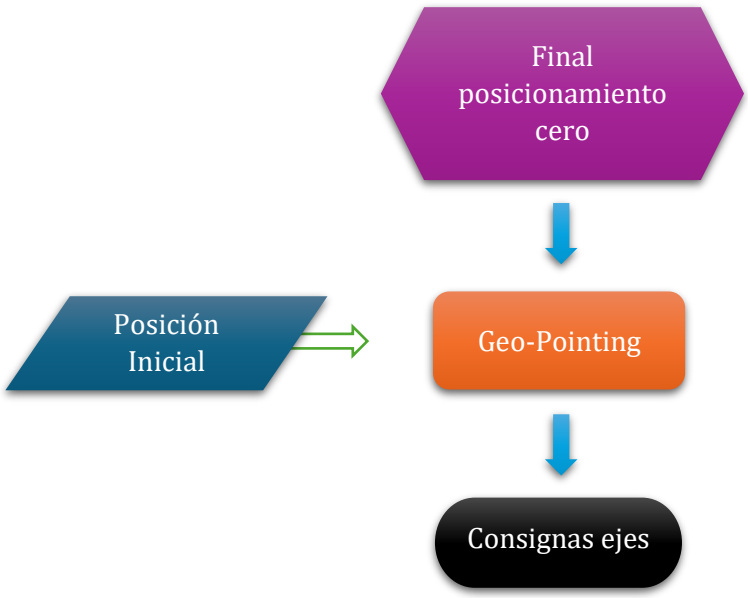
Fig. 13 Esquema de funcionamiento del subprograma “PTU”.

El objeto “fso_ptu” contempla los valores por defecto de algunos parámetros y restricciones físicas y dinámicas del movimiento de la unidad. Por ejemplo, el ángulo máximo que se puede cubrir o la aceleración máxima para cada uno de los ejes.

En la segunda parte del script, se desarrolla un listado de funciones independientes. Cada una de ellas tendrá la misión de enviar una orden a la unidad. Existen dos tipos de funciones, las del primer tipo serán ejecutadas (si la acción impuesta está dentro de los límites establecidos, de lo contrario devuelve un error) o con las que se nos responde por el valor que le habíamos preguntado, los detalles acerca de los comandos que genera cada función se pueden apreciar en el “Manual del Programador”.

3.4 “Posicionar”

Su misión es orientar los ejes hacia la referencia inicial del vehículo, antes de que este comience el movimiento. El esquema de la tarea se muestra en la figura 14.



Donde:

- morado = inicio del programa.
- naranja = procedimiento.
- azul = datos.
- negro = terminación programa

Fig. 14 Esquema de funcionamiento del subprograma “Posicionar”.

Una vez que nos hemos asegurado de que la unidad mantiene orientados ambos ejes en la posición cero (estado de reposo), podemos llevar a cabo el posicionamiento inicial con las coordenadas absolutas de la referencia del vehículo. Para ello, calculamos las consignas de los ángulos mediante los procedimientos explicados en el capítulo 2.2 *“Cálculo de las consignas para los ejes*”.

Debido a que el posicionamiento inicial hasta la referencia del vehículo no conlleva restricción temporal alguna. Imponemos los máximos valores posibles para las consignas de velocidad y aceleración eso sí, respetando los principios introducidos en el apartado “Método para el seguimiento con una trayectoria rectilínea”.

El procedimiento para la obtención de dichas consignas se describe a continuación:

$\Delta\theta_{PAN} = \theta_{PAN,i} - \theta_{PAN,i-1}$
$\Delta\theta_{TILT} = \theta_{TILT,i} - \theta_{TILT,i-1}$

$Si\ (\Delta\theta_{PAN} \geq \Delta\theta_{TILT})$

$$V_{PAN} = V_{MAX}$$

$$V_{TILT} = |\, V_{MAX} * \frac{\Delta\theta_{PAN}}{\Delta\theta_{TILT}}\,|$$

$$A_{PAN} = A_{MAX}$$

$$A_{TILT} = |\, A_{MAX} * \frac{\Delta\theta_{PAN}}{\Delta\theta_{TILT}}\,|$$

$Si\ (\Delta\theta_{PAN} < \Delta\theta_{TILT})$

$$V_{PAN} = |\, V_{MAX} * \frac{\Delta\theta_{TILT}}{\Delta\theta_{PAN}}\,|$$

$$V_{TILT} = V_{MAX}$$

$$A_{PAN} = |\, A_{MAX} * \frac{\Delta\theta_{TILT}}{\Delta\theta_{PAN}}\,|$$

$$A_{TILT} = A_{MAX}$$

$\Delta\theta_{PAN}$	Variación del eje θ_{PAN}
$\Delta\theta_{TILT}$	Variación del eje θ_{TILT}
V_{MAX}	Velocidad máxima unidad
V_{PAN}	Velocidad eje θ_{PAN}
V_{TILT}	Velocidad eje θ_{TILT}
A_{MAX}	Aceleración máxima unidad
A_{PAN}	Aceleración eje θ_{PAN}
A_{TILT}	Aceleración eje θ_{TILT}

Tabla 3 Consignas para el posicionamiento.

Lo que se consigue con esto es describir con el láser una trayectoria rectilínea, salvo que, sin la necesidad de ceñirnos a un periodo de control, para agilizar el proceso del posicionamiento inicial, imponemos velocidad y aceleración máximas permitidas para aquel eje que tiene un recorrido angular más largo, y con la relación entre los recorridos angulares de ambos, obtenemos las consignas para el otro eje.

Después se le transmite dichas consignas de movimiento a la unidad con las funciones correspondientes:

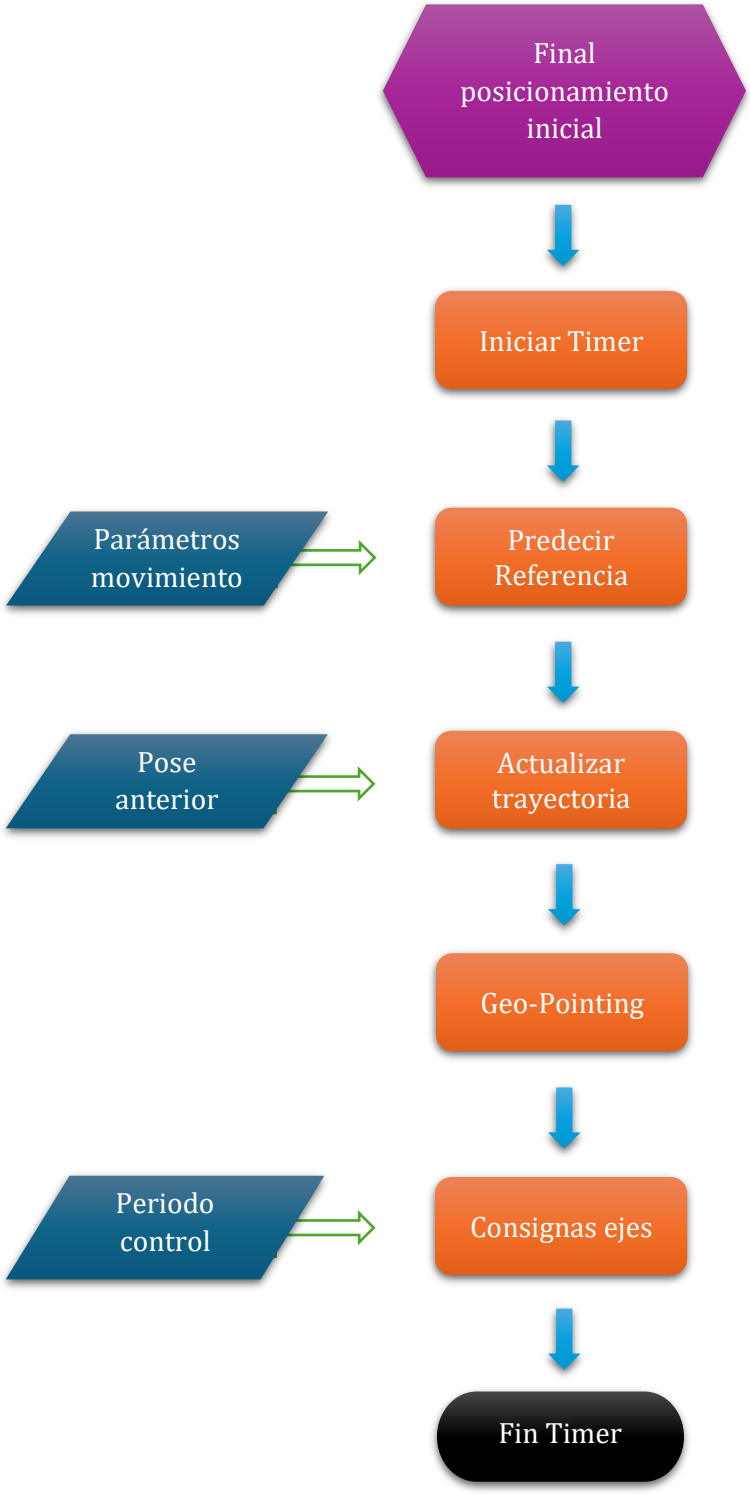
- MandarAceleraciónes(PAN, A_{PAN}, TILT, A_{TILT})
- MandarVelocidades(PAN, V_{PAN}, TILT, V_{TILT})

Y finalmente llevamos a cabo el movimiento, salvo que en esta ocasión, los ejes se deben sincronizar mediante las órdenes de parada y puesta en marcha.

