Universidad Zaragoza
1542

Trabajo Fin de Máster

Seguimiento óptimo de una persona utilizando un robot móvil equipado con una cámara y un PTZ

Optimal tracking of a person with a mobile robot equipped with a camera and a PTZ

Autor

Víctor Fuertes Bueno

Directores

Eduardo Montijano Muñoz

ESCUELA DE INGENIERÍA Y ARQUITECTURA
2020

AGRADECIMIENTOS

Como preámbulo a este Trabajo Fin de Master (TFM) quiero agradecer su apoyo y colaboración a todas aquellas personas que lo han hecho posible.

En primer lugar, agradecer a Don Eduardo Montijano Muñoz, Director de este Proyecto Final de Máster, su interés y apoyo constantes. En todo momento me ha transmitido la energía y el ánimo necesarios para afrontar los problemas y obstáculos que han ido surgiendo semana tras semana. Sus conocimientos y su experiencia han constituido un pilar fundamental para el resultado obtenido, sin olvidar el buen trato recibido y la confianza en mí depositada. Sin duda alguna su labor ha sido esencial para la consecución de este proyecto.

En segundo lugar, agradecer al Instituto Universitario de Investigación en Ingeniería de Aragón (I3A) la oportunidad que me ha brindado de realizar el TFM con una de sus becas de investigación, en especial al área de Robótica, Percepción y Tiempo Real, que me ha permitido hacer uso de sus instalaciones durante los meses de trabajo.

Por otra parte, mi agradecimiento a todos los compañeros del grupo de Tesis y TFM's, tutorizado por Eduardo Montijano, por haber hecho de estos meses una experiencia inolvidable y productiva.

Agradecer igualmente a todos los profesores y compañeros de clase que durante toda esta etapa de formación me han acompañado, su aportación a mi enriquecimiento personal e intelectual.

Por último, agradecer a mi familia su apoyo incondicional. Mis padres han hecho posible que haya llegado hasta aquí al haber dedicado buena parte de su tiempo y sus recursos a mi formación y a mi desarrollo personal.

A todos ellos, muchas gracias.

RESUMEN

El objetivo principal de este TFM es desarrollar un algoritmo de control que permita calcular trayectorias óptimas para monitorizar a una persona utilizando un robot móvil equipado con una cámara RGB-D y un Pan-Tilt-Zoom (PTZ).

Para hacer esto posible, el trabajo se divide en tres partes. En la primera se desarrolla el control del sistema, subdividido en el control del PTZ para mantener a la persona en el campo de vista con sus giros y en el control del robot terrestre, que proporciona el movimiento necesario al conjunto para enfocar al objetivo desde una perspectiva deseada. La segunda parte se centra en la implementación de ambos controles en un software de simulación para analizar su eficacia en entornos de complejidad creciente. Por último, se realizan experimentos en una plataforma real para validar el correcto funcionamiento de la propuesta de control.

Tras analizar diferentes propuestas de control para el PTZ en el TFM se ha propuesto un control basado en geometría. A través de la posición relativa entre la cámara y la región 3D de la persona a centrar en la imagen se calculan directamente los dos ángulos de giro del PTZ. Para el control del robot se ha utilizado *Model Predictive Control (MPC)*, una estrategia de control óptimo. En el TFM se ha estudiado la teoría asociada a este tipo de control y se ha propuesto una formulación del problema de seguimiento consistente con su metodología.

Para la evaluación de la propuesta se han realizado tanto pruebas en simulación como con una plataforma real. La simulación se ha realizado, en primer lugar, utilizando *Matlab*. Se ha creado un entorno completo de simulación e implementado los controles propuestos. Para comprobar su eficacia, se ha realizado un análisis de *Montecarlo* variando diferentes parámetros e introduciendo situaciones cada vez más realistas. Además, como paso previo a la implementación en dispositivos reales, se ha utilizado la herramienta *Stage*, propia del software *ROS*, donde se obtienen resultados más cercanos a la realidad. Para ello se ha adaptado el código de *Matlab* y añadido comandos para la comunicación de *Matlab* con *ROS*.

Finalmente, para corroborar todo lo anterior con dispositivos reales se ha empleado el PTZ *Arbotix turret*, que tiene dos ángulos de giro, la cámara *Realsense* con sensor de profundidad y los robots terrestres *Turtlebot*. Para su implementación se ha estudiado el funcionamiento de la plataforma *ROS* y creado los nodos de comunicación entre los diferentes dispositivos.

Índice

Capítulo 1

INTRODUCCIÓN

1.1. Motivación

El siglo XXI se caracteriza por un uso masivo de robots móviles, tanto terrestres como aéreos. La versatilidad de los mismos ha permitido su utilización en ámbitos tan dispares como la seguridad o el ocio. Hay aplicaciones en las que los robots se utilizan principalmente en tareas de vigilancia, debido a su movilidad y control remoto, ya que con una cámara equipada en un robot móvil es posible monitorizar grandes zonas y entornos dinámicos. Por otro lado, a los apasionados del mundo de la fotografía o el deporte, se les hace imprescindible el uso de estos robots para poder tomar las mejores instantáneas. Cada vez son más las marcas que desarrollan sus propios robots con cámara incorporada para diferentes usos, principalmente drones, pero también robots terrestres, Figura 1.1.

Figura 1.1: Dron recreativo.

En un principio, los robots requerían un operador humano que controlase sus movimientos. No obstante, el desarrollo de nuevas estrategias y técnicas de control automático ha permitido, en gran cantidad de aplicaciones, moverse de manera autónoma en el entorno. Tanto las restricciones a tener en cuenta en el movimiento, e.g., la distancia mínima de seguridad para evitar colisiones; como la adaptación de la respuesta a entornos generalmente dinámicos suponen todavía en la actualidad un

desafío para la teoría de control.

En aplicaciones como la vigilancia o deporte las estrategias de control se basan en el cálculo de las acciones necesarias para mantener al objetivo a monitorizar en el campo de visión con una perspectiva y encuadre adecuados. Es decir, se trata de un problema de control orientado a la percepción. Las restricciones de visibilidad del sensor se encuentran estrechamente ligadas a la posición del robot, lo que provoca que no todas las configuraciones permitan una observación directa del objeto y que sea necesaria la implementación de un control que permita introducir restricciones de forma dinámica y que sea capaz de trabajar con posición relativa entre el objeto a seguir y el sensor.

Los objetivos del problema de seguimiento pueden ser muy diversos. Hay situaciones en las que es necesario utilizar varios robots para monitorizar a la persona desde varios puntos de vista, por tanto, el control debe ser capaz de evitar los choques y priorizar el movimiento como formación manteniendo al sujeto en el campo de vista [1]. Otras aplicaciones requieren precisión en la posición desde la que se está enfocando al sujeto, por tanto el control deberá planificar la trayectoria sobre zonas que tengan la menor incertidumbre posible [2]. Por otra parte, para alcanzar dichos objetivos existen diferentes tipos de soluciones y metodologías de control. En [1] se aplica un control basado en geometría para calcular las trayectorias de la formación de los robots. Este control es directo y rápido pero se puede hacer más complejo rápidamente si se añaden restricciones en función del entorno. En [3] el control es en imagen utilizando una homografía calculada a partir de un patrón conocido que porta el sujeto a seguir, la imagen es obtenida por una cámara fijada al robot. Es un control robusto, aunque no tiene en cuenta los grados de libertad adicionales que proporcionan dispositivos tipo PTZ. Tanto en [4] como en [5] se aplican dos controles *PID*, uno para la velocidad lineal y otro para la angular, el cual es alimentado por las estimaciones en posición y velocidad relativa robot/persona calculadas mediante un filtro de *Kalman*, en estos dos últimos artículos el seguimiento del sujeto solo tiene en cuenta la distancia relativa y no la orientación entre ambos. Por otro lado, en [6] se propone una estrategia de movimiento basada en el control óptimo donde se busca posicionar a dos drones para seguir el movimiento de una persona, Figura 1.2. En general esta metodología de control resulta adecuada para este tipo de aplicaciones, ya que es capaz de gestionar sistemas MIMO con dinámicas de comportamiento y restricciones de movimiento no lineales.

Por otra parte, la solución propuesta en [6] presenta algunas limitaciones de interés

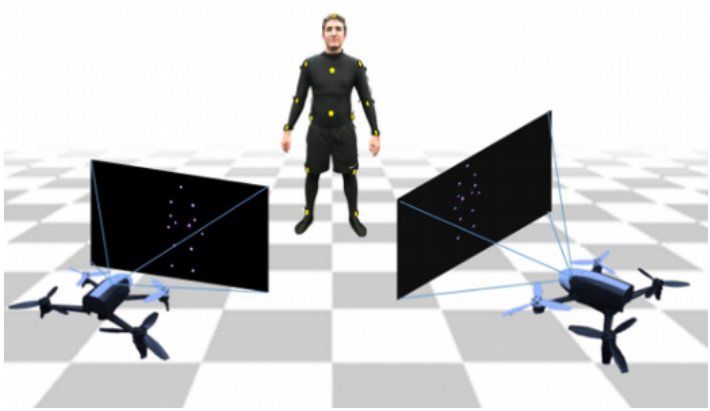


Figura 1.2: Artículo *Flycon* seguimiento con drones.

práctico. En particular, al utilizar drones el problema de visibilidad no es tan acusado como en el caso de robots terrestres debido al desacople que existe en el movimiento de sus ejes. La utilización de esta estrategia con robots con dinámicas de movimiento no holónomas introduce una complejidad adicional en el problema de planificación que limita las trayectorias permitidas para mantener al objetivo en el campo de vista. Esta es la motivación que ha llevado a plantear este TFM, en el cual se busca realizar un seguimiento de un sujeto enfocándolo con un solo robot con cámara más PTZ respecto de una perspectiva deseada aplicando un control óptimo.

1.2. Objetivos y alcance

El objetivo principal de este TFM es desarrollar un algoritmo de control que permita calcular trayectorias óptimas para monitorizar a una persona utilizando un robot móvil equipado con una cámara RGB-D y un PTZ.

Por tanto, para alcanzar el objetivo principal, se plantean los siguientes sub-objetivos:

1. Desarrollar el control que centre a la persona en la imagen con los giros propios del PTZ independientemente de la posición del robot.

2. Proponer una estrategia de control óptimo que calcule y aplique comandos de velocidad al robot terrestre para llegar a la posición relativa deseada respecto a la persona.

3. Evaluar ambos controles en diferentes entornos de complejidad y realismo creciente analizando las prestaciones y la calidad de la monitorización.

(a) *Arbotix turret.* (b) *Realsense.*

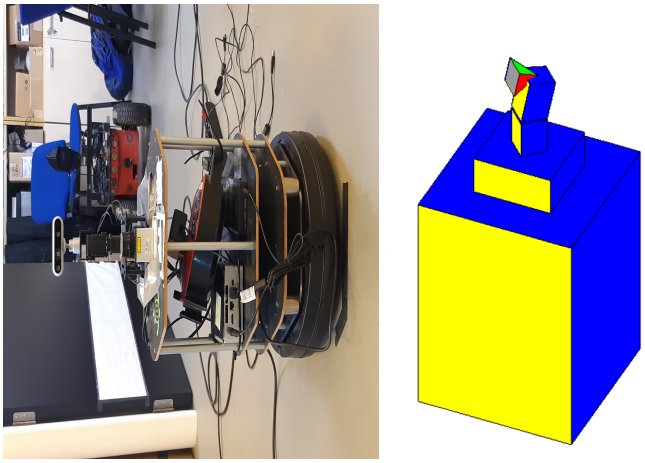
Figura 1.3: PTZ y cámara de profundidad.

Para desarrollar el primer sub-objetivo se ha considerado el PTZ *Arbotix turret*, que tiene dos ángulos de giro, y la cámara *Realsense* con sensor de profundidad, Figura 1.3. Este conjunto se encuentra instalado en la parte superior del robot terrestre, ver Figura 1.4. En el TFM se ha desarrollado un control basado en la geometría del problema debido a la dificultad de calcular los giros del PTZ a partir de su modelo inverso. Utilizando el modelo *pinhole* para la cámara y la profundidad de la escena se calculan los ángulos necesarios que debe girar el PTZ para conseguir que la persona se encuentre en la posición deseada dentro de la imagen. Se ha implementado el software necesario para comunicar los dos dispositivos, cámara y PTZ, a través de ROS, consiguiendo un bucle de control capaz de centrar a la persona en la imagen.

Para lograr el segundo sub-objetivo se ha estudiado *MPC* como método de control óptimo. Este se basa en la minimización de una función de coste para calcular las acciones del robot. En el TFM se han propuesto distintas funciones de coste con el objetivo de observar la influencia que produce en la trayectoria controlar diferentes variables del robot con cámara más PTZ. Todas las variables utilizadas en dicha función son relativas entre la persona y el robot y obtenidas a partir del único sensor disponible, la cámara RGBD.

En cuanto a la consecución del tercer sub-objetivo, en primer lugar se ha realizado un análisis de sensibilidad con el método *Montecarlo* con el objetivo de evaluar el control en diferentes entornos cada vez más realistas utilizando *Matlab*. Se ha implementado *MPC* a través de una toolbox proporcionada por el propio software y se ha construido un entorno de simulación 3D, donde se ha creado un modelo del robot, Figura 1.4(b), el cual se desplaza con el objetivo de enfocar a la persona desde un punto de vista deseado.

En el análisis se han modificado dos tipos de variable. Por un lado internas del robot, como su dinámica y parámetros del control y por otro se han modificado variables del entorno ya sean condiciones iniciales o el tipo de trayectoria del sujeto a seguir. En segundo lugar, como paso previo a la implementación del control en dispositivos reales, se ha utilizado el simulador *Stage* y el visualizador *RViz* propio de *ROS*, para ello se ha modificado el código de *Matlab*, donde solo se calculan las acciones, añadiendo comandos para su comunicación con las dos herramientas anteriores. Finalmente se han realizado experimentos en dispositivos reales implementando parte del código en *ROS* para comunicarse con los dispositivos manteniendo el cálculo del control óptimo en *Matlab*.



(a) Conjunto real. (b) Conjunto simulado.

Figura 1.4: Conjunto del robot con PTZ y cámara.

1.3. Contenidos

La memoria de este TFM se divide en siete capítulos y un anexo. En el Capítulo 1 se resume la motivación y los objetivo de este trabajo. En el Capítulo 2 se desarrolla el modelo matemático de la cámara y el control del PTZ. Seguidamente, en el Capítulo 3 se expone *MPC* como un tipo de control óptimo y se enfoca su utilización al problema de seguimiento de un objetivo móvil con un enfoque deseado. Una vez expuesto *MPC*, en el Capítulo 4 se implementan los controles en *Matlab* y se detallan los primeros resultados de la simulación. A continuación, en el Capítulo 5 se realiza un análisis de sensibilidad a distintos parámetros de nuestro sistema para comprobar su robustez. En el capítulo 6 se exponen los resultados de los experimentos realizados con *Stage*, simulador realista propio de *ROS* y los resultados finales con dispositivos reales. Finalmente en el Capítulo 7, se exponen las conclusiones obtenidas fruto de la realización de este TFM y las líneas

futuras que quedan abiertas. Adicionalmente, se ha incluido un anexo, el cual contiene un desarrollo más amplio del control del PTZ.