- DetenerEjes()
- MandarPosiciones(PAN, θ_{PAN}, TILT, θ_{TILT})
- MoverEjes()

3.5 “Trayectoria”

Similarmente al script anterior, la misión de este es localizar la referencia del vehículo con los ejes de la unidad, salvo que en esta ocasión, tendremos que predecirla en función de la trayectoria que venga dada por los parámetros de movimiento del robot, y también es necesario obtener una trayectoria rectilínea con el Láser entre dos referencias consecutivas; con tal de aproximarnos lo máximo posible a la trayectoria original.



Donde:

- morado = inicio del programa.
- naranja = procedimiento.
- azul = datos.
- negro = final del programa

Fig. 15 Esquema de funcionamiento del subprograma "Trayectoria".

Se comienza con un temporizador que mide el tiempo de cómputo de cada iteración. Esto nos va a permitir llevar a cabo la ejecución periódica de esta tarea.

Después, se predice la variación de posición y orientación que va a sufrir en los tres ejes el vehículo a causa de los parámetros del movimiento entre el instante actual y el fin de esta ejecución ($Destino_i \rightarrow Destino_{i+1}$) mediante el procedimiento explicado en el capítulo 2.3 *"Método para el seguimiento de un punto cinemático".*

A continuación, calculamos las consignas de los ángulos para alcanzar el nuevo destino mediante los procedimientos explicados en el capítulo 2.2 *"Cálculo de las consignas para los ejes*".

Se calculan las consignas de velocidad para cada eje tal y como se describe en al apartado "Método para el seguimiento con una trayectoria rectilínea". Salvo que en esta ocasión, al contrario que en el algoritmo posicionar, debemos realizar el movimiento ajustándonos al periodo de control, el procedimiento para obtener dichas consignas se muestra a continuación:

$$\Delta\theta_{PAN} = \theta_{PAN,i} - \theta_{PAN,i-1}$$
$$\Delta\theta_{TILT} = \theta_{TILT,i} - \theta_{TILT,i-1}$$

$$V_{PAN} = \frac{|\Delta\theta_{PAN}|}{res * T_{MOV}}$$

$$V_{TILT} = \frac{|\Delta\theta_{TILT}|}{res * T_{MOV}}$$

Donde $\Delta\theta_{PAN}$ y $\Delta\theta_{TILT}$ son los incrementos de posición angular para cada eje de la unidad y T_{MOV} es el tiempo que tienen los ejes para realizar el movimiento.

Y se le comunican a la unidad estas consignas:

- MandarVelocidades(PAN, V_{PAN}, TILT, V_{TILT})

Las consignas para las aceleraciones no serán calculadas en este procedimiento, ya que el tiempo para procesar las órdenes correspondientes por parte de la unidad

es excesivo, lo que hace que sea imposible ceñirse al periodo de control impuesto sobre esta tarea.

Y finalmente llevamos a cabo el movimiento, los ejes de la unidad deben sincronizarse mediante las órdenes de parada y puesta en marcha.

- DetenerEjes()
- MandarPosiciones(PAN, θ_{PAN}, TILT, θ_{TILT})
- MoverEjes()

Para finalizar, se corta el timer y se registra el tiempo que ha tardado todo el código en ejecutarse. Este tiempo se le resta al periodo de control de la aplicación, definido en el script *"Workspace"*, para dar lugar al tiempo sobrante, el mismo tiempo que los ejes de la unidad tienen para realizar el movimiento. Para ello, y con la intención de conservar la periodicidad de la aplicación, se le ordena al programa que espere este mismo tiempo antes de comenzar a ejecutar la siguiente iteración.

- PausarPrograma($T_{CONTROL} - T_{TIMER}$)

Siendo T_{TIMER} el tiempo de cómputo que ha transcurrido desde que se inició la tarea hasta el momento en el que comienza el movimiento.

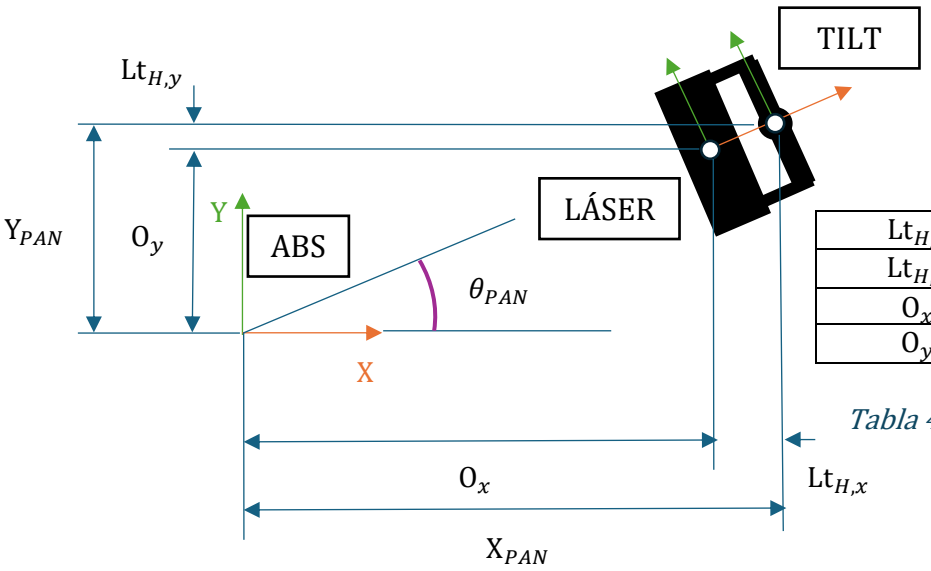
Una vez que hemos descrito el proceso que se lleva a cabo para acometer los objetivos del proyecto, definimos los subprogramas que nos permitirán registrar y mostrar visualmente todo aquello que se ha calculado mediante el algoritmo, para así llevar a cabo simulaciones del sistema completo con ejemplos prácticos y sacar conclusiones de los resultados obtenidos.

3.6 "Graficar_Posición"

Este subprograma contiene un método para comprobar el correcto funcionamiento del geo-pointing de una manera visual, los elementos que muestra en la figura son: el punto Destino, y la línea que une las referencias del punto láser y la de posición absoluta calculada en función de la disposición angular de los ejes.

En primer lugar, se localiza la referencia del láser en coordenadas absolutas:

- Operando sobre lo expuesto en la figura 16 se pueden obtener las componentes horizontales de la referencia del láser:



$Lt_{H,x}$	Proyección X del soporte
$Lt_{H,y}$	Proyección Y del soporte
O_x	Offset X del Láser
O_y	Offset Y del Láser

Tabla 4 Cotas para la referencia del Láser.

Fig. 16 Referencia absoluta del Láser

$$Lt_{H,x} = Lt_H * \cos(\theta_{PAN})$$

$$Lt_{H,y} = Lt_H * \sin(\theta_{PAN})$$

$$O_x = X_{PAN} - Lt_{H,x}$$

$$O_y = Y_{PAN} - Lt_{H,y}$$

- De la figura 5, se puede deducir la componente vertical de la referencia del láser O_z:

$$O_z = Lt_V + Z_{TILT}$$

Tras esto, se calculan las coordenadas X e Y resultantes de la disposición de los dos ejes θ_{PAN} y θ_{TILT} con dos sencillas ecuaciones:

$$O_x = X_{ABS}$$
$$I_y = O_y + O_x * \sin(\theta_{PAN})$$
$$I_z = O_z + O_x * \sin(\theta_{TILT})$$

O_x se iguala a la componente X de la referencia DESTINO (X_{ABS}), para que la comparación de las otras componentes ($I_y \rightarrow Y_{ABS}$ *e* $I_z \rightarrow Z_{ABS}$) se realice sobre el plano XY para que podamos obtener fácilmente las métricas de los errores.

Finalmente, se grafica el punto Destino junto con la recta que une ambas referencias O (LÁSER) y la posición estimada I ($\widehat{DESTINO}$).

3.7 "Graficar_trayectoria"

Este subprograma contiene un método para comprobar el correcto funcionamiento del seguimiento de una manera visual, los elementos que muestra en la figura son: la trayectoria original del vehículo robótico, junto con las referencias de los puntos que conforman la discretización de esta, y la trayectoria formada por el punto de vista del Láser.