Capítulo 2

CONTROL DEL PTZ CON CÁMARA

El primer aspecto que se va a estudiar para implementar un control orientado a la monitorización de la persona es el diseño del movimiento del PTZ. Se comienza con el desarrollo de las ecuaciones que permiten modelar la cámara y obtener la posición 3D de la región de la persona en la imagen. Seguidamente, se desarrollan las ecuaciones geométricas que se aplican para caracterizar el PTZ y dar las órdenes necesarias a sus motores para conseguir que la persona se encuentre centrada en la imagen.

2.1. Formulación del problema

Se considera un entorno 3D en el que se mueve una persona y un robot terrestre con un PTZ con cámara incorporada. De la imagen obtenida a través de la cámara se quiere centrar, con el movimiento del PTZ, un punto perteneciente a la persona cuyas coordenadas en la referencia "mundo" son (X_p, Y_p, Z_p). Con esto se logra mantener a la persona en el campo de vista. Además, para enfocar a la persona con una perspectiva deseada se aplica un control al robot para que se mueva hasta la posición deseada relativa con la persona. La posición del robot en la referencia "mundo" es dada por las coordenadas (X_r, Y_r, Z_r). Debido a que no se dispone ningún de sensor exterior que posicione al robot y a la peronsa respecto de la referencia "mundo", se debe trabajar en coordenadas relativas robot/persona. Por tanto, el punto a centrar de la persona en 3D respecto de la cámara, lo designamos como (X_{pc}, Y_{pc}, Z_{pc}) y la proyección del punto 3D en la imagen como (x, y) o (u, v, s) en coordenadas homogéneas. En la aplicación propuesta se considera suficiente con centrar un punto ya que el objetivo de este control es mantener a la persona en el campo de vista, no enfocarla con un determinado ángulo ya que este último objetivo se logra con el movimiento del robot.

2.2. Modelo de la cámara

La cámara la modelamos como cámara *pinhole*. En esta sección se muestra cómo obtener las coordenadas 3D (X_{pc}, Y_{pc}, Z_{pc}) a partir del correspondiente píxel (x, y) de la imagen, ver Figura 2.1, mediante el modelo de cámara *pinhole* y los parámetros propios de la cámara *Realsense*.

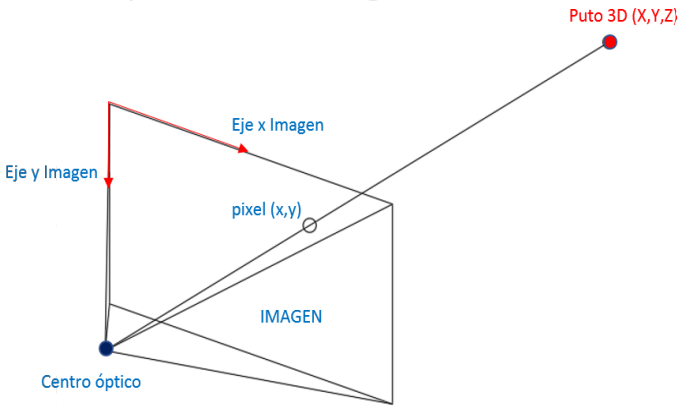


Figura 2.1: Geometría proyectiva.

El modelo de cámara *pinhole* se define a través de tres matrices: la de calibración, la de perspectiva y la de cambio de coordenadas.

La matriz de calibración,

$$\begin{bmatrix} f_x & S & X_0 \\ 0 & f_y & Y_0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{2.1}$$

engloba todos los parámetros propios o intrínsecos de la cámara, ver Figura 2.2:

- f_x y f_y: distancias focales en los ejes x e y, respectivamente.
- X_0 e Y_0, posición del centro óptico respecto de la referencia imagen.
- S o *Skew*, parámetro que cuantifica la falta de ortogonalidad de los píxeles.

Estos parámetros se pueden obtener a través de distintos algoritmos de calibración mediante la toma de distintas fotos a un patrón de calibrado. En nuestro caso el propio fabricante proporciona el valor numérico de todos los parámetros, ver Tabla 2.1.

La matriz de cambio de perspectiva, cuya función es pasar de coordenadas homogéneas en 3D a coordenadas del plano imagen,

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \tag{2.2}$$

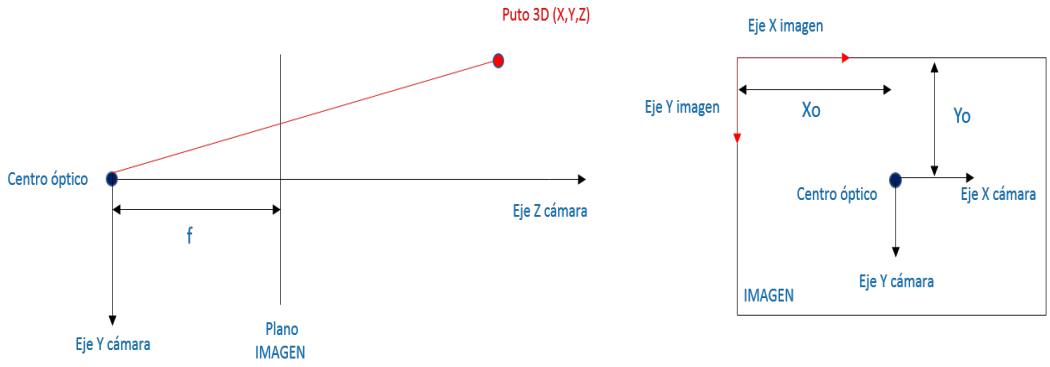


Figura 2.2: Parámetros intrínsecos.

Parámetro	Valor	Unidades
f_x	613	píxeles
f_y	613	píxeles
X_0	322	píxeles
Y_0	250	píxeles
S	0	-

Tabla 2.1: Parámetros intrínsecos de la cámara *Realsense*.

La matriz de cambio de coordenadas,

$$\begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix}, \tag{2.3}$$

compuesta por la matriz de rotación y de traslación, traslada el punto 3D, en coordenadas globales, a la referencia cámara. En nuestro caso la referencia cámara se toma como referencia global ya que no disponemos de ningún sensor exterior que nos indique dónde se encuentra el robot respecto de una referencia "mundo". Por consiguiente, la matriz de rotación es la matriz identidad y el vector de traslación son todo ceros.

Juntando las tres matrices se obtiene la relación final en coordenadas homogéneas entre el punto 3D y las coordenadas del punto en la imagen,

$$\begin{bmatrix} u \\ v \\ s \end{bmatrix} = \begin{bmatrix} f_x & S & X_0 \\ 0 & f_y & Y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} X_{pc} \\ Y_{pc} \\ Z_{pc} \\ 1 \end{bmatrix}. \tag{2.4}$$

Tras operar, y sabiendo que nuestra cámara no tiene *skew*, obtenemos un sistema de tres ecuaciones,

$$u = f_x\, X_{pc} + X_0\, Z_{pc}, \tag{2.5}$$

$$v = f_y\, Y_{pc} + Y_0\, Z_{pc}, \tag{2.6}$$

$$s = Z_{pc}. \tag{2.7}$$

Para obtener el punto 3D (X_{pc}, Y_{pc}, Z_{pc}) a partir del píxel objetivo (x, y) a centrar, dividimos las ecuaciones (2.5), y (2.6) por (2.7) y despejamos X_{pc} e Y_{pc},

$$X_{pc} = \frac{(x - X_0)\, Z_{pc}}{f_x}, \tag{2.8}$$

$$Y_{pc} = \frac{(y - Y_0)\, Z_{pc}}{f_y}, \tag{2.9}$$

con la coordenada Z_{pc} proporcionada directamente por la cámara a través de la imagen de profundidad.

Por tanto, con las ecuaciones (2.8) y (2.9) se obtiene directamente la posición del punto 3D (X_{pc}, Y_{pc}, Z_{pc}) que queremos centrar de la persona, a partir de las coordenadas en píxeles (x, y) de ese punto en la imagen.

2.3. Geometría del PTZ

Una vez calculado el punto 3D de un píxel respecto de la referencia cámara, pasamos a desarrollar la geometría del PTZ con la que obtendremos las relaciones trigonométricas necesarias para centrar el píxel en la imagen a través del movimiento de los servomotores del PTZ, ver Figura 2.3.

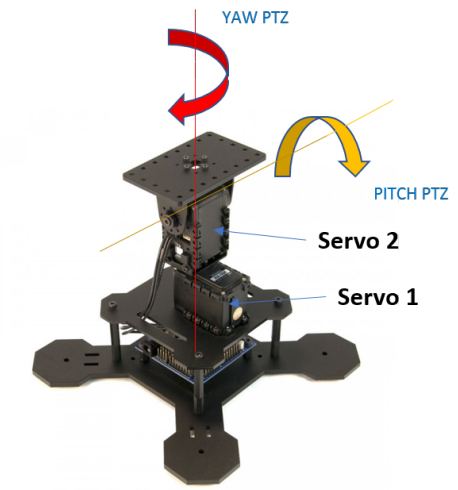


Figura 2.3: PTZ.

El PTZ consta de dos servomotores que permiten posicionar la cámara con los ángulos *pitch* y *yaw* deseados, a los que nos referiremos como θ y ψ respectivamente. En un primer momento se intentó calcular los ángulos de movimiento del PTZ mediante el modelo inverso de Denavit-Hartenberg. Sin embargo, esto no fue posible debido a que el modelo de brazo considerado solo tiene dos grados de libertad. Por tanto, en el TFM se ha optado por una solución basada únicamente en geometría de los ángulos. Así, centraremos primero el píxel en la coordenada x de la imagen mediante el movimiento en ψ del PTZ y, posteriormente, haremos lo mismo en y con el movimiento en θ. Es fundamental ejecutarlo en ese orden puesto que el movimiento en ψ afecta al movimiento en θ, pero no a la inversa, debido al offset del servo 2.

Finalmente, es importante resaltar que ambos ángulos de giro, θ y ψ, son incrementales respecto de la posición actual del PTZ. Además, se supone que el punto 3D que se quiere centrar en la imagen siempre está por delante de la cámara y en su rango de visión.

2.3.1. Centrado en *Yaw*

El objetivo es que el movimiento del servo 1 haga que la coordenada relativa X_{pc} del punto 3D seleccionado sea cero. Para ello se proyecta el punto $(0, Y_{pc}, Z_{pc})$ en el plano (X, Y) del sistema de referencia del servo 1 del PTZ. El ángulo que forma esta proyección con respecto a la coordenada X_{pc}, calculado mediante la función *arctan*, representa el ángulo relativo ψ^* a girar. Para centrar el píxel en la coordenada x de la imagen se necesita considerar dos situaciones: que la cámara enfoque hacia arriba o hacia abajo. En cada una de ellas se diferencian, a su vez, tres zonas en las que puede estar ubicado el punto 3D en función del píxel que queremos centrar, ver Figura 2.4.

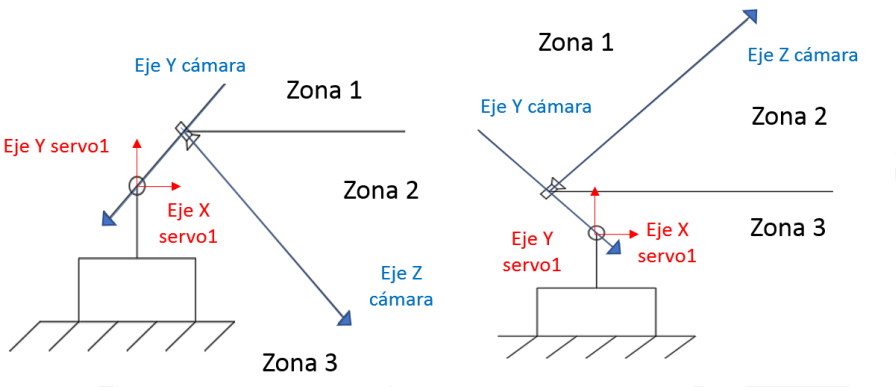


Figura 2.4: Distinción de zonas para centrar con ψ.

En ambas situaciones, las zonas están separadas por el eje z de la cámara y por la

horizontal que pasa por ella. En cada zona se obtienen seis ecuaciones que relacionan la geometría del problema. La mayoría son idénticas para todas las zonas, excepto dos. Por simplicidad, en la memoria se describe únicamente el desarrollo geométrico para obtener el ángulo relativo ψ^* a girar en la *zona 2*, con la configuración "cámara hacia arriba", ver Figura 2.5. En el Anexo A se encuentra el desarrollo de las restantes configuraciones.

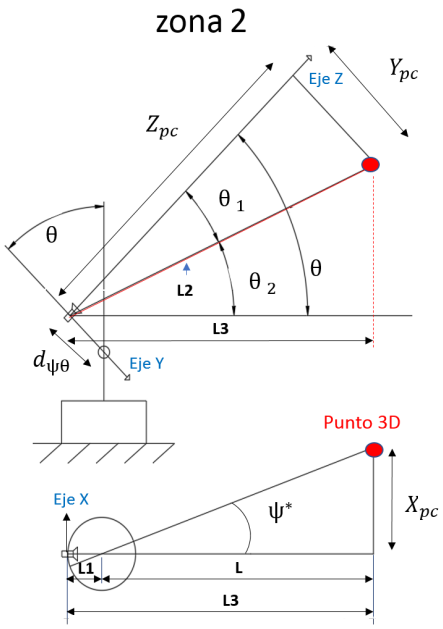


Figura 2.5: Centrado del píxel con ψ. Cámara arriba. Planta y Alzado.

El ángulo relativo a girar por el servo 1, ψ^*, se calcula como,

$$\psi^* = \arctan \frac{X_{pc}}{L}, \tag{2.10}$$

siendo L, la distancia en horizontal desde el punto 3D suprimiendo su componente X_{pc} hasta el eje de giro del servo 1,

$$L = L_3 - L_1. \tag{2.11}$$

La distancia L_1 se calcula utilizando el ángulo de giro actual del servo 2, θ, y la distancia entre los dos ejes de giro, $d_{\theta\psi}$,

$$L_1 = d_{\theta\psi} \cos\theta. \tag{2.12}$$

Por otra parte, L_3 es la proyección de L_2 en el eje X del servo 1

$$L_3 = L_2 \cos\theta_2, \tag{2.13}$$

siendo L_2 la hipotenusa del ángulo formado por las coordenadas (Y_{pc},Z_{pc}) y θ_2,

$$\theta_2 = \theta - \theta_1. \tag{2.14}$$

Por último,

$$\theta_1 = \arctan \frac{|Y_{pc}|}{Z_{pc}} \tag{2.15}$$

es el ángulo formado por Z_{pc} y Y_{pc}.

2.3.2. Centrado en *Pitch*

En esta sección se calcula el ángulo que centra el píxel en la vertical de la imagen (*pitch*) para hacer $Y_{pc} = 0$, suponiendo que el píxel ya está centrado en *yaw* según el procedimiento anterior. De esta manera, se considera que la coordenada 3D X_{pc} del píxel es cero. Además suponemos que el valor de la coordenada Z_{pc} se mantiene antes y después de centrar el píxel con ψ. Al aplicar esta hipótesis en el cálculo del ángulo θ introducimos un pequeño error, ya que dicha coordenada puede cambiar al girar el servo 1. El valor real de Z_{pc} suponiendo ya girado el servo 1 se obtendría realizando dos cambios de coordenadas. Primero se trasladaría el punto 3D a la referencia servo 1 y seguidamente con el giro calculado ψ^* se volvería a trasladar a la referencia cámara. Con esta aproximación evitamos dos cambios de coordenadas, simplificando el análisis geométrico. Los resultados experimentales del Capítulo 6 demuestran que el error provocado por esta aproximación no es significativo. Es importante notar que la hipótesis de que el punto ya está centrado en *yaw* se considera únicamente para realizar los cálculos pero el movimiento de los dos servos del PTZ se ejecutan simultáneamente.

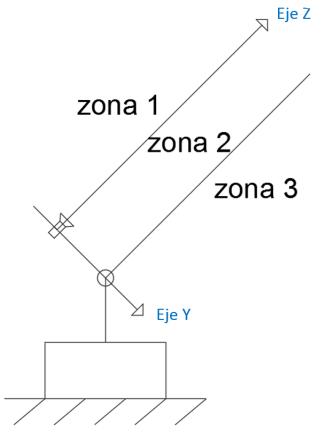


Figura 2.6: Distinción de zonas para centrar con θ.

En este caso, se centra el píxel calculando la recta tangente desde el punto 3D a la circunferencia que genera la cámara al rotar 360º con el giro θ, lo que proporciona dos rectas con distinta pendiente. Según en qué zona se encuentre el punto 3D elegiremos una recta u otra, ver Figura 2.6. La metodología para obtener el ángulo relativo θ^* a girar en la *zona 3* es la siguiente, Figura 2.7.

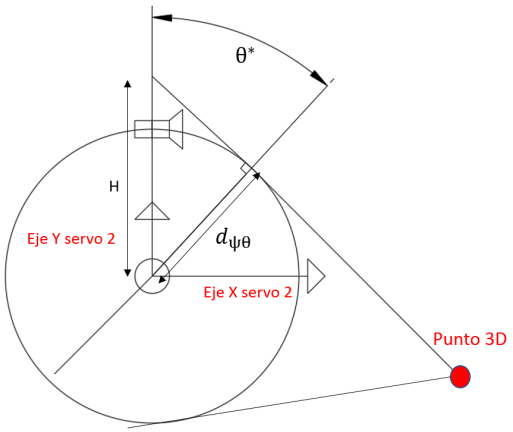


Figura 2.7: Centrado del píxel con θ. Alzado.

El ángulo θ^* relativo a girar se calcula como,

$$\theta^* = \arccos \frac{d_{\theta\psi}}{H}, \tag{2.16}$$

siendo H la distancia desde el eje de giro del servo 2 hasta el corte que genera la recta tangente a la circunferencia con el eje Y de dicho servo. La pendiente de la recta tangente a la circunferencia se calcula resolviendo una ecuación de segundo orden,

$$d^2_{\theta\psi} = \frac{(m + Y - mX)^2}{1 + m^2}, \tag{2.17}$$

siendo m la pendiente de la recta tangente y (X, Y) las coordenadas del punto 3D (Y_{pc}, Z_{pc}) respecto de la cámara trasladadas a la renferencia servo 2, ver Figura 2.8.

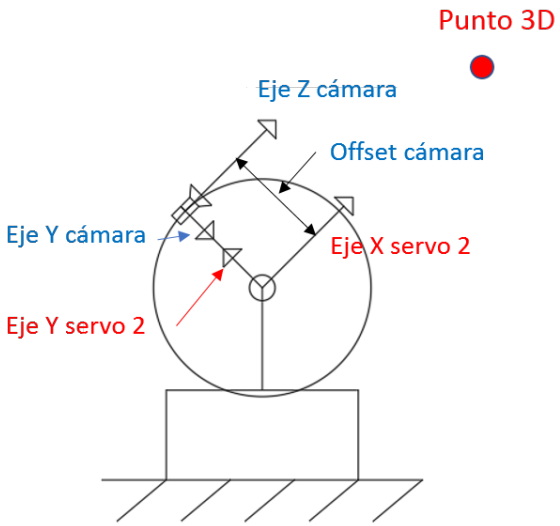


Figura 2.8: Cambio de coordenadas desde cámara a servo2.

Al resolver la ecuación (2.17) de segundo grado se obtienen dos pendientes una por cada recta tangente a la circunferencia, el valor de la pendiente y por tanto la

recta tangente, se escogerá dependiendo de la zona en la que se encuentre el punto 3D. Finalmente con la pendiente y las coordenadas del punto objetivo se obtiene al punto de corte H.

Capítulo 3

CONTROL DEL ROBOT

Una vez que el PTZ es capaz de centrar a la persona y mantenerla en el campo de visión, el siguiente paso es desarrollar un control que permita posicionar al robot en un punto respecto de la persona y mantener dicha posición relativa a lo largo del tiempo. El tipo de control que se ha escogido es el control óptimo. En este capítulo se motiva el uso de este tipo de control y se desarrolla su aplicación en el problema de seguimiento planteado.

3.1. Control óptimo del robot terrestre

Para resolver el problema de control se ha utilizado *MPC* (Model Predictive Control), método particular de resolución del control óptimo. Es un control predictivo que emplea el modelo del sistema para predecir el comportamiento futuro y resolver un problema de optimización seleccionando las mejores acciones en un horizonte de tiempo finito. Además, tiene la capacidad de funcionar como un control *Feed-forward* pudiendo reaccionar mejor ante cambios si se conoce con antelación cómo se va a modificar el sistema a controlar.

Se ha seleccionado *MPC* como método de control ya que permite manejar de manera sencilla sistemas tipo MIMO con interacciones complejas entre las entradas y las salidas. Con controles tradicionales como el *PID* esto sería mucho más complicado de abordar. Además este control maneja bien las no linealidades del problema, de forma que no es necesario trabajar en torno a un punto de linealización para obtener buenos resultados, como es el caso de otro tipo de controles en espacio de estados. A su vez, permite incluir restricciones, cualidad necesaria en el control de trayectorias para evitar colisiones.

En cada iteración del bucle de control *MPC* minimiza en un horizonte de *N-pasos* una función de coste creada por el usuario teniendo en cuenta la dinámica del sistema

y las restricciones impuestas.

La función de coste a minimizar se suele representar como una combinación lineal de diferentes errores, C_k, ponderados según su importancia, p_k,

$$J_k = p_1 C_1(x_k) + p_2 C_2(x_k) + ... + p_n C_n(x_k), \tag{3.1}$$

siendo p_n y C_n valores escalares positivos y x_k las variables de estado del sistema en el instante k. En algunas aplicaciones también se suelen incluir términos en función de las acciones para evitar o favorecer controles más o menos agresivos.

Las acciones del robot en cada paso de simulación, u_k, se obtienen a partir de la solución del problema de optimización para los siguientes N-pasos y aplicando las acciones del primero,

$$\min_{x_k, u_k} \sum_{k=0}^{N-1} J_k, \tag{3.2}$$

sujeto a respetar la dinámica del robot y las restricciones impuestas.

A continuación, se describen en detalle los principales elementos para utilizar *MPC* en nuestro problema de monitorización.

3.1.1. Descripción de la función de coste

En el TFM se ha diseñado una función de coste de tres términos con los que se controla la distancia euclídea robot-persona (ρ), la orientación relativa robot-persona (ϕ) e indirectamente el ángulo *yaw* (ψ) del PTZ,

$$J_k = p_i \cdot C_i(x_k) + p_a \cdot C_a(x_k) + p_e \cdot C_e(x_k). \tag{3.3}$$

Los tres términos desarrollados son los siguientes:

1. Término C_i. Establecemos que la distancia euclídea, entre el robot y la persona en 2D, sea igual a una distancia deseada,

$$C_i = \left(\sqrt{(X_{pr})^2 + (Y_{pr})^2} - \rho_d \right)^2, \tag{3.4}$$

donde (X_{pr}, Y_{pr}) son las coordenadas relativas de la persona respecto del robot y ρ_d la distancia euclídea deseada. Este término ya impone una no linealidad en el problema al incluir la raíz cuadrada del cuadrado de algunas variables de estado. Por otro lado, todos los costes se elevan al cuadrado para que la función de coste sea siempre positiva y tenga como valor mínimo el cero.

2. Término C_a. Fijamos un ángulo deseado entre la persona y el robot, ángulo entre vectores X_r y X_p, ver Figura 3.1,

$$C_a = (\phi_{pr} - \phi_d)^2, \tag{3.5}$$

donde ϕ_{pr} es el ángulo relativo entre la persona y el robot y ϕ_d es el ángulo relativo deseado.

3. Término C_e. Establecemos indirectamente un ángulo deseado al giro ψ del primer servomotor del PTZ.

$$C_e = \left(\underbrace{\arctan(\frac{Y_{pr}}{X_{pr}})}_{\psi} - \psi_d \right)^2, \tag{3.6}$$

siendo ψ_d el ángulo deseado del giro perteneciente a *yaw*. El ángulo ψ es controlado por el PTZ para centrar a la persona en la imagen. Sin embargo, indirectamente su valor depende de la posición de la persona respecto del robot a través de la función *arctan*. En la Figura 3.1 se muestra al robot enfocando a la persona con cuatro perspectivas distintas, para todas ellas, independientemente de lo que haga el robot, el PTZ ha girado el ángulo ψ necesario para centrarla en la imagen pero dicho ángulo depende del posicionamiento relativo.

En principio, se puede pensar que la ecuación (3.6) se puede calcular simplemente restando al ángulo ψ real y el ángulo ψ_d deseado. No obstante, si se plantea de esta forma, se obtiene un término de coste que no depende de las variables de decisión del problema de optimización, x_k, u_k, por lo que su presencia en MPC sería irrelevante. Esto es debido al desacople del control planteado para ambos dispositivos, PTZ y robot.

La Figura 3.1 muestra un resumen de las implicaciones de cada término de la función de coste en el problema de optimización. Con el término C_i el robot se puede posicionar sobre la circunferencia de color negro, de radio igual a ρ_d, con cualquier orientación. Añadiendo el segundo término C_a, la orientación debe ser la deseada, ϕ_d, entre el vector X_p y X_r. En este supuesto el ángulo deseado entre la persona y el robot es de cero grados. Se han dibujado cuatro posibles casos de los infinitos que existen. Finalmente, si establecemos un ψ_d de cero grados solo existe una posible solución que dé valor cero a los tres términos.

Señalar que el control del PTZ y del robot se encuentran desacoplados. Aunque ambos están relacionados a través del ángulo ψ, ángulo controlado por el PTZ pero

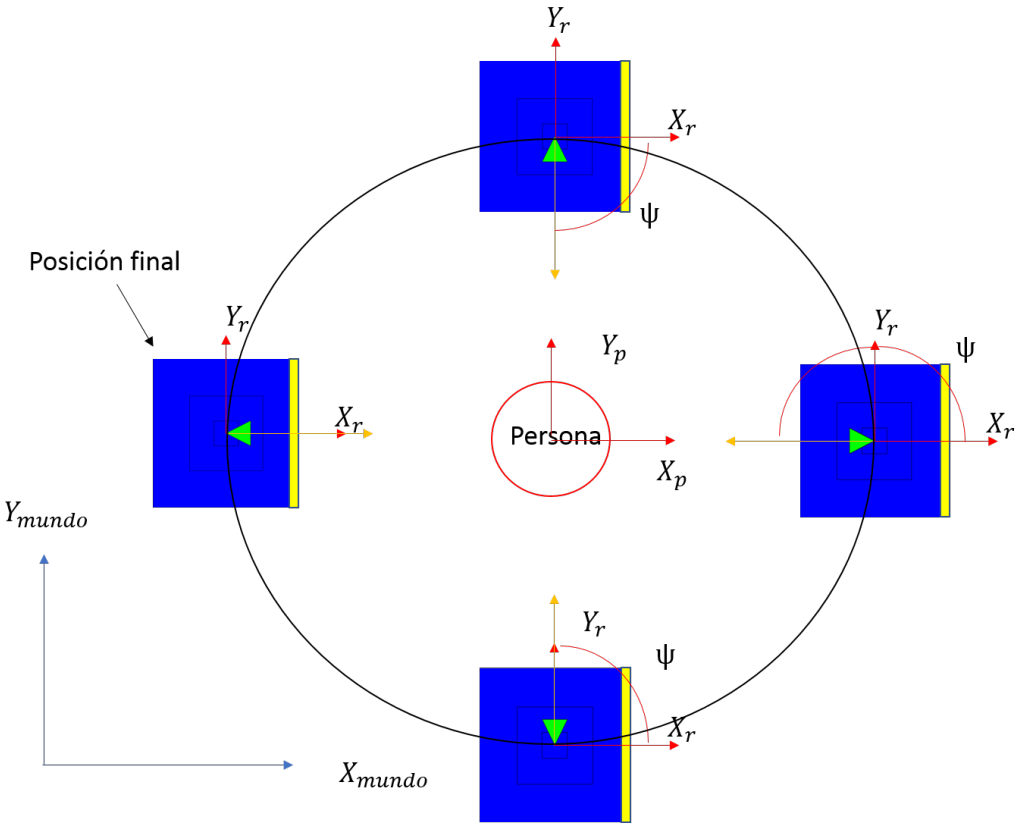


Figura 3.1: Representación gráfica del posicionamiento del robot respecto a la persona.

dependiente de la posición relativa del robot/persona, el control del PTZ va a centrar a la persona en la imagen independientemente de lo que haga el robot. Por ejemplo, si se quiere mantener a la persona en otro cuadrante de la imagen, el control del robot se mantendrá idéntico y solo el del PTZ se verá afectado. Ello implica una reducción de la complejidad de ambos controles. Por otro lado, tal y como hemos creado los controles, no es posible aplicar un control basado únicamente en imagen, principalmente porque el control del PTZ se focaliza en centrar un solo punto, cuando es necesario un número mayor según el tipo de control en imagen. Además el control debería englobar todas las variables de estado del PTZ y del robot.

De la misma manera que se incluye un término en la función de coste asociado al *yaw* del PTZ, se podría plantear la inclusión de otro término asociado al *pitch*. Este término podría ejercer de sustituto (o de "contrapeso") al término asociado a la distancia, C_i, expresándose como

$$C_i = (\arctan(\frac{Z_{pr} - \triangle}{\sqrt{(X_{pr})^2 + (Y_{pr})^2}}) - \theta_d)^2, \tag{3.7}$$

siendo θ_d el ángulo *pitch* deseado del servo 2 y $\triangle$

$$\triangle = h_r + \frac{d_{\theta\psi}}{\cos(\theta)}, \tag{3.8}$$

donde h_r es la altura del robot más la altura del servo 1, ver Figura 3.2. La fuerte no linealidad de la ecuación (3.7), comparada con (3.4) hizo que finalmente se descartara esta alternativa.

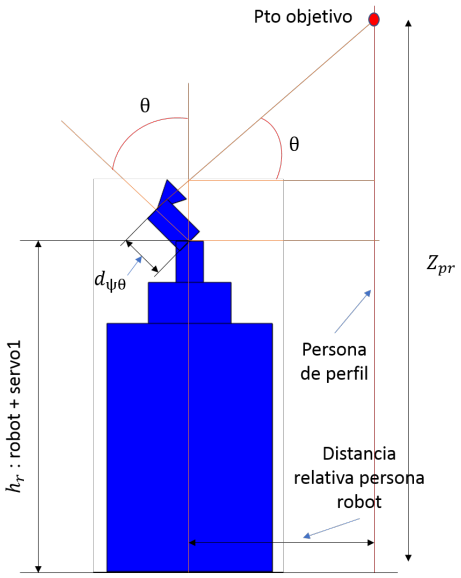


Figura 3.2: Coste C_i con θ.