- Trayectoria del vehículo:

Se ha empleado el método desarrollado en el apartado: "Implementación algorítmica"; se describe un bucle *"for"*, en el que el periodo de control se ha fraccionado en subperiodos de mucha menor duración, y se realiza el método de manera iterativa hasta que se completa un periodo de control completo:

$$for\ \left(\frac{T_{CONTROL}}{100}, T_{CONTROL}\right) \rightarrow Odometría\ (v, \omega, EUL)$$

Donde v es la velocidad lineal, ω la angular y EUL es el vector que contiene los tres ángulos de inclinación del plano sobre el que sucede el movimiento.

- Cálculo de las referencias para la discretización de la trayectoria del vehículo:

A la referencia relativa $DESTINO_i$ calculada en el algoritmo del subprograma "trayectoria" se le añade el triedro correspondiente a la inclinación del plano del movimiento y además se orienta tangente al movimiento estimado, los tres vectores que componen el triedro se obtienen de la siguiente manera (figura 17):

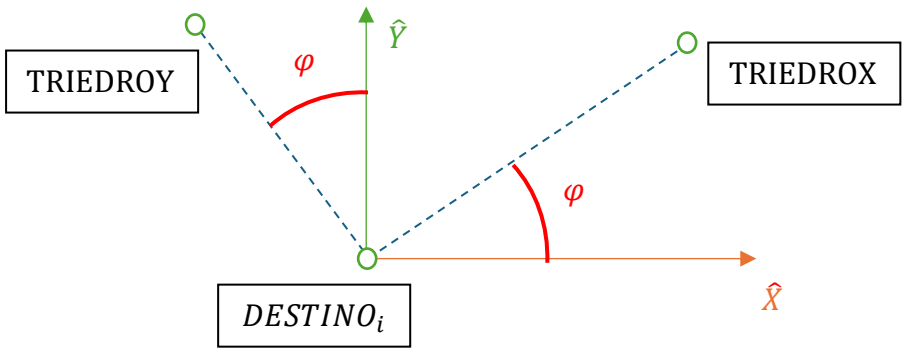


Fig. 17 Obtención del triedro de la referencia $DESTINO_i$

$$TRIEDROX = [\,0.1 * \cos(\varphi)\,, 0.1 * \sin(\varphi), 0]$$

$$TRIEDROY = [\,-0.1 * \sin(\varphi)\,, 0.1 * \cos(\varphi), 0]$$

$$TRIEDROZ = 10 * (TRIEDROX \times TRIEDROY)$$

$TRIEDROX$ se calcula en la dirección del movimiento sobre el plano conforme al ángulo girado φ y es tangente a la trayectoria, $TRIEDROY$ es el vector normal a la trayectoria sobre el plano $\hat{X}\hat{Y}$, y $TRIEDROZ$ se corresponde con el producto

vectorial entre ambos, dando lugar a un vector normal al plano, todos ellos tienen como referencia de origen $DESTINO_i$.

- Trayectoria del Láser:

Se describe un bucle *"for"*, en el que el periodo de control se ha fraccionado en subperiodos de mucha menor duración, y se hace una llamada al script "graficar_posición" de manera iterativa, hasta que se completa un periodo de control completo:

$$for\ \left(\frac{T_{CONTROL}}{100}, T_{CONTROL}\right) \rightarrow Graficar\_posición\ (\widehat{DESTINO}_i, \theta_{PAN,i}, \theta_{TILT,i})$$

Donde $T_{CONTROL}$ es el periodo de control; $DESTINO_i$ es la referencia del vehículo predicha y $\theta_{PAN,i}$, $\theta_{TILT,i}$ son las posiciones angulares calculados mediante el algoritmo "trayectoria".

Se elaboran dos trayectorias trapezoidales de velocidad para el cálculo de las posiciones angulares de cada eje de la unidad (figuras 18-20):

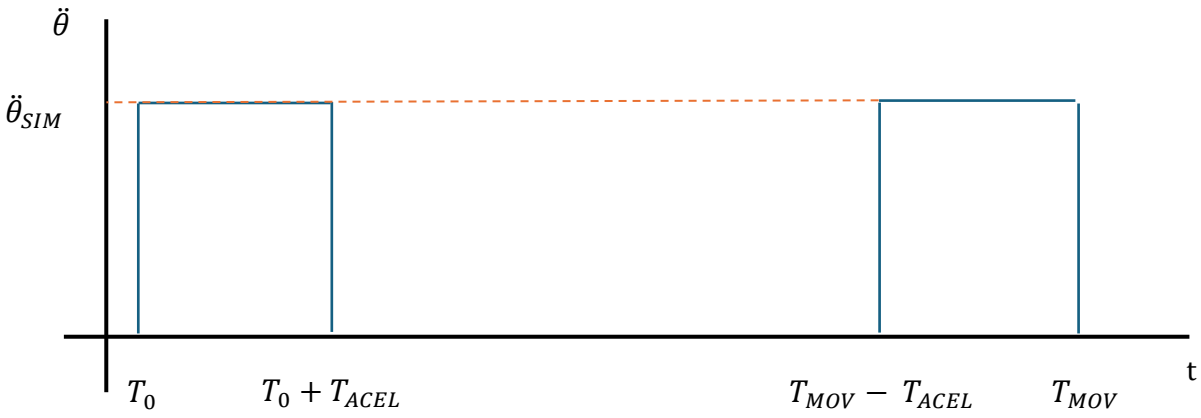


Fig. 18 Consigna de aceleración para perfil trapezoidal de velocidad

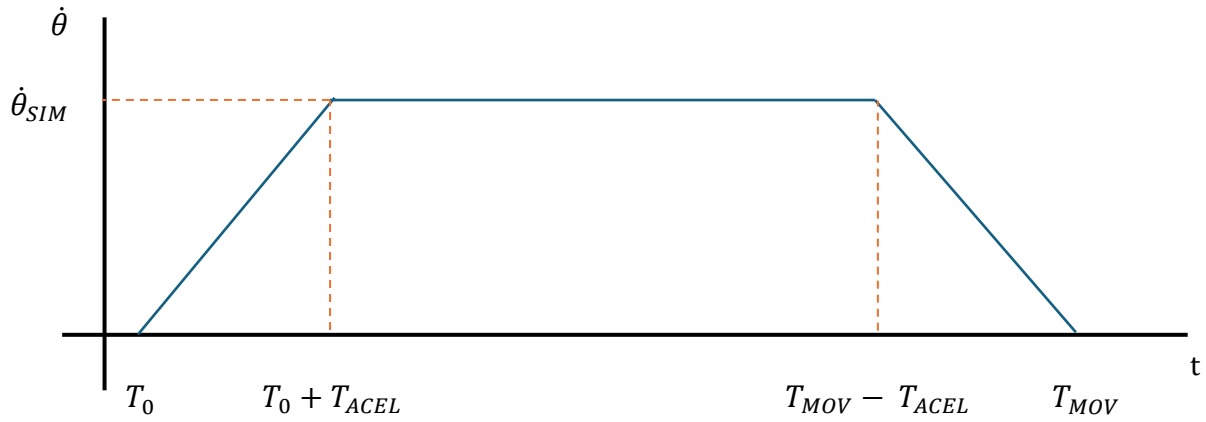


Fig. 19 Perfil velocidad trapezoidal para la posición angular de los ejes

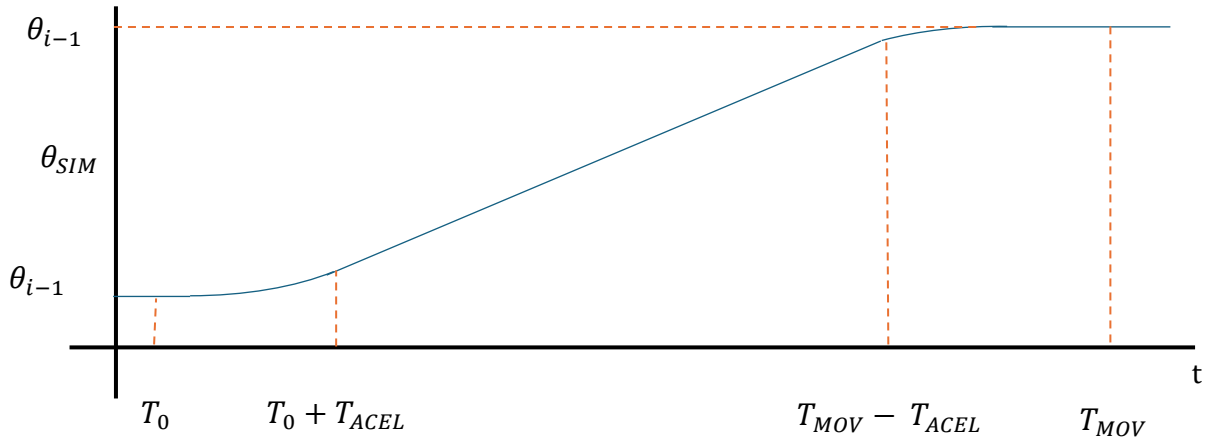


Fig. 20 Variación de posición angular para un perfil de velocidad trapezoidal

- La trayectoria descrita por el Láser se divide en varios tramos para facilitar los cálculos y se obtiene de la siguiente manera:

$if\ (t < T_{ACEL})$ $\quad - \boldsymbol{Aceleración} -$

$$\theta_{SIM} = \theta_{i-1} + \frac{1}{2}\ddot{\theta}_{SIM}\, t^2$$

$if\ (t > T_{MOV} - T_{ACEL})$ $\quad - \boldsymbol{Deceleración} -$

$$\theta_{SIM} = \theta_i - \frac{\dot{\theta}_{SIM} T_{ACEL}}{2} + \dot{\theta}_{SIM}\left(t - (T_{MOV} - T_{ACEL})\right) - \frac{1}{2}\ddot{\theta}_{SIM}\left(t - (T_{MOV} - T_{ACEL})\right)^2$$

$else$ $\quad - \textbf{Crucero} -$

$$\theta_{SIM} = \theta_{i-1} + \frac{\dot{\theta}_{SIM} T_{ACEL}}{2} + \dot{\theta}_{SIM}\,(t - T_{ACEL})$$

Capítulo 4. Realización de los ensayos.