3.2. Dinámica del robot

En el TFM se ha considerado principalmente una dinámica no holónoma, típica de robots terrestres. La ecuación diferencial que modela este comportamiento es,

$$\begin{bmatrix} \dot{X}_r \\ \dot{Y}_r \\ \dot{\phi}_r \end{bmatrix} = \begin{bmatrix} V_r \cos(\phi_r) \\ V_r \sin(\phi_r) \\ W_r \end{bmatrix}, \tag{3.9}$$

que discretizada mediante la aproximación de Euler, para su uso dentro de MPC, queda de la forma

$$\begin{bmatrix} X_r \\ Y_r \\ \phi_r \end{bmatrix}_{k+1} = \begin{bmatrix} X_r \\ Y_r \\ \phi_r \end{bmatrix}_{k} + \triangle_t \begin{bmatrix} \cos(\phi_r) & 0 \\ \sin(\phi_r) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} V_r \\ W_r \end{bmatrix}. \tag{3.10}$$

donde $\triangle_t$ es el periodo de muestreo, X_r, Y_r y ϕ_r son la posición y orientación del robot y Vr y Wr las acciones, es decir, la velocidad lineal en un eje y la velocidad angular, ver Figura 3.3.

Con el objetivo de analizar la influencia que tiene la no linealidad de la dinámica planteada en el control MPC, también se ha implementado como alternativa una

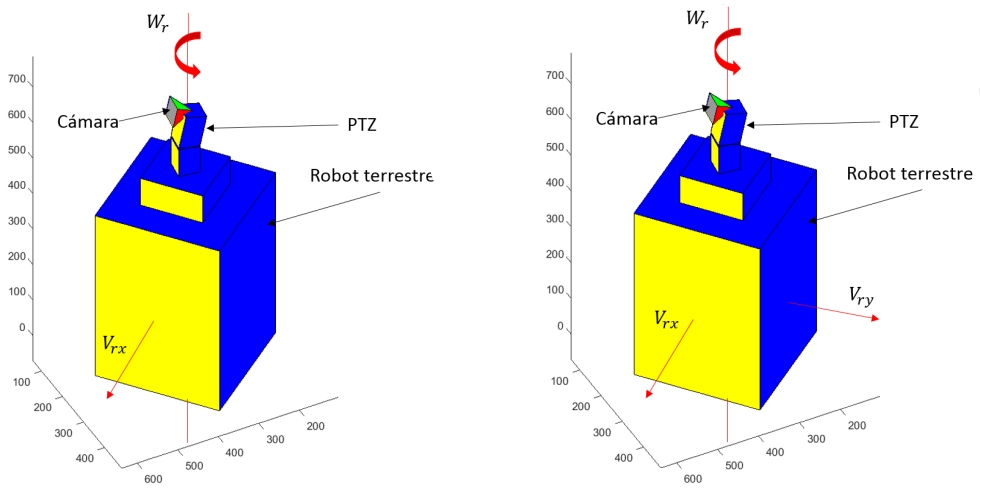
dinámica holónoma sencilla, que permite el movimiento desacoplado en el eje X e Y, ver Figura 3.3 (b). La ecuación diferencial en este caso es

$$\begin{bmatrix} \dot{X}_r \\ \dot{Y}_r \\ \dot{\phi}_r \end{bmatrix} = \begin{bmatrix} V_{rx} \\ V_{ry} \\ W_r \end{bmatrix}, \tag{3.11}$$

y una vez discretizada,

$$\begin{bmatrix} X_r \\ Y_r \\ \phi_r \end{bmatrix}_{k+1} = \begin{bmatrix} X_r \\ Y_r \\ \phi_r \end{bmatrix}_{k} + \triangle_t \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} V_{rx} \\ V_{ry} \\ W_r \end{bmatrix}, \tag{3.12}$$

donde V_{rx} y V_{ry} son las velocidades lineales en ambos ejes. Esta dinámica introduce una variable de decisión adicional en el problema de optimización, pero a cambio es lineal.



(a) Robot noholonomo. (b) Robot holonomo.

Figura 3.3: Modelado del robot terrestre.

3.3. Algoritmo de control

Para finalizar, se expone de forma general el algoritmo completo de control utilizado para el seguimiento de la persona, Algoritmo 1. Se engloba tanto el control del PTZ como el del robot terrestre. El bucle de control simplificado es el siguiente:

Algorithm 1 Algoritmo de control

```
Inicializar MPC (dinámica, función de coste J)
repeat
    Capturar imagen.
    Calcular las coordenadas 3D del píxel a centrar en referencia cámara.
    Transformar el pto 3D en referencia cámara a la referencia del robot.
    Calcular las acciones del robot con MPC.
    Calcular acciones PTZ mediante geometría.
    Aplicar las acciones.
until forever
```

La metodología que se ha seguido en el bucle es la siguiente:

1. Mediante el modelo de cámara pinhole y la coordenada Z obtenida de la imagen de profundidad, se calculan las coordenadas 3D del punto a centrar respecto de la cámara.

2. Con el estado del PTZ se trasladan las coordenadas del punto 3D de la referencia cámara a la referencia robot.

3. Se calculan las acciones del robot utilizando *MPC* para enfocar a la persona con un ángulo deseado.

4. Se calculan las acciones del PTZ para centrar el píxel en la imagen y mantener a la persona en el campo de visión.

5. Se aplican las acciones tanto en el robot como en el PTZ.

Capítulo 4

IMPLEMENTACIÓN EN *MATLAB*

En este capítulo se desarrolla la implementación del control *MPC* planteado en el capítulo anterior en el software *Matlab* donde se pueden abstraer los aspectos de percepción sin afectar a la propuesta de control.

4.1. *MPC* en *Matlab*

El software *Matlab* incluye una *toolbox* que permite implementar el control *MPC*. Los parámetros de implementación del control óptimo son los siguientes:

1. Tf: horizonte temporal durante el que se tiene que alcanzar la posición cuya función de coste sea cero. En nuestro caso es igual a un segundo.

2. $ControlHorizon$: pasos en los que dividimos el horizonte temporal, N. Se le asignan 10 pasos.

3. $StateFcn$: dinámica de la planta a controlar, en nuestro caso el robot, implementado con movimiento holónomo y no holónomo.

4. $OutputFcn$: indica las salidas de la planta del sistema. Se mide todo el estado, tanto posición como orientación.

5. $CustomCostFcn$: establece la función de coste a minimizar.

6. $CustomIneqConFcn$: indica las restricciones a partir de inecuaciones. Con este tipo de restricciones se impide, por ejemplo, que el robot se acerque a menos de una determinada distancia de la persona o que se apliquen velocidades por encima del límite físico del robot.

7. $MVInterpolationOrder$, $Stepolerance$: parámetros del algoritmo de optimización para determinar su convergencia.

4.1.1. Implementación de la función de coste

Debido al problema de atrapamiento en mínimos locales propio de la optimización se ha implementado la función de coste por partes, ecuación (4.1). Esta ecuación es análoga a (3.2) pero engloba dos funciones de coste J_k y J_N con distinto horizonte temporal,6

$$\min_{x} \sum_{k=0}^{N-1}((1-\alpha)J_k) + \alpha J_N, \tag{4.1}$$

siendo $\alpha = 0$ cuando el robot se encuentra "cerca"del punto final y $\alpha = 1$ en el caso contrario. El sumatorio de J_k tiene en cuenta el coste durante toda la trayectoria, lo que proporciona una mejor aproximación al punto final y un menor error durante el seguimiento. Sin embargo se ha comprobado que, si el punto final está lejos de la posición inicial, MPC es incapaz de evitar quedar atrapado en regiones de mínimos locales al minimizar la función de coste. Por ello hasta que no se ha alcanzado una región relativamente cercana al punto final, la función de coste solo ha tenido en cuenta el coste de la posición final J_N. Que el robot esté "lejos" o "cerca" se decide con los errores en los términos referentes a los ángulos ϕ y ψ de la función de coste. Si ambos errores están al mismo tiempo por debajo de un umbral se considera que el robot está suficientemente cerca del punto objetivo para poder evitar atrapamientos en mínimos locales. En la Figura 4.1 se muestra el problema de atrapamiento en mínimo local cuando solo se utilizan los términos J_k para la minimización de la función de coste. Al robot se le ha dado la consigna de mirar a la persona desde delante y éste comienza a cuatro metros por detrás. Como el punto objetivo se encuentra lejos de su posición inicial, MPC se estanca en un mínimo local cuando está mirando a la persona de perfil.

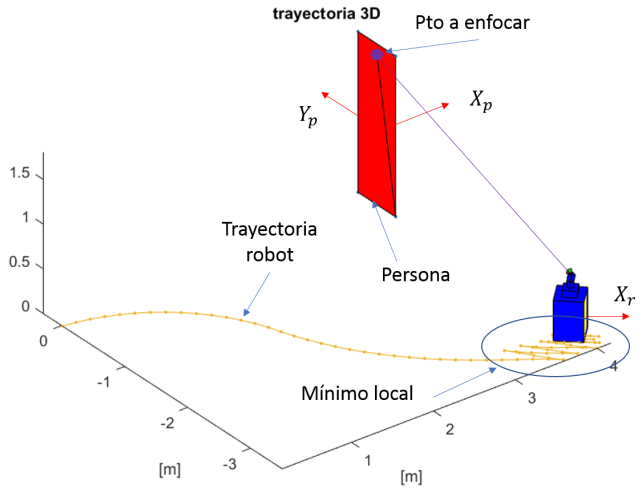


Figura 4.1: Mínimo local.

4.1.2. Consignas de los parámetros deseados

La manera en que el robot se posiciona o sigue a la persona depende de los valores que se asignan a los parámetros ρ_d, ϕ_d y ψ_d de la función de coste.

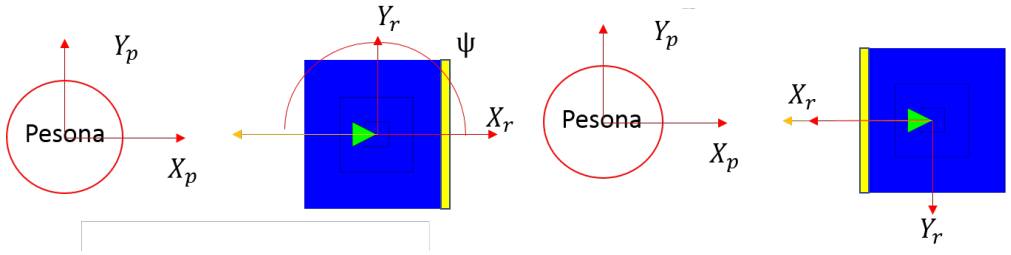
Al tratarse de un robot con dinámica no holónoma y con el fin de que la trayectoria sea lo más natural posible, es adecuado que el vector X_r esté alineado con la dirección en la que se desplaza la persona X_p, es decir, $\phi_d = 0$. Con ello se evitan maniobras cada vez que la persona realiza algún movimiento. La distancia relativa entre el robot y la persona va a depender del objetivo del seguimiento. En principio se ha fijado $\rho_d = 3[m]$, distancia suficiente para evitar colisiones ante cambios bruscos de dirección de la persona. Finalmente, con el ángulo ψ_d se le da la consigna al robot de que enfoque a la persona desde diferentes ángulos. A modo de ejemplo:

1. Desde detrás: $\psi_d = 0[^o]$.
2. Perfil derecho: $\psi_d = 90[^o]$.
3. Perfil izquierdo: $\psi_d = -90[^o]$.
4. Desde delante: $\psi_d = 180[^o]$.

Enfocar a la persona desde delante con la configuración $\psi_d = 180[^o]$ y $\phi_d = 0[^o]$ no es posible, ver Figura 4.2 por dos motivos: una restricción física y una discontinuidad matemática:

1. El PTZ físico no puede girar $180[^o]$ debido a la torsión que se genera en los cables.
2. El término C_e contiene la función, $\arctan(\frac{Y_{pr}}{X_{pr}})$, que en el punto de trabajo presenta una discontinuidad. La coordenada Y_{pr} puede cambiar de signo debido a oscilaciones en la trayectoria, y X_{pr} siempre va a ser negativo si el robot se encuentra delante de la persona, lo que provoca que el resultado en dos pasos de simulación consecutivos pueda tener el mismo valor pero de signo contrario.

Una forma de resolver este problema es que el robot se mueva de espaldas, es decir, $\psi_d = 0[^o]$ y $\phi_d = 180[^o]$.



(a) Ángulo $\phi_d = 0[^o]$. (b) Ángulo $\phi_d = 180[^o]$.

Figura 4.2: Seguimiento enfocando de frente.

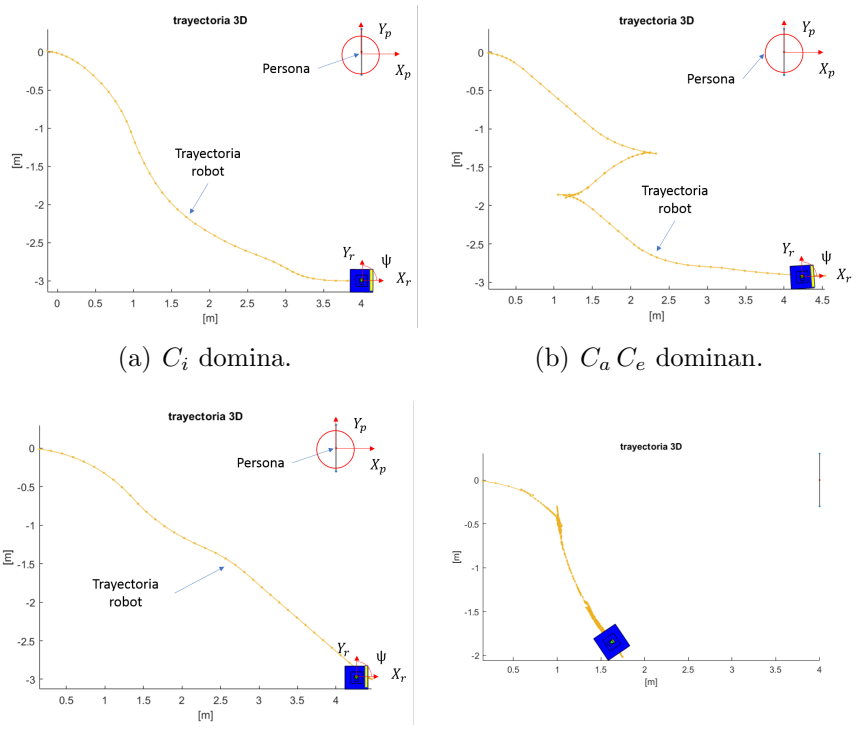
4.1.3. Valor de los pesos en la función de coste

Por último, es necesario proporcionar valores a los pesos de la función de coste. Unos valores dispares en los pesos originan trayectorias muy diferentes e incluso soluciones que no convergen al punto objetivo desde el cual se desea enfocar a la persona.