Los experimentos que vamos a ir realizando seguirán la misma continuidad que la que se ha descrito con la teoría, confirmaremos el funcionamiento del modelo y nos detendremos según se determine oportuno para sacar algunas conclusiones, posibles mejoras, nuevas problemáticas que puedan surgir e incluso algunas correcciones en última instancia.

En vista de que el sistema completo, con vehículo, base, comunicaciones, etc. todavía está en línea de desarrollo, es decir, que aún no está listo para experimentos reales, se propone simular algunos casos prácticos como vía alternativa a la verificación de la aplicación, donde seremos nosotros quienes por programa, mediante el script *"referencias"*, describiremos las consignas del movimiento y las metas a alcanzar de una manera similar (aunque simplificada) a como se llevaría a cabo mediante telemanipulación del vehículo robótico.

4.1 Análisis del Geo-Pointing

El uso más básico que se le impone a esta aplicación es el correcto apuntado a un objetivo situado en cualquier lugar del espacio tridimensional, siempre y cuando este sea alcanzable dadas las limitaciones mecánicas de la unidad. Basta con definir puntos al azar junto con sus tres coordenadas y pasarlos al subprograma "posicionar". Este debería ser capaz de obtener las consignas de posición para sendos ángulos. Con la combinación de estos, recogemos la información acerca del punto exacto al que estamos apuntando y comparamos con el valor inicial que sirvió como entrada para el cálculo; en este caso, se ha designado la referencia inicial mediante las coordenadas [2.15, -0.33, 0.63]. A continuación, se muestran dos figuras donde se puede apreciar el desempeño de nuestra aplicación, (figuras 21 y 22):

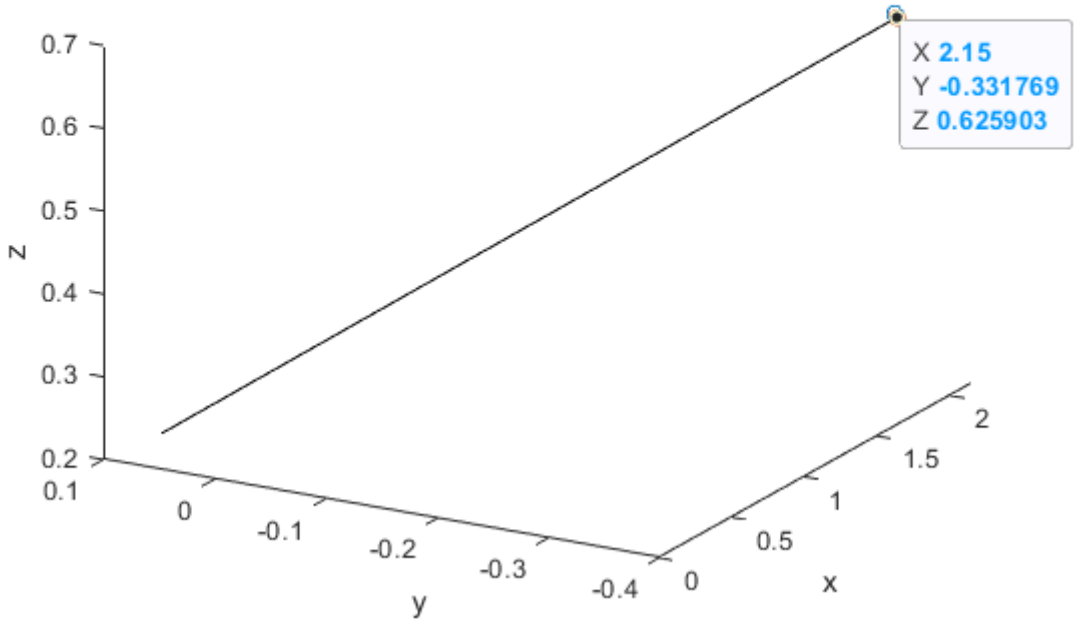


Fig. 21 Demostración Geo-Pointing

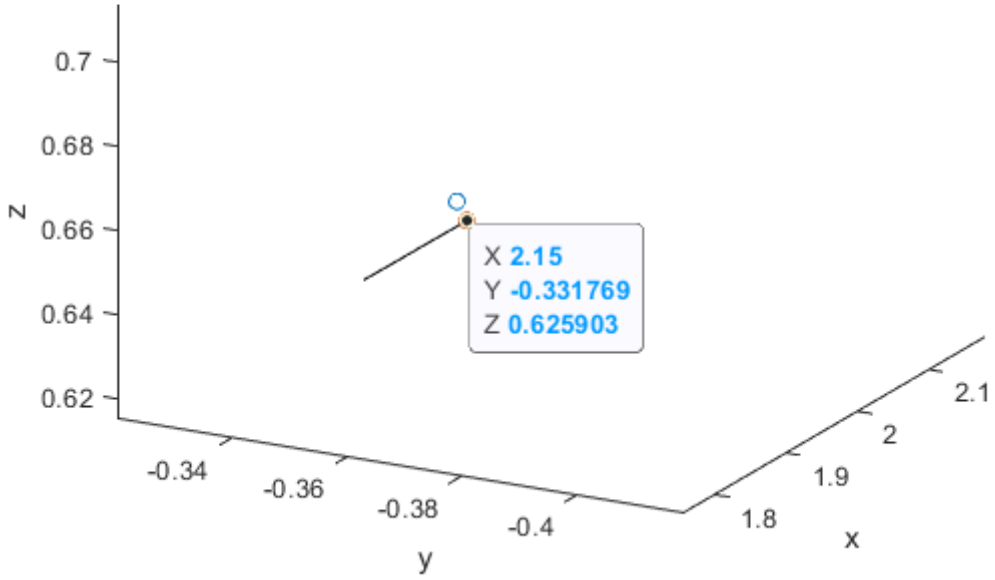


Fig. 22 Detalle demostración Geo-Pointing

A primera vista, observamos que las coordenadas calculadas se aproximan razonablemente al destino inicial, sin embargo, se puede inferir el error que se ha cometido comparando las coordenadas de ambos puntos

Los valores obtenidos se traducen en unos errores de 1.769 mm y 4.097 mm para "Y" y "Z" respectivamente. Esto supone unos errores relativos de 0, 0.536, y 0.65 por ciento respectivamente sobre el plano YZ. Sin embargo, es un resultado satisfactorio debido a una única razón, la resolución de la unidad según se puede extraer mediante comunicación serie es de 0.8976 mRad por cada posición *"discreta" d*e los ejes. Realizando una operación sencilla llegamos a la siguiente conclusión:

$$H_{100} = 100 * \sin\left(Res_{ejes}\right) = 5.1432\ m$$

En la operación que se realiza calculamos el segmento que llega a cubrir una sola posición angular de cualquiera de los ejes, (pues tienen la misma resolución) a una distancia horizontal (que no necesariamente en X) de 100 metros (5 por ciento). En conclusión, se puede decir que nuestros cálculos son lo suficientemente precisos como para determinar un desempeño más que aceptable y que la tarea más básica de nuestro proyecto rinde adecuadamente.

4.2 Análisis del seguimiento

Una vez que hemos logrado localizar el objetivo en el espacio, el siguiente paso es su seguimiento. Para ello, se dota de movimiento a la referencia del vehículo, tal y como se describió en el apartado 2.3 "Método para el seguimiento de un punto cinemático".

El movimiento que se va a describir es una circunferencia, con unos valores de 1 $\frac{m}{s}$ y 2.74 $\frac{rad}{s}$ para sendas velocidades lineal y angular respectivamente, y [0 pi/4 0] es el vector que representa la inclinación de la trayectoria del movimiento mediante los ángulos de Euler.
Como se puede observar en las siguientes figuras (23-25), en cada una de ellas se representan varios puntos de vista diferentes de la misma trayectoria, para facilitar su comprensión, cada elemento se ha ilustrado con un color diferente.