Tras realizar diferentes pruebas, se ha detectado que unos pesos que hagan que los términos tengan la misma magnitud proporcionan trayectorias naturales, es decir, sin excesivas maniobras. Al introducir términos con distintas unidades en la función de coste es importante utilizar pesos que homogeneicen el valor de los errores. Por el contrario, si hay un término dominante en la función de coste, el control prioriza su reducción obviando los demás, lo que al final provoca que, debido a las restricciones de movilidad de la dinámica no holónoma, se realicen muchas más maniobras. Para ilustrar la influencia de los pesos en la trayectoria se han realizado distintas simulaciones con el mismo valor en los parámetros deseados, $\rho_d = 3[m]$, $\phi_d = 0[^o]$, $\psi_d = 90[^o]$, pero con distintos pesos. Ver Figura 4.3. Para homogeneizar los distintos términos hemos escogido como valores "estandar" de los pesos:

- Para J_k: $p_i = 1$, $p_a = 1 \cdot 10^6$, $p_e = 1 \cdot 10^6$.
- Para J_N: $p_i = 1$, $p_a = 1 \cdot 10^6$, $p_e = 1 \cdot 10^5$

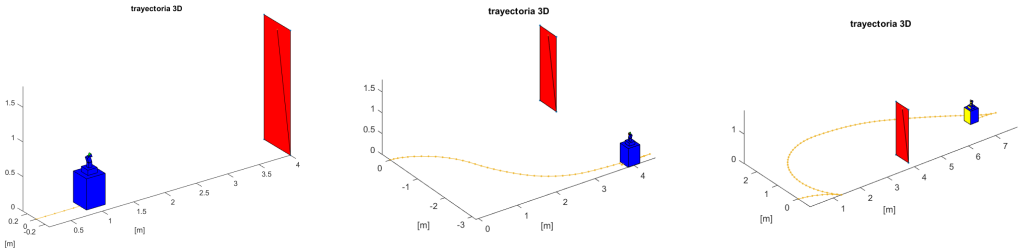
En la Figura 4.4 se muestra un ejemplo de varias simulaciones, donde al robot se le ha indicado que enfoque a la persona desde tres perspectivas con los valores "estandar" de los pesos.



(a) C_i domina. (b) $C_a\,C_e$ dominan.

(c) C_a domina. (d) C_e domina.

Figura 4.3: Influencia de los pesos en la trayectoria.



(a) Desde detrás. (b) Perfil derecho. (c) Desde delante.

Figura 4.4: Trayectorias con pesos "estandar".

Si la persona ya se encuentra de espaldas al robot, cuando a éste se le ordena mirarla desde atrás, su movimiento es rectilíneo hasta reducir la distancia entre ambos a la distancia deseada. Con las mismas condiciones iniciales, si el objetivo es mirarla desde el perfil derecho, el robot realizará un cuarto de vuelta. Por último, si se le ordena enfocarla desde delante, el robot comenzará efectuando una maniobra para cambiar de dirección y realizar la media vuelta desplazándose hacia atrás.

Capítulo 5

ANÁLISIS EMPÍRICO

En el capítulo anterior las simulaciones se han realizado con idénticas condiciones iniciales de dinámica del sujeto, orientación y distancia relativa inicial: la persona está inmóvil, de espaldas al robot y a cuatro metros de distancia. Además, en dichas simulaciones se han asignado a los pesos de la función de coste unos valores fijos. En este capítulo, en cambio, se van a variar esas condiciones así como el valor de los pesos. Se realizará un análisis de sensibilidad para apreciar su influencia en el resultado del control óptimo, es decir, en las acciones y en la trayectoria del robot. El objetivo del análisis es discernir qué conjunto de valores asignados a los pesos es el óptimo para un mayor número de escenarios posibles.

5.1. Diseño de los experimentos

El robot comenzará siempre mirando hacia la persona, pero sin necesidad de enfocarla directamente. Para cada combinación de parámetros habrá cuatro puntos objetivo desde donde queremos que el robot mire a la persona:

1. Enfocar a la persona desde atrás: $\rho_d = 3[m]$, $\phi_d = 0[^o]$ y $\psi_d = 0[^o]$.
2. Enfocar el perfil derecho de la persona: $\rho_d = 3[m]$, $\phi_d = 0[^o]$ y $\psi_d = 90[^o]$.
3. Enfocar el perfil izquierdo de la persona: $\rho_d = 3[m]$, $\phi_d = 0[^o]$ y $\psi_d = -90[^o]$.
4. Enfocar a la persona desde delante: $\rho_d = 3[m]$, $\phi_d = 180[^o]$, $\psi_d = 0[^o]$.

Para realizar el análisis de sensibilidad se han escogido cuatro parámetros:

1. Movimiento de la persona en el espacio. Los movimientos son tres, de menor a mayor complejidad, para observar cómo responde el control ante el seguimiento de un sujeto con diferentes dinámicas:
   - Persona inmóvil.

- Movimiento a velocidad constante, 2.7[km/h], en linea recta.
- Movimiento a velocidad constante, 2.7[km/h], con giros.

2. Orientación inicial de la persona respecto del robot. Las diferentes orientaciones relativas iniciales permiten acercar o alejar los puntos objetivos de la posición inicial del robot, ya que el resultado es distinto si, para enfocar a la persona desde el mismo punto objetivo, se necesita recorrer un cuarto de circunferencia o ir en línea recta. Se consideran los siguientes valores:

   - De espaldas: entre X_r y X_p 0[º].
   - De perfil: entre X_r y X_p 90[º].
   - De frente: entre X_r y X_p 180[º].

3. Distancia relativa inicial. Reduciendo o ampliando la distancia inicial se proporciona más o menos espacio al robot para maniobrar teniendo en cuenta la restricción de que no puede acercarse a menos de 1.5[m] de la persona:

   - 4 [m].
   - 2 [m].

4. Valores de los pesos que multiplican en la función de coste. Tomando como referencia los valores asignados en el Capítulo 4, se crean cinco configuraciones de valores; en cada grupo, a uno o varios pesos se les asigna un valor superior en dos magnitudes al valor de referencia. Ver Tabla 5.1.

   - Configuración 1 (Equilibrados): se utilizan los valores de referencia del Capítulo 4, que hacen que los términos tengan la misma magnitud.
   - Configuración 2 (Distancia): se le proporciona más peso al primer término, ρ, y por tanto a la distancia relativa robot/persona. Se aumenta p_i.
   - Configuración 3 (Alineación): se le proporciona más peso al segundo término, ϕ, y por tanto a que el robot tenga la misma dirección que la persona. Se aumenta p_a.
   - Configuración 4 (Ángulo PTZ): se le proporciona más peso al tercer término, ψ, y por tanto, a que el servo 1 tenga el ángulo deseado. Se aumenta p_e.
   - Configuración 5 (Ángulos): se le proporciona más peso a los términos de los ángulos, ϕ y ψ, con respecto a la distancia. Se aumenta p_a y p_e .

Peso	Equilibrados	Distancia	Alineación	Ángulo PTZ	Ángulos
$p_i\ N$	$1 \cdot 10^0$	$1 \cdot 10^2$	$1 \cdot 10^0$	$1 \cdot 10^0$	$1 \cdot 10^0$
$p_a\ N$	$1 \cdot 10^6$	$1 \cdot 10^6$	$1 \cdot 10^8$	$1 \cdot 10^6$	$1 \cdot 10^8$
$p_e\ N$	$1 \cdot 10^6$	$1 \cdot 10^6$	$1 \cdot 10^6$	$1 \cdot 10^8$	$1 \cdot 10^8$
$p_i\ k$	$1 \cdot 10^0$	$1 \cdot 10^2$	$1 \cdot 10^0$	$1 \cdot 10^0$	$1 \cdot 10^0$
$p_a\ k$	$1 \cdot 10^6$	$1 \cdot 10^6$	$1 \cdot 10^8$	$1 \cdot 10^6$	$1 \cdot 10^8$
$p_e\ k$	$1 \cdot 10^5$	$1 \cdot 10^5$	$1 \cdot 10^5$	$1 \cdot 10^7$	$1 \cdot 10^7$

Tabla 5.1: Configuraciones de valores en los pesos para el análisis de sensibilidad.

El tiempo de simulación varía en función de la dinámica de la persona, siendo de 250 [s] si ésta se mantiene inmóvil o se desplaza en línea recta y de 450[s] si se desplaza realizando giros.

Sumando todas las combinaciones de los parámetros, más los cuatro puntos objetivo, obtenemos un total de 360 simulaciones. Adicionalmente, se implementará la dinámica holónoma en ciertas simulaciones para ver la influencia que supone un movimiento con menos restricciones en los resultados.

5.2. Métricas del análisis

El objetivo es analizar que combinación de pesos permite obtener los mejores resultados en el mayor número de situaciones posibles. Que un conjunto de valores sea "bueno" o "malo" depende de lo "suave" y "natural" que sea la trayectoria, y esa cualidad depende de parámetros tales como el error en régimen permanente, la magnitud de las aceleraciones del robot, etc.

En función de la dinámica de la persona vamos a considerar distintos indicadores para poder establecer comparaciones. Si la persona está quieta interesa más comparar valores absolutos, en cambio, si ésta se desplaza, y con el fin de tener una visión global de un valor durante la trayectoria, hemos creído oportuno comparar con porcentajes. Los índices a comparar son los siguientes:

1. Sujeto inmóvil.
   - T_{est}: tiempo que tarda el robot hasta que alcanza su posición final con velocidad nula.
   - $\int A_{lin}$: integral de la aceleración lineal en valor absoluto.
   - $\int A_{ang}$: integral de la aceleración angular en valor absoluto.
2. Sujeto en movimiento.

- $\%P_c$: porcentaje de tiempo durante el cual el robot está enfocando a la persona desde el lugar correcto, con un umbral de tolerancia del 10 % con respecto al error inicial en distancia y de $\pm 10[^o]$ relativo a ϕ_d y ψ_d.
- $\%A_{lin}$: porcentaje de tiempo durante el cual el robot aplica una aceleración lineal por debajo de un umbral, 3 $[m/s^2]$.
- $\%A_{ang}$: porcentaje de tiempo durante el cual el robot aplica una aceleración angular por debajo de un umbral, 1.57 $[rad/s^2]$.

Si el sujeto está inmóvil, interesa que los indicadores tengan valores bajos ya que supondrá que el robot alcanza la posición final rápidamente y con una trayectoria "suave" al no haber picos en las aceleraciones angular y lineal. Por el contrario, si el sujeto se desplaza durante la simulación, valores altos de los indicadores supondrán que el robot enfoca a la persona con un ángulo correcto durante la mayor parte del tiempo respetando los umbrales en las aceleraciones.

5.3. Resultados del robot noholónomo

En este apartado se utiliza la dinámica noholónoma para realizar las simulaciones. En la memoria se describen los resultados de los experimentos más representativos y se resumen las principales conclusiones. Por ello si no se indica lo contrario, las condiciones iniciales en distancia y orientación se fijan en 4[m] y 0[º] (robot detrás), respectivamente. Y lo que varía son las configuraciones de los pesos y los puntos desde donde se enfoca a la persona.

5.3.1. Comparación con la persona: inmóvil/movimiento rectilíneo

En este apartado se agrupan y resumen los principales resultados obtenidos con la persona inmóvil y desplazándose en línea recta con una velocidad constante de 2.7[km/h]. En la Tabla 5.2 se muestran, para cada combinación, omitiendo la distinción entre perfil izquierdo y perfil derecho puesto que arrojan los mismos resultados, los indicadores utilizados para la persona inmóvil. El apunte NaN significa que el robot no ha alcanzado el régimen permanente una vez terminada la simulación. En la Tabla 5.3 se muestran los indicadores escogidos para evaluar el seguimiento de un sujeto desplazándose. Como en el caso anterior y por el mismo motivo, obviamos la distinción entre los perfiles izquierdo y derecho. Del análisis de las tablas extraemos las siguientes conclusiones.

Pto objetivo	Configuración pesos	T_{est} [s]	$\int A_{lin}\ [m/s^2]$	$\int A_{ang}\ [rad/s^2]$
Detrás	TODOS	0.8	1.5	0
Derecha	Equilibrados	7.6	49.2	12.8
	Distancia	NaN	-	-
	Alineación	5	6.4	2.7
	Ángulo PTZ	5.9	21.3	13.2
	Ángulos	4.6	4.5	5.4
Delante	Equilibrados	8.5	7.5	3.4
	Distancia	NaN	-	-
	Alineación	8.7	7.9	3.2
	Ángulo PTZ	NaN	-	-
	Ángulos	NaN	-	-

Tabla 5.2: Sujeto inmóvil y robot noholonomo.

Pto objetivo	Configuración pesos	$\%P_c$	$\%A_{lin}$	$\%A_{ang}$
Detrás	TODOS	94.8	99.60	100.00
Derecha	Equilibrados	77.7	96.41	92.03
	Distancia	71.3	96.81	92.03
	Alineación	NaN	99.20	94.82
	Ángulo PTZ	53	86.45	75.69
	Ángulos	76.9	99.60	97.21
Delante	Equilibrados	44.2	98.40	98.41
	Distancia	NaN	80.48	76.49
	Alineación	42.6	98.41	98.01
	Ángulo PTZ	NaN	94.02	91.23
	Ángulo	NaN	93.62	92.83

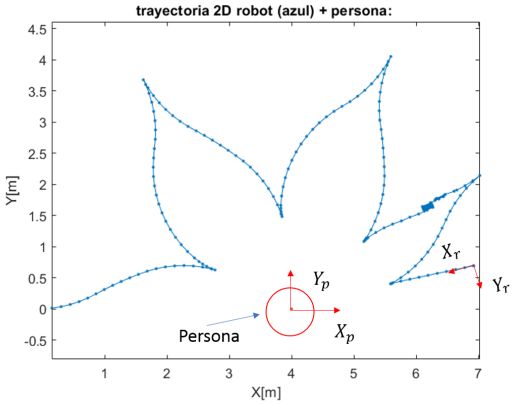
Tabla 5.3: Sujeto con movimiento rectilíneo y robot noholonomo.

Como el punto de partida del robot se encuentra detrás de la persona, todos los controles proporcionan resultados idénticos en el caso de que el punto objetivo sea "detrás". De los tres términos de la función de coste solo el primero, C_i, es distinto de cero durante la simulación y su minimización únicamente provoca un desplazamiento en línea recta sin ningún tipo de maniobra o curva en la trayectoria.

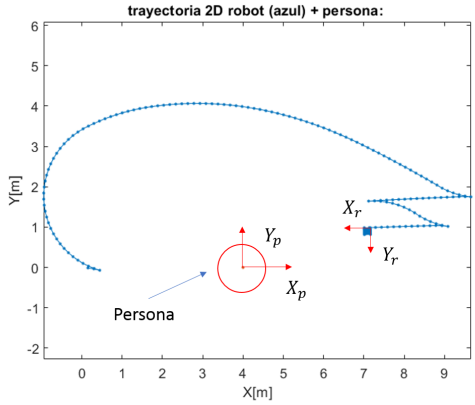
La configuración "Distancia" es la que menos puntos objetivo consigue alcanzar con la persona inmóvil y presenta peores resultados si la persona se desplaza. Por lo general, con la dinámica noholónoma, un peso excesivo del error en la distancia relativa suele provocar un mayor número de maniobras durante la trayectoria debido a que se reduce la distancia rápidamente hasta la deseada sin tener en cuenta el ángulo con el que se desea enfocar a la persona.