- Magenta: trayectoria real descrita por el vehículo.
- Negro: trayectoria dibujada por el láser sobre el plano del movimiento.
- Rojo: vector tangente a la trayectoria calculada que representa la dirección del movimiento.
- Verde: vector normal a la trayectoria, define junto al vector tangente el plano de movimiento.
- Azul: vector normal al plano de movimiento.

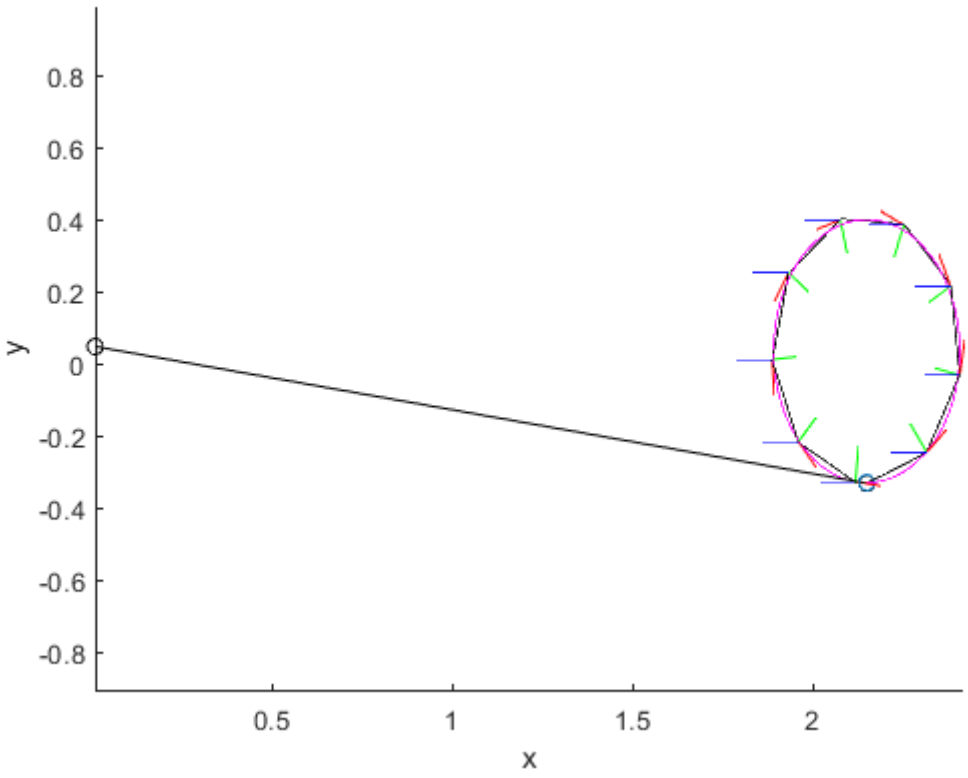


Fig. 23 Demostración trayectoria circular vista desde el plano XY.

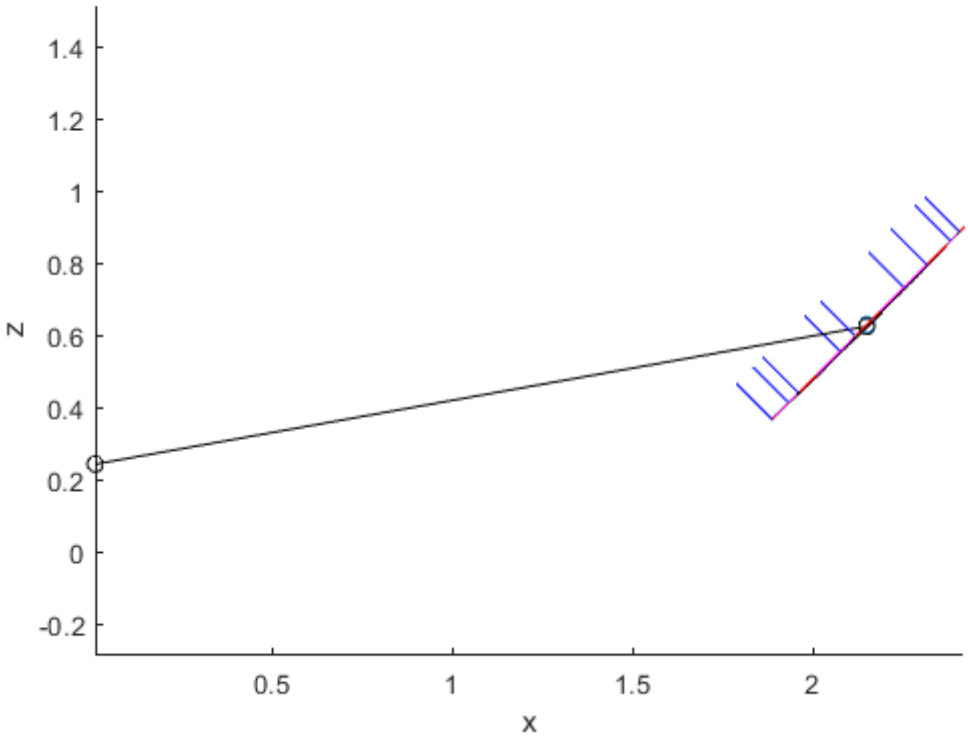


Fig. 24 Demostración trayectoria circular vista desde el plano XZ.

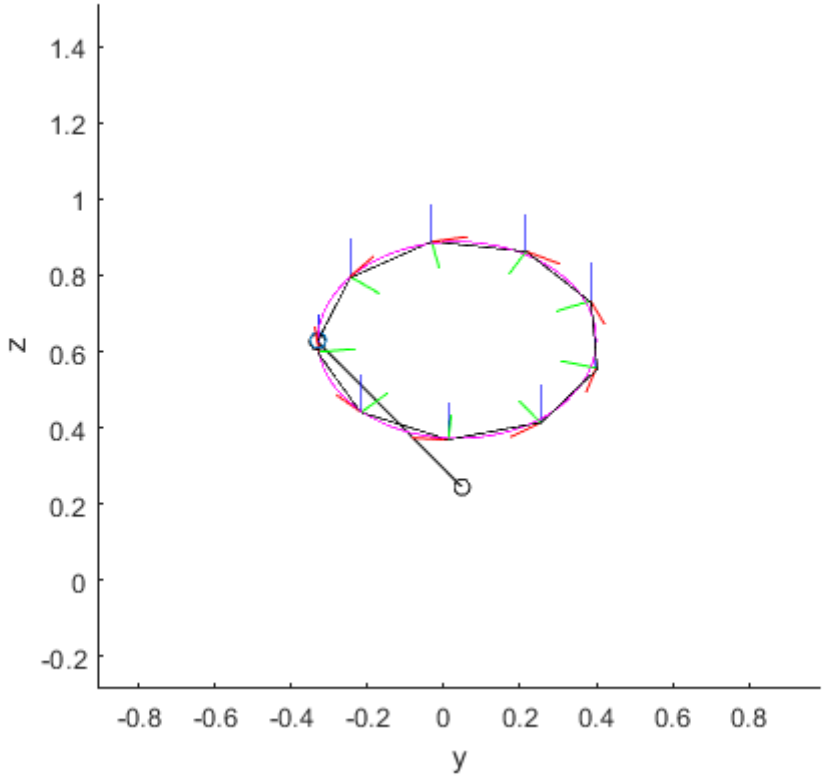


Fig. 25 Demostración trayectoria circular vista desde el plano YZ.

Tal y como se puede observar en las figuras anteriores, se lleva a cabo la discretización de la trayectoria real que se compone de las sucesivas referencias que se le imponen a la unidad. También se representa la trayectoria teórica que describiría el láser, y se puede observar cómo esta resulta bastante fiel con respecto a la original del vehículo robótico.

Para finalizar, vamos a exponer un ejemplo práctico en el que el vehículo realizará movimientos no constantes en una misma trayectoria (graficas 26-30).