Las configuraciones "Ángulo PTZ" y "Ángulos" presentan problemas para el punto

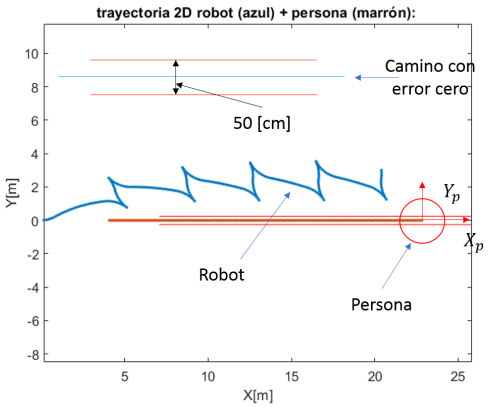
objetivo "Delante" con la persona inmóvil, ver Figura 5.1 (a) y (b). Con la primera configuración, la trayectoria es escalonada y en abanico. El robot se quiere dar la vuelta para reducir el error en ϕ, $\phi_0 = 0[^o]$ $\phi_d = 180[^o]$, pero el termino de ψ obliga a hacerlo en forma de escalones al tener mayor peso y exigir que se mantenga el ángulo ψ igual a cero, que es su valor deseado. En cambio, en la segunda configuración los escalones se producen solo al final, cuando los errores de ambos ángulos están próximos a cero puesto que los pesos hacen que los términos tengan magnitudes parecidas. El robot, con estas dos configuraciones, tampoco logra realizar el seguimiento de la persona desde delante, ver Figura 5.1 (c).



(a) Configuración "Ángulo PTZ".



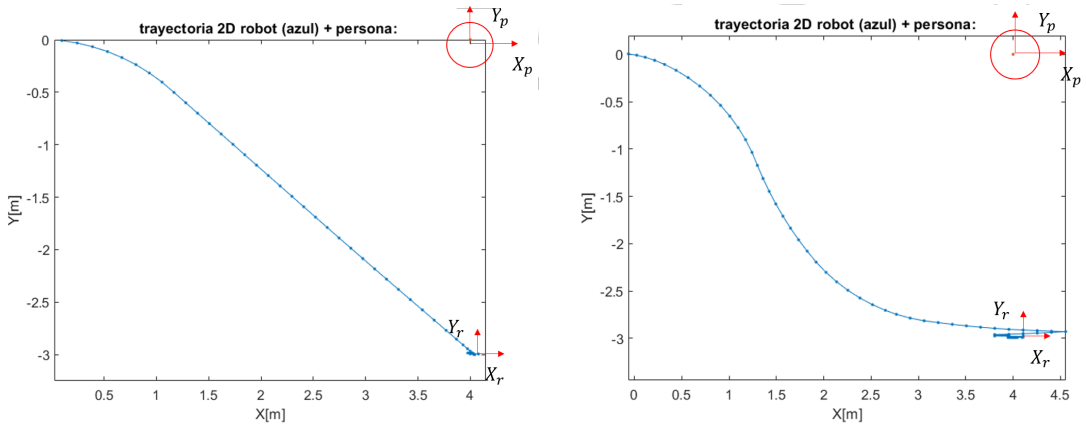
(b) Configuración "Ángulos".



(c) Configuración "Ángulo PTZ".

Figura 5.1: Trayectorias. Objetivo delante.

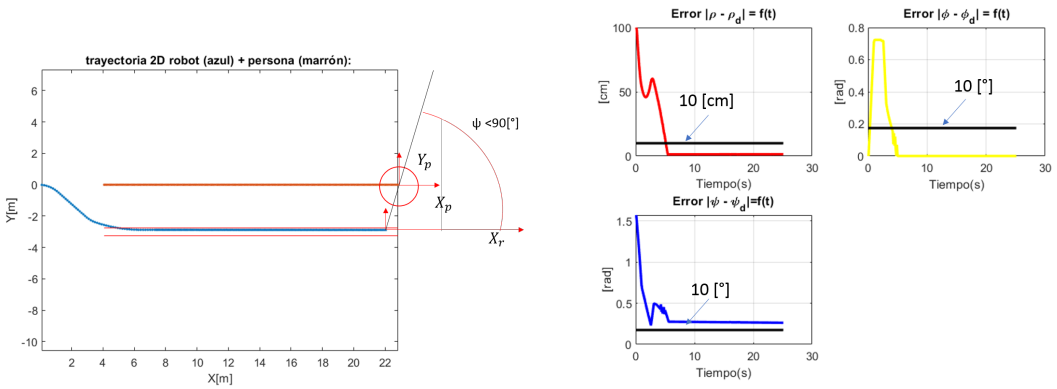
Las únicas configuraciones que proporcionan un buen resultado son "Alineación" y "Equilibrados". En todos los casos, el grupo "Alineación" proporciona menores aceleraciones tanto lineales como angulares debido a que su trayectoria es más recta al darle más peso al ángulo ϕ, ver Figura 5.2.



(a) Configuración "Alineación". (b) Configuración "Equilibrados".

Figura 5.2: Trayectorias. Objetivo derecha.

Sin embargo, al contrario que con el sujeto inmóvil, el grupo "Alineación" no supera el umbral de error cuando se quiere enfocar a la persona desde los laterales debido a que se prioriza que el robot tenga la misma dirección que la persona ignorando el error en el ángulo ψ del PTZ, ver Figura 5.3.



(a) Trayectoria. (b) Errores.

Figura 5.3: Configuración "Alineación". Objetivo derecha.

Por tanto, las dos configuraciones que proporcionan resultados satisfactorios en general son "Alineación" y "Equilibrados". Destacar que el control da mejores resultados si se aumenta el peso del segundo término C_a ("Alineación") en lugar del tercero C_e ("Ángulo PTZ") puesto que el ángulo ϕ es controlado directamente por MPC y el ángulo ψ por el PTZ.

5.3.2. Comparación con la persona en movimiento curvilíneo

La persona se mueve con una velocidad lineal de 2.7 [km/h] y durante el recorrido realiza una media vuelta y un cuarto de vuelta de radio 2.5[m]. En este caso se deben

mostrar los resultados de ambos perfiles ya que los giros de la persona provocan cambios en la trayectoria, Ver Tabla 5.4.

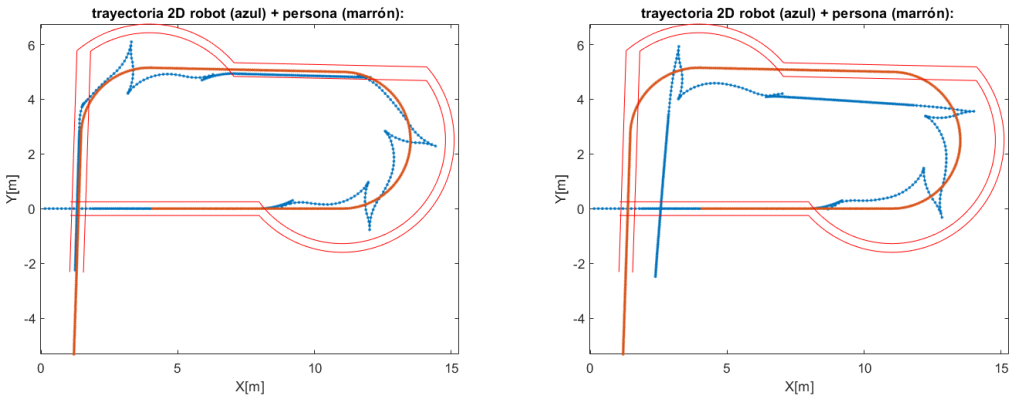
Pto objetivo	Configuración pesos	$\%P_c$	$\%A_{lin}$	$\%A_{ang}$
Detrás	Equilibrados	19.51	93.79	92.68
	Distancia	50.78	93.79	94.01
	Alineación	19.29	90.69	89.80
	Ángulo PTZ	51.88	82.70	86.70
	Ángulos	18.85	64.97	54.99
Derecha	Equilibrados	72.95	89.58	85.14
	Distancia	58.98	94.01	88.03
	Alineación	NaN	92.90	82.93
	Ángulo PTZ	30.59	87.80	73.17
	Ángulos	9.98	98.89	97.56
Izquierda	Equilibrados	87.58	67.40	65.41
	Distancia	84.03	68.51	74.06
	Alineación	53.65	96.45	96.23
	Ángulo PTZ	73.39	68.29	63.41
	Ángulos	87.14	98.89	97.56
Delante	Equilibrados	19.73	94.01	92.24
	Distancia	NaN	79.60	72.28
	Alineación	39.25	94.90	95.12
	Ángulo PTZ	5.99	88.69	85.81
	Ángulos	12.19	94.90	95.34

Tabla 5.4: Sujeto con movimiento curvilíneo y robot noholonomo.

A diferencia de lo que ocurre con los demás conjuntos de simulaciones, en este caso no existe una configuración de valores para los pesos que sea claramente mejor, ya que el resultado depende mucho del punto desde el que se enfoca a la persona.

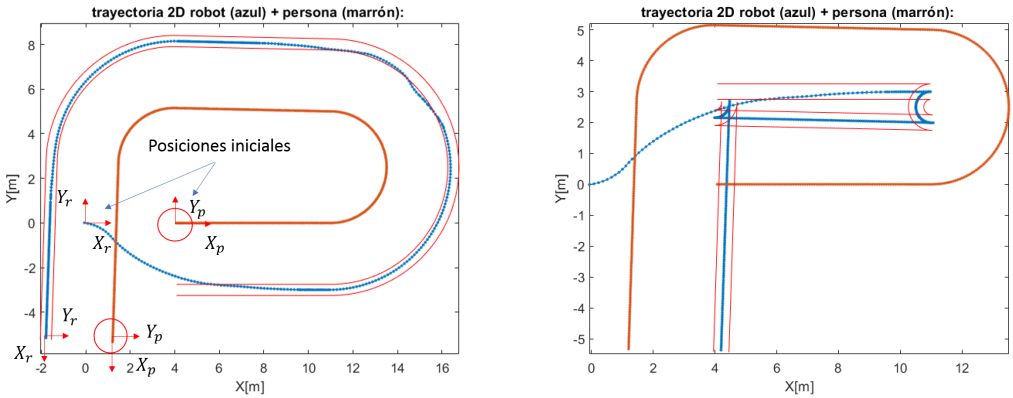
Si se enfoca desde atrás o delante, debido a las propias restricciones de movilidad que presenta la dinámica noholónoma, se obtienen porcentajes bajos en el indicador $\%P_c$. Esto es debido a que en las curvas, la dirección natural de movimiento del robot es contraria al ángulo de enfoque deseado. Entre dos giros de la persona consecutivos el robot debe realizar tres maniobras: el giro con el que orienta las ruedas hacia el nuevo punto de enfoque, el propio desplazamiento y finalmente el giro final con el que se alinea de nuevo con la persona, ver Figura 5.4. En cambio, si el punto objetivo del enfoque son los laterales, la configuración "Equilibrados" destaca con un 87 % desde el perfil derecho y un 73 % desde el izquierdo, ver Figura 5.5. Si se descontase el tiempo que tarda por vez primera en alcanzar la posición objetivo, los porcentajes serían mucho más elevados ya que en las veces sucesivas los errores casi nunca suben por encima del umbral, ver Figura 5.6. Esto es debido a que al ser el PTZ el que gira, en este caso 90[º],

para enfocar a la persona la dirección del robot si que concuerda en todo momento con la dirección del desplazamiento requerido.



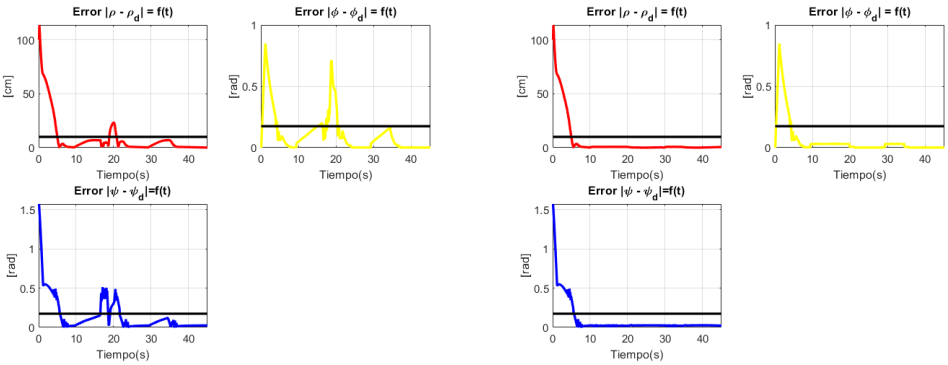
(a) Configuración “Distancia”. (b) Configuración “Equilibrados”

Figura 5.4: Trayectorias. Objetivo Detrás.



(a) Perfil derecho. (b) Perfil izquierdo.

Figura 5.5: Trayectorias. Configuración “Equilibrados”. Objetivo Perfiles.



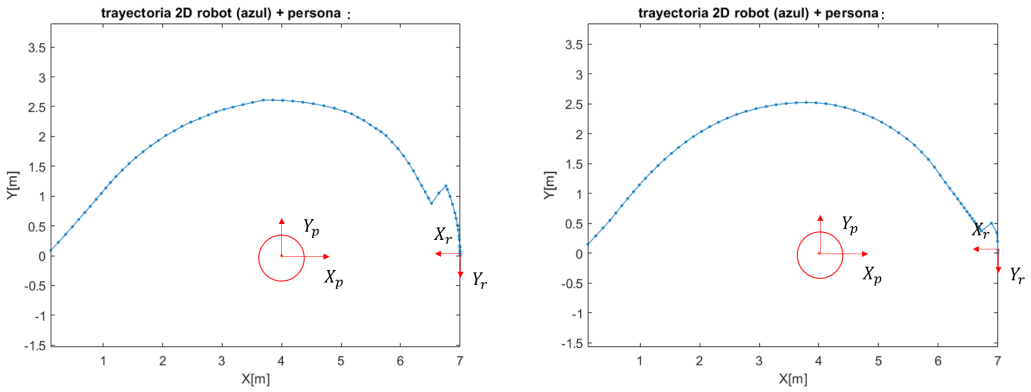
(a) Perfil derecho. (b) Perfil izquierdo.

Figura 5.6: Errores. Configuración “Equilibrados”. Objetivo Perfiles.

Antes de finalizar esta sección señalar que tanto la distancia relativa inicial como el ángulo entre el robot y la persona no influyen en gran medida en los resultados finales. Cuanto más alejados estén dichos valores de los deseados, para cualquier configuración de pesos, el robot necesitará más tiempo para llegar al régimen permanente y realizará más maniobras. Además, si se comparan lo índices entre las distintas configuraciones de pesos variando las condiciones iniciales se mantienen las mismas conclusiones descritas en esta sección.

5.4. Resultados del robot holónomo

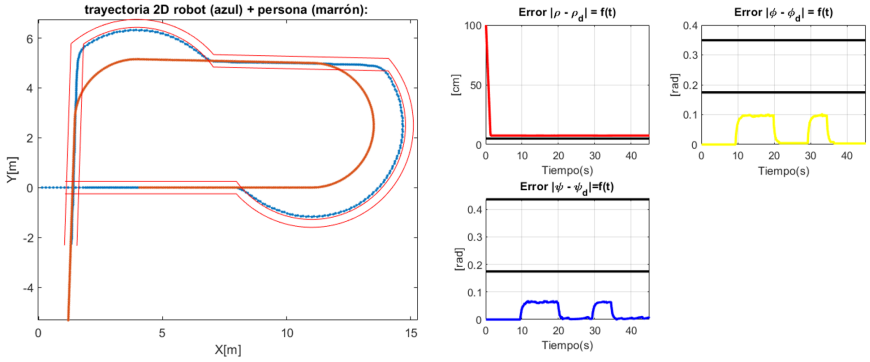
Se han realizado algunos de los experimentos con la dinámica holónoma para observar y comparar la influencia de la dinámica del robot en los resultados. Al proporcionar la dinámica holónoma una total libertad de movimiento, si el control se ha creado correctamente, el valor de los pesos y las condiciones iniciales no influyen en gran medida en el resultado. A modo de ejemplo, la Figura 5.7 muestra la trayectoria que sigue el robot cuando la consigna consiste en ver a la persona inmóvil desde delante con las configuraciones de pesos "Distancia" y "Ángulos".



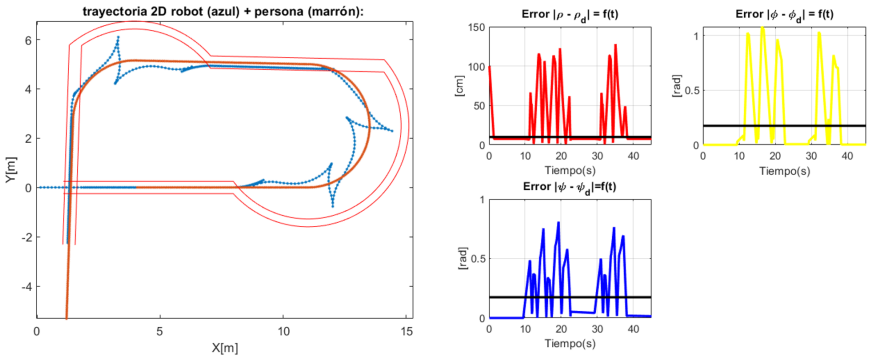
(a) Configuración "Distancia". (b) Configuración "Ángulos".

Figura 5.7: Trayectorias. Objetivo delante.

Además, para mostrar la importancia de la dinámica y la complejidad que supone el robot noholónmo, en las Figuras 5.8 (a) y 5.8 (b) se muestran las trayectorias y los errores con ambas dinámicas manteniendo las mismas condiciones iniciales y los mismos puntos objetivo pero escogiendo la configuración que proporciona mejores resultados para la dinámica noholónoma y la peor para la holónoma.



(a) Configuración "Ángulo PTZ". Dinámica holónoma.



(b) Configuración "Distancia". Dinámica noholónoma.

Figura 5.8: Objetivo detrás. Trayectoria y errores.

Capítulo 6

IMPLEMENTACIÓN EN *ROS*

En este capítulo se explica la implementación de la estrategia de control en un dispositivo real. Esta implementación se ha realizado en dos partes. Por un lado, se ha puesto a prueba el control geométrico en un PTZ real obteniendo la información necesaria con la cámara *realsense* de *Intel*. Por otro lado, se han implementado conjuntamente el control del PTZ con el control del robot simplificando los aspectos de percepción.

6.1. Software *ROS*

Para realizar la implementación se utiliza la plataforma *ROS* (Robot Operating System), creada para dar soporte al desarrollo de software de robots a través de distintas herramientas y bibliotecas. La característica principal de *ROS*, por la cual se ha escogido para realizar la implementación, es la facilidad de desarrollo de entornos para robots gracias a la utilización de herramientas y entornos previamente creados por otros usuarios. Su función principal es administrar todo el intercambio de información entre los diferentes partes que componen el sistema, ver Figura 6.1. La arquitectura interna de *ROS* está formada por:

1. Nodos: procesos que ejecutan cálculos y gestionan la información que se recibe y emite. Por ejemplo, cada dispositivo utilizado, como la cámara o el PTZ, tiene su propio nodo. Los nodos de *ROS* se pueden escribir en diferentes lenguajes de programación, en el TFM se ha optado por utilizar *Python*.

2. Topic: enlace que conecta dos o más nodos permitiendo el intercambio de información. Cada nodo debe indicar a qué topic está conectado y qué tipo de mensaje va a enviar o recibir a través de éste.

3. Mensajes: son estructuras de datos que se envían a través de publicaciones en los topics. Los mensajes pueden contener datos de tipo integer, float, arrays, etc.

ROS tiene integrado un conjunto de tipos de mensajes propios.

4. Elementos externos: Extensiones de *ROS* utilizadas para ampliar su funcionalidad, por ejemplo, mediante un vínculo con *Matlab*.

5. Herramientas de simulación, visualización de resultados, control de robots, etc.

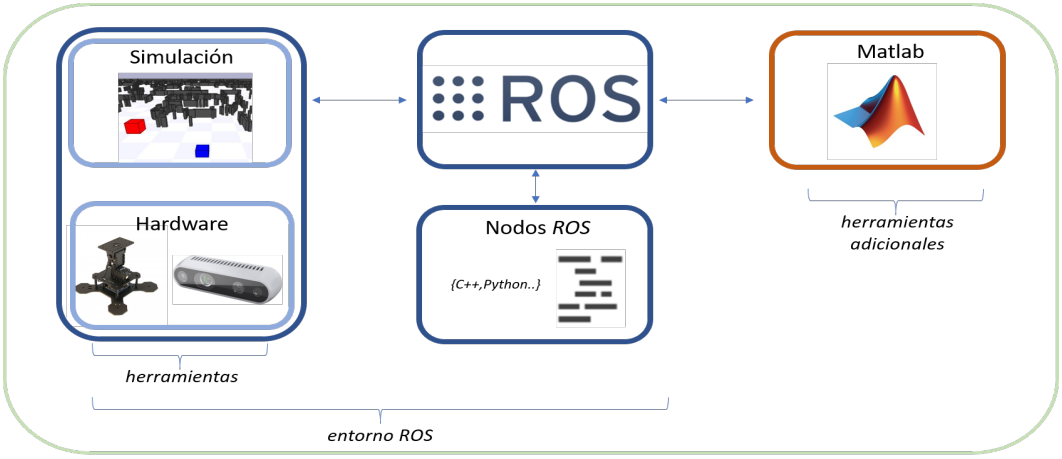


Figura 6.1: Esquema simplificado de nodos *ROS*.

ROS también juega un papel fundamental para facilitar las tareas relacionadas con el funcionamiento de los robots, ya que facilita el control de bajo nivel estos, el intercambio de mensajes entre los diferentes componentes, la abstracción del hardware, el control de los diferentes actuadores, etc. Todo esto permite dividir el problema en tareas más pequeñas programadas en distintos nodos. Por último destacar que se pueden añadir diferentes herramientas con distintos objetivos como la realización de simulaciones. Estas herramientas las puede crear el usuario o utilizar las ya existentes en el propio software *ROS*. Algunas de ellas son:

- Herramientas de simulación de entornos de robots, como *Stage*, capaz de crear un mundo virtual para simular diferentes robots con sus respectivos sensores y actuadores.

- Herramientas de visualización de los robots, como *RViz*, que permite la visualización en 3D de datos generados por los sensores.

- Software para el control de robots, cámaras u otros actuadores. Normalmente desarrolladas por los propios fabricantes de los robots para facilitar el desarrollo y uso de sus productos.

6.2. Implementación del control del PTZ con cámara

Los dispositivos que se van a utilizar son el PTZ *arbotix turret Phantom X* y la cámara *Realsense D435i* de *Intel*. Con ellos se pretende corroborar el correcto funcionamiento del control geométrico para ubicar un punto en el centro de la imagen. El problema de detección de la persona en la imagen se va a simplificar pidiendo directamente el píxel a centrar al usuario por terminal.

La arquitectura de *ROS* para el control se muestra en la Figura 6.2, compuesta por cuatro nodos. Dos de ellos gestionan los dispositivos físicos, nodo *Arbotix* y nodo *Realsense*, proporcionados por los fabricantes, facilitan la comunicación con la cámara y el PTZ. Los otros dos nodos se han desarrollado en el TFM, nodo *Cámara* y nodo *PTZ*. El primero se comunica a su vez con la cámara para obtener los datos necesarios de las imágenes, y el segundo con el nodo *Arbotix*, cuya función será calcular las acciones de los servos y transmitírselas.

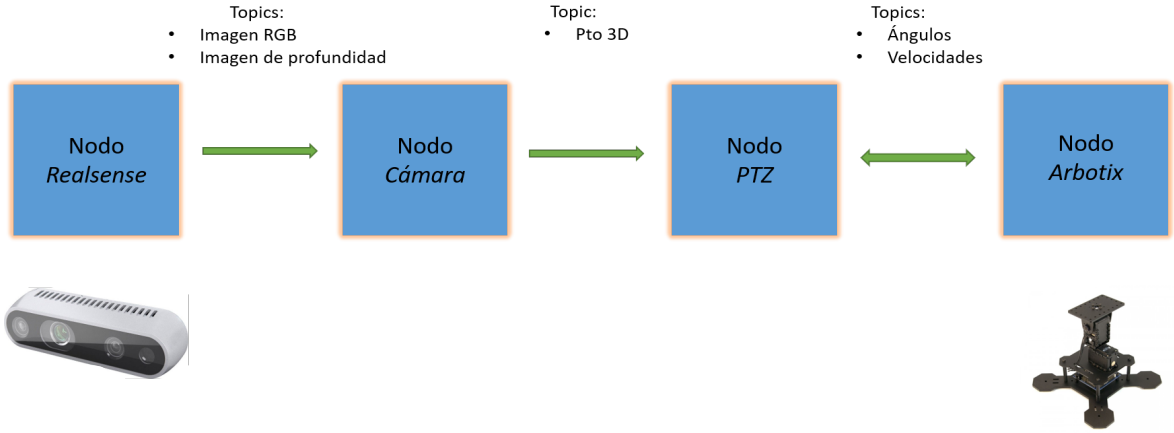


Figura 6.2: Arquitectura de las comunicaciones en *ROS*.

6.2.1. Nodo *Cámara*

El objetivo del nodo *Cámara* es comunicarse con el nodo *Realsense* para obtener las imágenes de profundidad y color, calcular las coordenadas 3D del píxel seleccionado en la imagen por el usuario y enviar esta información al nodo *PTZ*.

El nodo, con una frecuencia predeterminada y suficientemente rápida para formar un vídeo, obtiene de la cámara a través de un topic la imagen RGB en cada instante, dibuja un círculo azul para indicar el centro y se la muestra al usuario. Las imágenes

proporcionadas son de 640x480 píxeles, ver Figura 6.3 (a). El usuario escoge el píxel objetivo que quiere centrar en la imagen. Con ese dato disponible se crea otra ventana donde se congela la imagen y se indica, además del centro y con un círculo negro, el píxel a centrar, ver Figura 6.3 (b). Después, el nodo recibe la imagen de profundidad

(a) Imagen mostrada al usuario. (b) Imagen congelada con píxel objetivo.

Figura 6.3: Imagen cámara *realsense*.

a través de otro topic y, aplicando el modelo de la cámara pinhole expuesto en el Capítulo 2, obtiene el punto 3D cuya proyección en la imagen coincide con el píxel seleccionado. En realidad, se realiza un promedio de la profundidad de los píxeles que se encuentran alrededor del píxel seleccionado, para tener en cuenta la posibilidad de que el píxel elegido carezca de valor de profundidad, debido al propio sensor de profundidad de la cámara. Adicionalmente, con la profundidad obtenida mediante un promedio se suavizan los posibles valores espurios y se obtienen mejores resultados. Por último, mediante otro topic se le envía al nodo *PTZ*, con un array, el punto 3D.

Destacar que una alternativa ya implementada en la propia cámara *realsense* es obtener directamente la nube de puntos 3D de la imagen sin necesidad de realizar los cálculos anteriores, sin embargo, es descartada por el alto coste computacional asociado a trabajar con la nube de puntos. Además, el valor de la profundidad puede ser no calculable en algunos píxeles lo que provoca que el orden en el que se almacenan los datos de la nube de puntos no coincida con el número del píxel en la imagen.

6.2.2. Nodo *PTZ*

El objetivo del nodo *PTZ* es, a partir de las coordenadas 3D del píxel objetivo, calcular los ángulos de giro del PTZ con el desarrollo matemático del Capítulo 2 para centrar el píxel seleccionado en la imagen.

El nodo tiene seis conexiones con el nodo *Arbotix* mediante las que transmite velocidades y posiciones angulares para ambos servos y recibe las posiciones actuales de los mismos. Además tiene una conexión con el nodo *cámara* de donde toma las coordenadas (X,Y,Z) del punto objetivo, ver Figura 6.4.

Cada vez que el nodo recibe un array con las coordenadas del punto, éste obtiene los ángulos de giro actuales de los servos, publicados por el nodo *Arbotix* en dos topics diferentes. Seguidamente, se calculan los ángulos incrementales de giro utilizando las ecuaciones desarrolladas en el Capítulo 2. Antes de efectuar el giro, el nodo *PTZ* le transmite al PTZ la velocidad angular de ambos servos. Finalmente, los ángulos calculados se envían al nodo *Arbotix* para que los ejecute con las velocidades deseadas.

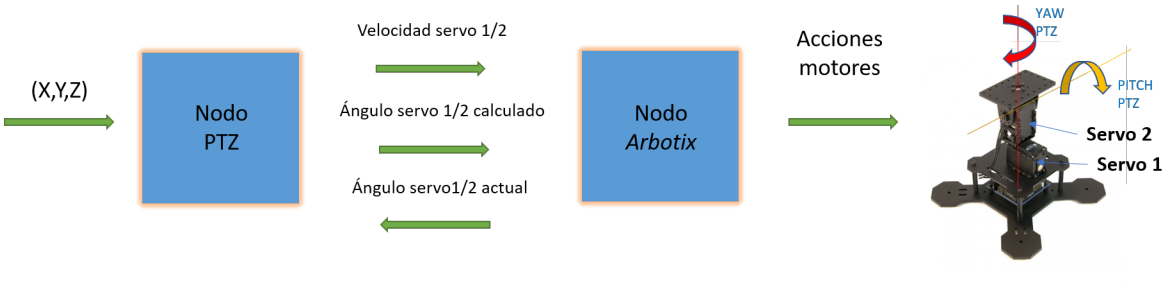


Figura 6.4: Comunicaciones del nodo *PTZ*

La velocidad de los servos se mantiene fija a 1[rad/s] pero se podría modelar en función del valor del ángulo de giro. Además, el código del nodo se ha implementado de tal manera que puede recibir la orden de centrar otro punto antes de que finalice el movimiento para centrar el anterior, característica necesaria para evitar que el objeto a seguir se salga de la imagen ante movimientos rápidos.

6.2.3. Resultados

Para concluir, se exponen algunas de las pruebas realizadas para conocer la precisión con la que se centra el píxel objetivo de la imagen. El centro de la imagen se encuentra en las coordenadas (320,240) [píxeles].