La descripción completa de los parámetros que componen cada movimiento a lo largo de toda la trayectoria han sido indicados en el script *"referencias":*

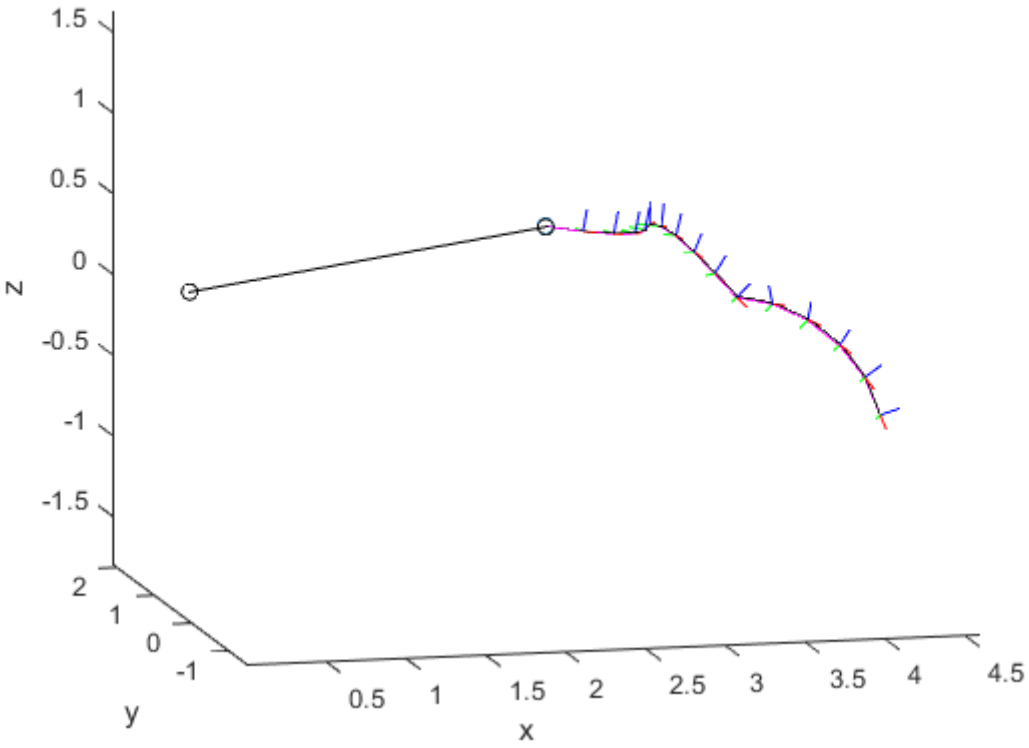


Fig. 26 Demostración ejemplo práctico, vista de tres dimensiones.

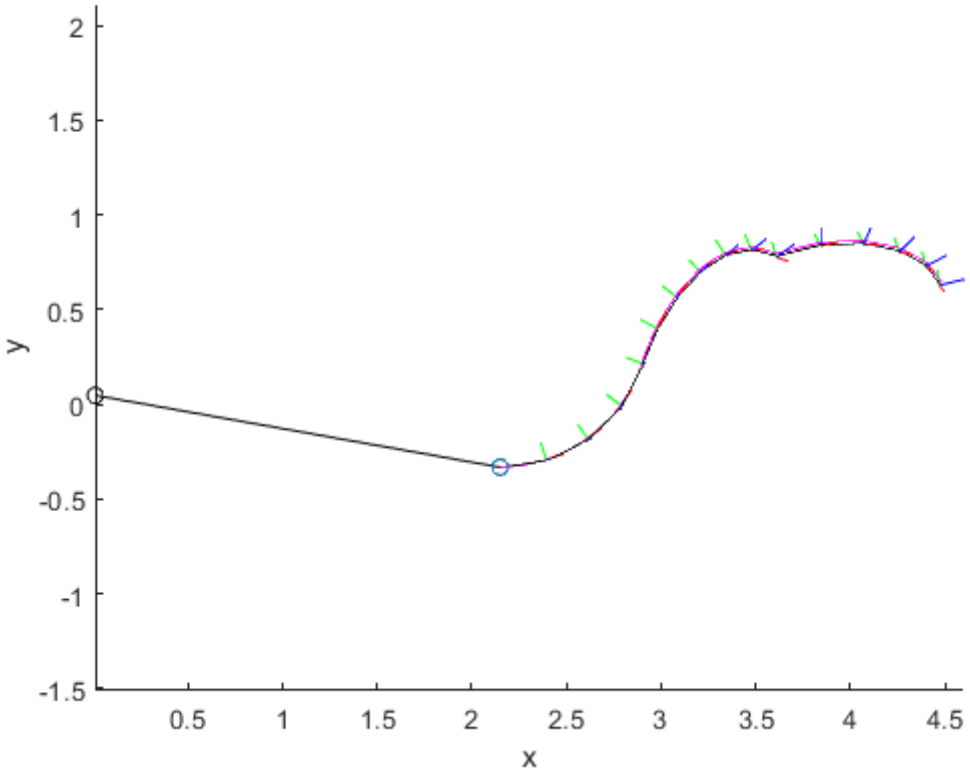


Fig. 27 Demostración ejemplo práctico, vista desde el plano XY.

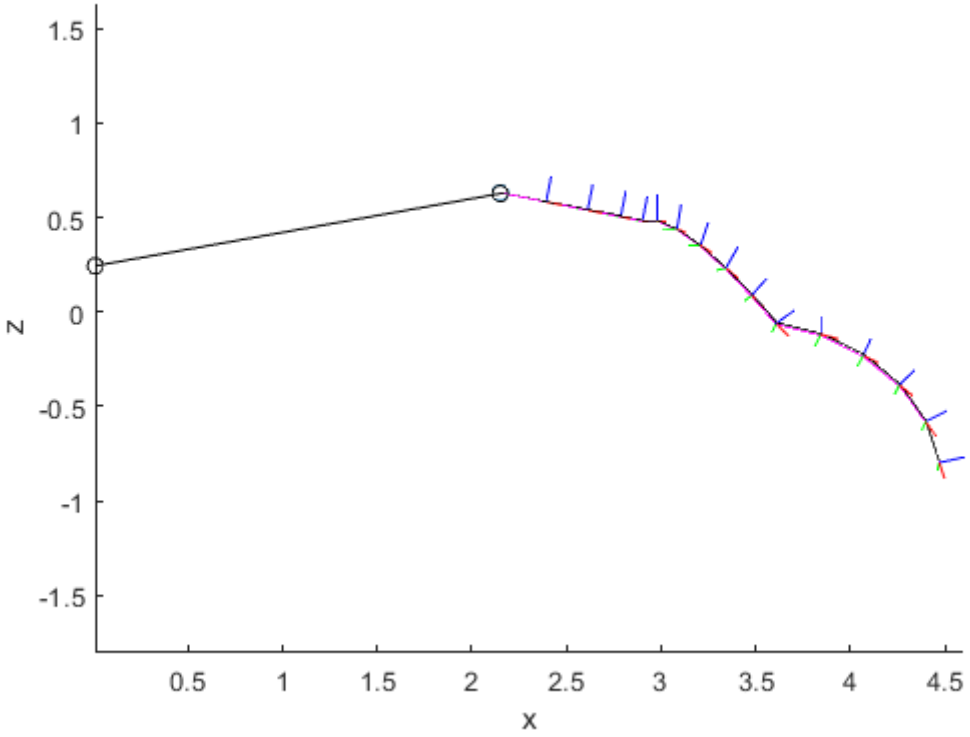


Fig. 28 Demostración ejemplo práctico, vista desde el plano XZ.

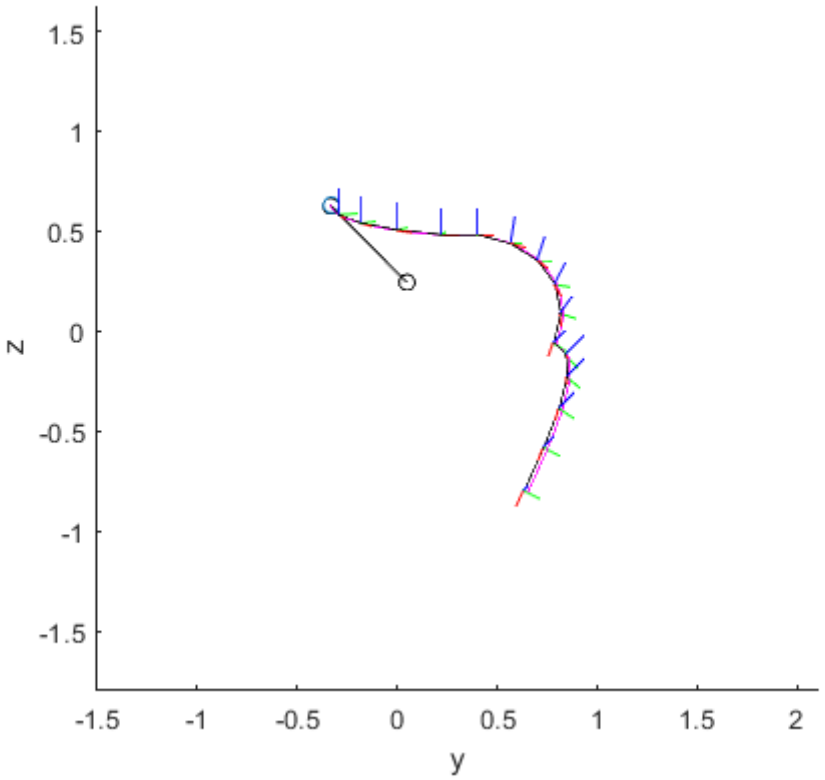


Fig. 29 Demostración ejemplo práctico, vista desde el plano YZ.

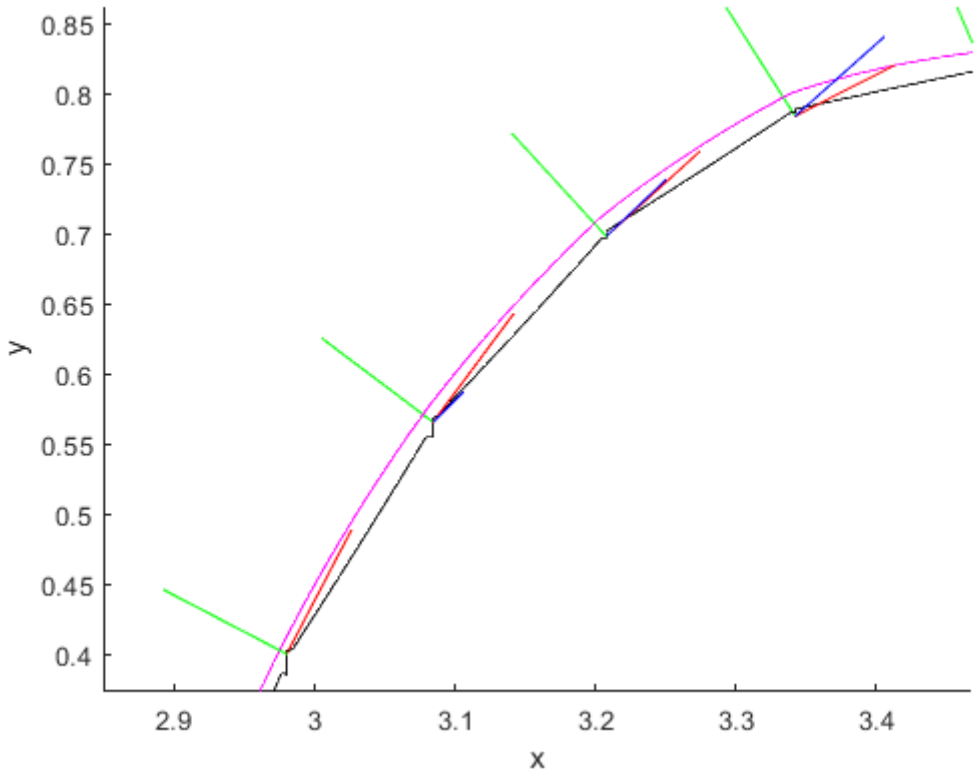


Fig. 30 Detalle demostración ejemplo práctico, vista desde el plano XY.

En las figuras anteriores se puede apreciar el parecido entre ambas trayectorias, tanto real como calculada, y destacamos la robustez del programa ante cambios de orientación.

Y con las funciones correspondientes podemos obtener la siguiente tabla de valores:

ITER	$\Delta\theta_{PAN}$	$\Delta\theta_{TILT}$	$\dot{\theta}_{PAN}$	$\dot{\theta}_{TILT}$	VEL_{PAN}	VEL_{TILT}	$\ddot{\theta}_{PAN}$	$\ddot{\theta}_{TILT}$	$ACEL_{PAN}$	$ACEL_{TILT}$
1	**-0,17469**	**0.1759**	1,78	1,79	1,78	1,79	71,31	71,80	71,31	71,80
2	**0,048392**	**-0.0761**	0,69	1,0	0,69	1,08	71,80	71,80	71,80	71,80
3	**0,0928867**	**0.0945**	1,32	0,13	1,32	0,13	71,80	71,80	71,80	71,80
4	**0,0336687**	**0.0853**	0,48	1,21	0,48	1,21	71,80	71,80	71,80	71,80
5	**-0,063516**	**0.0647**	0,90	0,92	0,90	0,92	71,80	71,80	71,80	71,80
6	**-0,089136**	**0.0265**	1,27	0,37	1,27	0,37	71,80	71,80	71,80	71,80
7	**-0,016222**	**-0.0880**	0,23	1,25	0,23	1,25	71,80	71,80	71,80	71,80
8	**0,0740935**	**-0.0533**	1,05	0,76	1,05	0,76	71,80	71,80	71,80	71,80
9	**0,0834523**	**0.0423**	1,19	0,60	1,19	0,60	71,80	71,80	71,80	71,80
10	**0,0448159**	**0.0085**	0,64	0,12	0,63	0,12	71,80	71,80	71,80	71,80
11	**0,0592037**	**0.0092**	0,84	0,13	0,84	0,13	71,80	71,80	71,80	71,80
12	**0,0737338**	**0.0038**	1,05	0,05	1,05	0,05	71,80	71,80	71,80	71,80
13	**0,0843569**	**-0.0082**	1,20	0,11	1,20	0,11	71,80	71,80	71,80	71,80
14	**0,0871330**	**-0.0254**	1,24	0,36	1,24	0,36	71,80	71,80	71,80	71,80
15	**0,0803302**	**-0.0443**	1,14	0,63	1,14	0,63	71,80	71,80	71,80	71,80
16	**0,0844055**	**-0.0770**	1,20	1,09	1,20	1,09	71,80	71,80	71,80	71,80
17	**0,0884887**	**-0.0683**	1,26	0,97	1,26	0,97	71,80	71,80	71,80	71,80
18	**0,087224**	**-0.0574**	1,24	0,82	1,24	0,82	71,80	71,80	71,80	71,80
19	**0,0802881**	**-0.0459**	1,14	0,65	1,14	0,65	71,80	71,80	71,80	71,80
20	**0,0689612**	**-0.0357**	0,98	0,51	0,98	0,51	71,80	71,80	71,80	71,80

Donde:

- ITER indica el número de la iteración actual.
- $\Delta\theta_{PAN}$ es la variación de posición del eje PAN.
- $\Delta\theta_{TILT}$ es la variación de posición del eje TILT.
- $\dot{\theta}_{PAN}$ es la velocidad de consigna para el eje PAN
- $\dot{\theta}_{TILT}$ es la velocidad de consigna para el eje TILT
- VEL_{PAN} es la velocidad de consigna para el eje PAN leída por la unidad.
- VEL_{TILT} es la velocidad de consigna para el eje TILT leída por la unidad.
- $\ddot{\theta}_{PAN}$ es la aceleración de consigna para el eje PAN.
- $\ddot{\theta}_{TILT}$ es la aceleración de consigna para el eje TILT.
- $ACEL_{PAN}$ es la aceleración de consigna para el eje PAN leída por la unidad.
- $ACEL_{TILT}$ es la aceleración de consigna para el eje TILT leída por la unidad.

Tabla 5 Tabla de valores de las consignas de movimiento de la unidad.

Dicho esto, concluimos el apartado de la simulación de nuestro sistema, donde hemos visualizado el rendimiento satisfactorio de nuestra aplicación ante consignas tanto de posicionamiento como de predicción de movimiento, y la precisión con la que estas tareas se llevan a cabo.

- *Análisis de tiempos.*

Una vez que hemos comprobado que el programa funciona correctamente, es decir, las consignas que se crean son adecuadas, y que el movimiento de los ejes se realiza ajustándose a las restricciones impuestas, nos proponemos analizar el aspecto temporal. No podemos olvidar que la imposición principal para el correcto funcionamiento del programa es el tiempo, la periodicidad, y la repetibilidad de todos los cálculos e interacciones con la unidad.

Debido a que el proyecto como compendio todavía está en fase de realización, aún es demasiado pronto para hablar de cuál es el tiempo exacto entre el que se deben llevar a cabo todas las tareas propias de la unidad. Así que vamos a intentar conseguir que estas se ejecuten lo más rápido posible, ya que a nuestro entender, cuanto más tiempo invirtamos en ellas, más tiempo perderemos de vista al vehículo robótico, y por lo tanto mayor probabilidad de fracaso en las comunicaciones.

Para lograrlo, hemos llevado a cabo un estudio de los tiempos de computación de las tareas más demandantes, obteniendo los datos que se exponen en la tabla 5:

Nombre Tarea	Tiempo de Cómputo (ms)
MandarPosición()	47
MandarVelocidad()	11
Mandaraceleración()	60
Detener/Mover ejes	6
trayectoria	200
posicionar	320
Tabla de valores	70
Graficar_Posición	13
Graficar	2300

Tabla 6 Tiempos de cómputo de las diferentes tareas.

* Nota: el tiempo de ejecución del resto de los cálculos es despreciable en comparación. *

Para realizar un buen seguimiento de la referencia del robot móvil, tenemos que acotar el periodo de control. Se ha observado mediante simulación que las perturbaciones en la trayectoria rectilínea del Láser debido a los tramos de aceleración y deceleración son despreciables, de tal manera que en el subprograma "trayectoria" no se calcularán; además, tanto realizar la tabla con los valores de las consignas para los ejes como graficar la trayectoria es demasiado costoso computacionalmente, así que hemos decidido que estas tareas no se lleven a cabo por nuestra aplicación, y que se valore una manera alternativa de ser procesadas, como tal vez otra instancia distinta.

En conclusión, para imponer una duración asequible al periodo de control, comentamos las secciones de código que pregunta a la unidad por los valores de las consignas ("Tabla de valores") y estimamos el tiempo total que cuesta cada iteración del algoritmo "trayectoria":

$$2 * MandarPosición() + 2 * MandarVelocidad() + 2 * Detener/Mover\ ejes = 128\,ms$$

Por lo que asignamos los periodos de control y movimiento.

$$\boldsymbol{T_{CONTROL}} = 200\,ms$$

$$\boldsymbol{T_{MOV}} = 200 - 128 = 72\,ms$$

Capítulo 5. Posibles mejoras y modificaciones.

Este capítulo está dedicado a aquellas propuestas e hipótesis que, tras la finalización de este proyecto, podrían llegar a ser implementadas con la intención de ampliar las funcionalidades del sistema.

Algunas de las ideas más interesantes se desarrollan a continuación:

- *Control de velocidad*

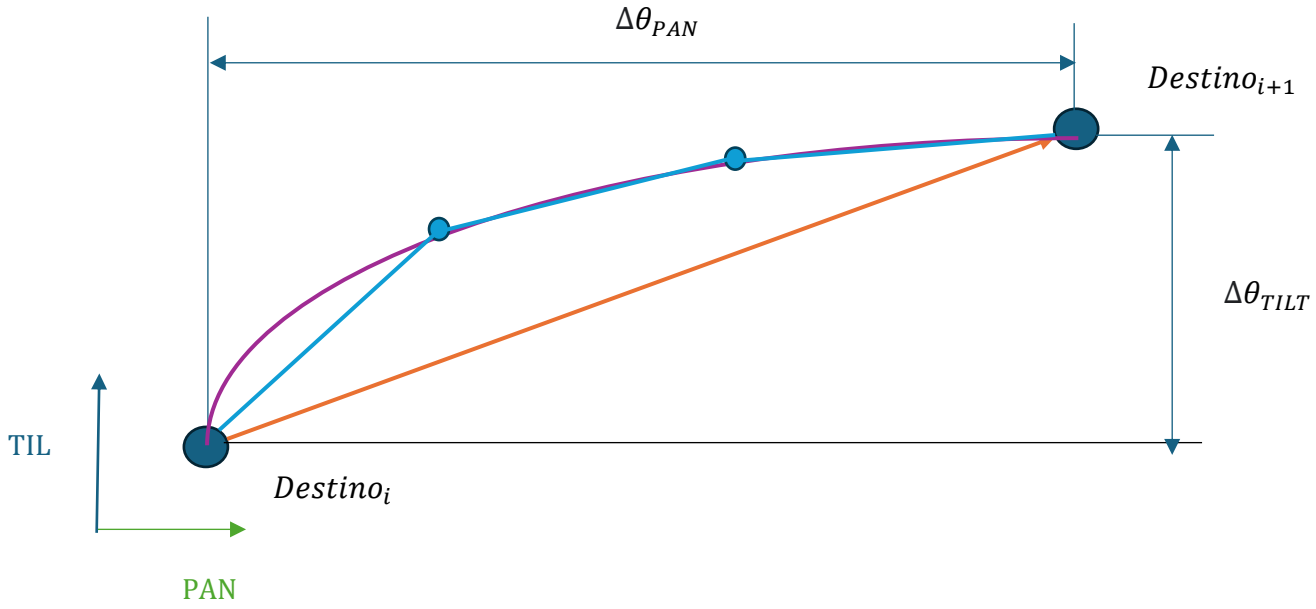
Tal como se ha observado durante la experimentación, el mayor problema para realizar un control preciso en tiempo real es el tiempo de cómputo del algoritmo, en particular, el tiempo total de procesamiento y comunicaciones de las órdenes comandadas a la unidad. Muchas de ellas, (especialmente las que conciernen al movimiento) toman varias decenas de milisegundos cada una. Si estas se acumulan, el periodo de muestreo de programa deberá aumentar con tal de ajustarse a este tiempo, y por lo tanto funciona en detrimento del control, haciéndolo mucho menos robusto.

Lo que se propone para solucionar esto es el control de la unidad mediante la velocidad. Con esto conseguimos que el movimiento del láser comience antes, es decir, que perderemos al objetivo de vista menos tiempo, sin embargo, esto podría conllevar el empeoramiento de la precisión, ya que ahora la posición no es el objeto principal del control.

Esto podría dar pie a otra posible mejora; la idea es emular la trayectoria que va a seguir el vehículo con el láser de la manera más fiel posible, puesto que el control de velocidad podría llegar a ser considerablemente más rápido.

(1) Podríamos enviar dos o incluso tres consignas de velocidad diferentes en el tiempo que tardábamos con el control de posición. Y variando las consignas podríamos interpolar entre varios puntos determinados de la trayectoria real para aproximarnos lo máximo posible a esta con el puntero láser.

(2) Con una sola consigna de velocidad para todo el recorrido, encontrar las velocidades para cada eje que mejor emulan la trayectoria original vista desde la unidad. Ahora no tendríamos que sincronizar los ejes, sino aplicar más velocidad a uno de los ejes para llegar a “moldear” la trayectoria descrita por el láser para que esta se parezca lo máximo posible a la trayectoria original.



Donde:

- morado = trayectoria real vista desde la unidad.
- naranja = trayectoria rectilínea del láser calculada.
- azul = trayectoria del láser interpolada entre varios puntos.

Fig. 31 Trayectorias alternativas para el Láser.

- *Comunicaciones de vuelta desde el vehículo.*

La idea detrás de esto es ubicar otra unidad *"PAN-TILT"* en el propio vehículo, conociendo los parámetros del movimiento de este, tendríamos que usar y ampliar el subprograma "Posicionar" para así poder apuntar al receptor de señal que se encuentra en la base de operaciones con respecto al vehículo en movimiento y así completar el circuito de las comunicaciones entre base y vehículo.

- *Calibración del láser*

Hasta ahora se ha supuesto que el láser está perfectamente localizado y orientado con respecto a los ejes de movimiento, como medida adicional, se propone ajustar la referencia LÁSER mediante medidas experimentales, para así poder corregir las consignas de posición θ_{PAN} y θ_{TILT} y mejorar la precisión.

- *Añadir robustez al posicionamiento de los ejes*

Otra de las funcionalidades que todavía no hemos tenido en cuenta es el localizador vía GPS que el vehículo robótico tiene incorporado. Esto abre la posibilidad de conocer la posición actual del vehículo, dato con el que podemos modificar la trayectoria que hemos calculado desde la unidad y así corregir posibles errores en el cálculo o fallos de aproximación que hemos ido arrastrando desde operaciones anteriores para obtener un control de posición mucho más preciso, flexible y robusto.

- *Expandir funcionalidad del algoritmo a drones*

Como propuesta final, se plantea expandir la aplicación para que las tareas que se han desarrollado en este documento sean también aplicables a vehículos voladores telemanipulados.

Capítulo 6: Bibliografía

[1] G. Dudek and M. Jenkin, *Computational Principles of Mobile Robotics*, 2nd ed. UK: Cambridge University Press, 2010.

[2] R. Siegwart and I. Nourbakhsh, *Introduction to Autonomous Mobile Robots. A Bradford book.* USA: The MIT Press, 2004.

[3] Manual del Programador para la unidad PTU-D46.

https://www.sustainable-robotics.com/reference/PTU/PTU-manual-D46-2.15.pdf

[4] J. Borenstein, H. R. Everett L. Feng, D. Wehe. *Mobile Robot Positioning: Sensors and Techniques.* Journal of Robotics Systems, 1997.

[5] Matlab web-page: https://es.mathworks.com/products/matlab-online.html

[6] TENVEO (EMPRESA) https://www.tenveo-video-conference-es.com/info/how-rs232-serial-communication-works-33863320.html

Capítulo 7: Anexo.

Carpeta drive con el siguiente contenido:

- “Workspace”.
- “Referencias”.
- “Graficar_trayectoria”.
- “Trayectoria”.
- “Posicionar”.
- “Graficar_posicion”.
- “PTU”.

Enlace:

https://drive.google.com/drive/u/0/folders/1xU2ziJAOfn0-jsJBvKO5JNR4GDOyXNHQ

TRABAJOS DE FIN DE GRADO / FIN DE MÁSTER

Escuela de Ingeniería y Arquitectura Universidad Zaragoza

DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD

(Este documento debe remitirse a seceina@unizar.es dentro del plazo de depósito)

D./Dª. Javier Franco Ramírez ,

en aplicación de lo dispuesto en el art. 14 (Derechos de autor) del Acuerdo de 11 de septiembre de 2014, del Consejo de Gobierno, por el que se aprueba el Reglamento de los TFG y TFM de la Universidad de Zaragoza,

Declaro que el presente Trabajo de Fin de Estudios de la titulación de

Grado en Ingeniería Electrónica y Automática (Título del Trabajo)

Geo-pointing mediante unidad Pan-Tilt aplicado al movimiento y las comunicaciones de un vehículo robótico con Matlab.

es de mi autoría y es original, no habiéndose utilizado fuente sin ser citada debidamente.

Zaragoza, a 24 de junio de 2024

Fdo: Javier Franco Ramírez