El error de posicionamiento se calcula en coordenadas de la imagen en píxeles, por separado en ambos ejes y en porcentaje del número de píxeles máximo de cada eje,

$$Error_x = \frac{x - 320}{640}, \tag{6.1}$$

$$Error_y = \frac{y - 240}{480}, \tag{6.2}$$

donde la coordenada x representa las columnas y la y las filas de la imagen.

Se van a calcular los errores en cuatro pruebas, ver Figura 6.5 y Tabla 6.1. Para cada prueba se muestran dos imágenes. La imagen izquierda representa el instante una vez elegido el píxel objetivo, redondeado en negro, y antes de girar el PTZ. La imagen de la derecha muestra, una vez aplicado el control, dónde se ha posicionado el centro de la imagen, redondeado en azul. Se han escogido píxeles en los cuatro cuadrantes de la foto y con distinta lejanía para apreciar si existe alguna diferencia en el resultado.

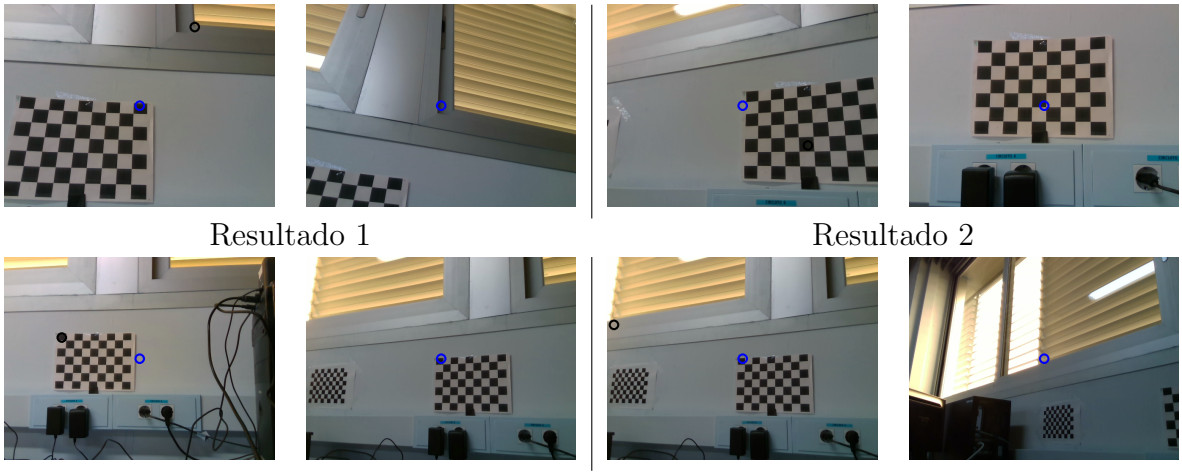


Figura 6.5: Resultados del control geométrico del PTZ. En todos los casos el objetivo es conseguir que el píxel marcado en negro acabe en el centro de la imagen.

Coordenada	Coordenadas iniciales	Coordenadas finales	Error [píxeles]	Error %
Resultado 1				
x	450	349	29	4.5
y	54	256	16	3.3
Resultado 2				
x	474	316	-4	0.62
y	334	232	-8	1.67
Resultado 3				
x	136	317	-3	0.47
y	190	247	7	1.46
Resultado 4				
x	16	311	-9	1.4
y	160	263	23	4.8

Tabla 6.1: Error en el centrado del píxel objetivo.

Como conclusión cabe decir que los errores no superan el margen del 5 %, siendo mayores cuanto más cerca está la cámara del punto objetivo y cuanto más alejado del centro de la imagen se encuentre el píxel seleccionado. No obstante, se considera

que el error obtenido en todas las pruebas es suficientemente pequeño para considerar adecuada la implementación para su uso en la tarea de percepción.

6.3. Implementación del control *MPC* en el robot

El siguiente apartado se centra en la implementación del control del robot. Esta implementación se ha realizado en dos etapas, simulación realista con la herramienta *Stage* y utilización del robot real. En ambos casos se utiliza un segundo robot para simular la posición de la persona real.

6.3.1. Simulación con *Stage*

En primer lugar se describen los nodos que conforman la arquitectura en *ROS*. Para ello se va a realizar un seguimiento de la información que requiere el nodo central, al que designaremos como *MPC-Matlab*, donde se implementa el control óptimo y de la información que transmite. Seguidamente se van a exponer los resultados obtenidos.

Arquitectura en *ROS*

La información requerida por el nodo *MPC-Matlab* es la posición y orientación globales de la persona y el robot terrestre. El nodo internamente transforma los valores globales a relativos entre el robot y sujeto y calcula el control óptimo, seguidamente se comunican las acciones al simulador *Stage* y se visualiza el resultado con *RViz*.

El nodo *MPC-Matlab*, donde se calcula el control, se crea en *Matlab* a partir del código ya utilizado para la simulación del capítulo anterior. Sin embargo, para su utilización en la plataforma real es necesario realizar algunas adaptaciones y modificaciones, debido principalmente a latencias del hardware real. Por un lado, se ha modificado el código de *Matlab* para que funcione como un nodo y se pueda comunicar con *ROS*. Mediante mensajes, el nodo de *Matlab* obtiene la posición y orientación absoluta, recalcula los valores relativos y aplica el control óptimo, seguidamente envía las acciones a otros nodos encargados del control de bajo nivel ya implementados que se encargan de ejecutarlas en *Stage*. Puesto que *MPC* calcula trayectorias del robot en un horizonte de tiempo, para la ejecución del control óptimo en la plataforma real se plantean dos alternativas:

1. Transmitir directamente las acciones calculadas por *MPC*, velocidad angular y lineal, correspondientes al primer paso del horizonte temporal.

2. Transmitir la trayectoria calculada por *MPC* durante el horizonte temporal preestablecido.

En el primer caso mediante mensajes se comunica directamente el nodo *MPC-Matlab* con el simulador *Stage*. Sin embargo, si se transmite la trayectoria se hace uso de un nodo intermedio designado como, *seguidor-trayectorias*, ya implementado que crea las consignas de velocidad a partir de la trayectoria calculada por *MPC* durante el horizonte temporal. Ver Figura 6.6.

Finalmente, tanto para la visualización con *RViz* como para la obtención de la posición y orientación absolutas robot/persona, se van a utilizar nodos ya implementados así como un mapa preestablecido. Para resolver el problema de la localización del robot se utiliza el algoritmo AMCL (Adaptive Monte Carlo Location), que estima la posición y orientación de ambos robots utilizando un filtro de partículas a través de un sensor LiDAR incorporados en los robots simulados (y también reales).

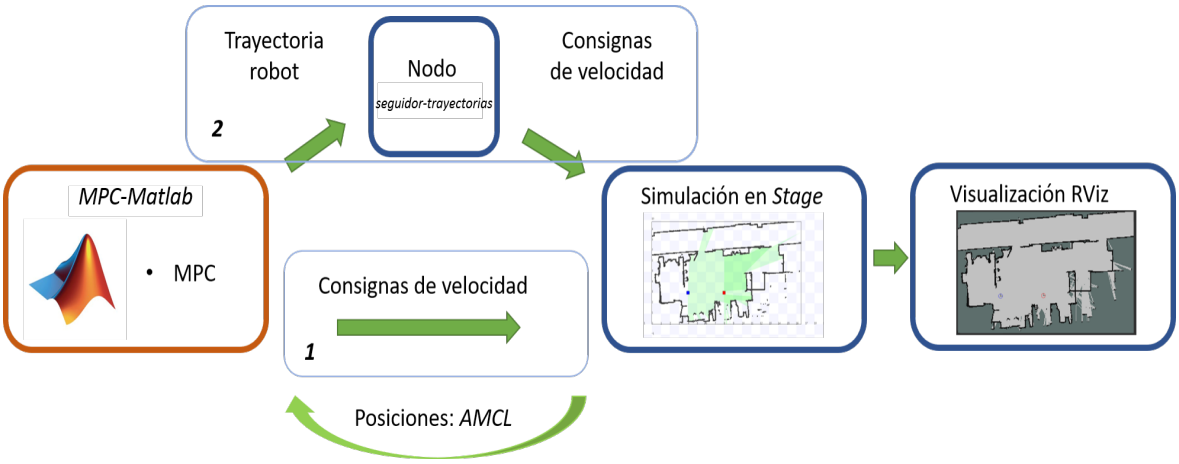


Figura 6.6: Esquema de comunicación *Matlab-ROS*.

Resultados

Los resultados obtenidos con ambas propuestas de ejecución del control son análogos, en la Figura 6.7, se muestra la visualización mediante *RViz* de la posición inicial del robot y la persona junto con el resultado final imponiendo una distancia deseada de 2[m] y enfoque desde el perfil izquierdo.

Transmitir las acciones es la opción más directa ya que se evita el uso del nodo *seguidor-trayectorias*, pero debido a la velocidad de cálculo finita puede haber periodos en los que el robot no avance al no recibir ninguna consigna de velocidad. La segunda

(a) Inicio simulación. (b) Resultado final.

Figura 6.7: Posición inicial y final visualizado con RViz.

opción es menos directa, ya que usa un nodo más, pero proporciona movimientos del robot más suaves y continuos ya que *MPC* es capaz de calcular una nueva trayectoria antes de que el robot haya finalizado la anterior. Sin embargo, esta opción presenta otro problema, también asociado a la velocidad de cálculo finita. Se puede dar el caso de que la trayectoria nueva comience por detrás de la posición actual del robot, debido al movimiento del robot paralelamente al cálculo de *MPC*. Esto provoca que el robot quiera retroceder realizando trayectorias de vaivén que en muchos casos no convergen a la posición objetivo , ver Figura 6.8(a).

(a) Aplicando la trayectoria completa. (b) Aplicando el tramo final de la trayectoria.

Figura 6.8: trayectorias calculadas por *MPC*.

Para solucionar este problema, se suprime la parte inicial de la trayectoria calculada por *MPC* en el mensaje, con lo que se asegura que el inicio de la trayectoria esté por delante de la posición del robot, ver Figura 6.8(b). A pesar de que esta implementación es, de alguna manera, contraria a la filosofía del control *MPC*, se obtienen resultados satisfactorios con movimientos menos bruscos del robot. Para el cálculo del control, a *MPC* se le ha dado consigna de que minimice la función de coste en un horizonte temporal de un segundo dividido en diez pasos iguales, el segmento inicial que se suprime de la trayectoria es el correspondiente a los pasos entre el uno y el cinco.

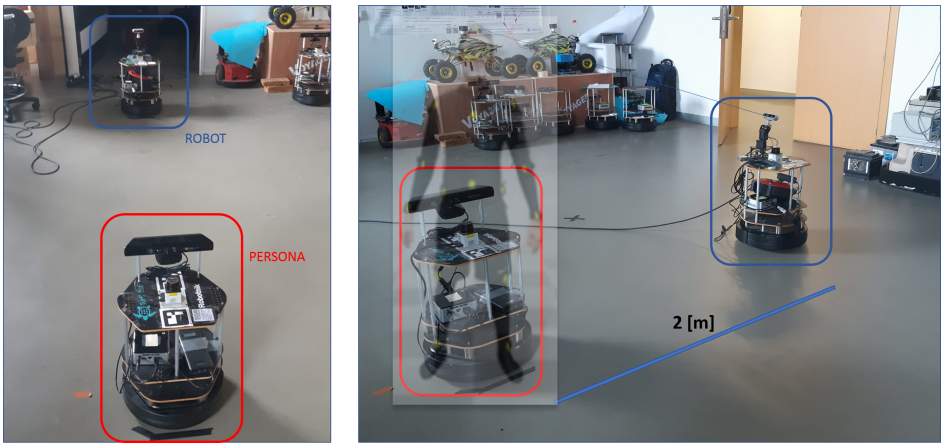
6.3.2. Implementación con dispositivos reales

Una vez que la implementación funciona adecuadamente con Stage, su utilización en la plataforma real es directa. Los cambios que se han tenido que realizar son los siguientes:

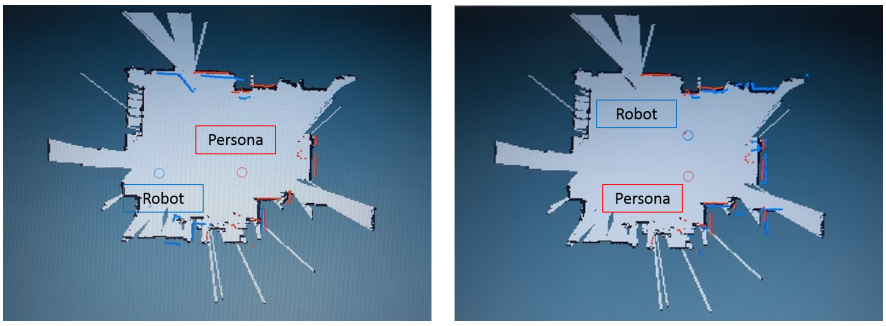
- Añadir el nodo *PTZ* para incluir el control geométrico.
- Redirigir los mensajes de *Stage* al robot real para comunicar las acciones obtenidas con *MPC*.

Como en el caso de la simulación, se utiliza una persona ficticia representada por un robot, esto permite simplificar la tarea de percepción y reconocimiento, ya que el cálculo de la posición relativa se realiza a través de la posición global obtenida con *AMCL* en el mapa del entorno donde se realiza el experimento y con la información de los láseres incorporados en los dos robots, el real y el que simula a la persona.

En la Figura 6.9 se muestra el set up inicial de un experimento así como el resultado final tanto en el visualizador de *RViz* como en fotografías reales.



(a) Visualización real



(b) visualización en *Rviz*

Figura 6.9: Experimentación final con dispositivos reales.

En el experimento se ha dado consigna al robot de que enfoque a una persona ficticia de 1.8[m] desde la izquierda a una distancia de dos metros. Como se puede observar el resultado final es satisfactorio ya que el robot consigue alcanzar la posición objetivo enfocando a la persona con el PTZ, lo que confirma el buen funcionamiento de ambos controles.

Capítulo 7

CONCLUSIONES Y LÍNEAS FUTURAS

En este capítulo se exponen las conclusiones finales y se plantean las líneas futuras de estudio que deja abiertas este TFM.

7.1. Conclusiones

En este Trabajo Fin de Máster se ha desarrollado una nueva estrategia de control orientada a percepción con el objetivo de monitorizar y seguir a una persona con un robot terrestre que tiene incorporado una cámara con PTZ. La propuesta de control está basada en dos controles complementarios y de diferente naturaleza. Por un lado, se ha aplicado un control geométrico al movimiento del PTZ, el cual permite mantener a la persona en el campo de vista independientemente del estado del robot terrestre. Por otro lado, se ha aplicado un control óptimo, *MPC*, al movimiento del robot terrestre, proporcionando el desplazamiento necesario para poder enfocar a la persona desde una perspectiva deseada.

Se han implementado y comprobado ambos controles en entornos de simulación de complejidad creciente. Se ha utilizado *Matlab* para ajustar los parámetros del control óptimo para su correcto funcionamiento en el mayor rango posible de situaciones. En el ajuste se ha priorizado la obtención de trayectorias adecuadas a la dinámica no holónoma del robot. Además, durante la implementación de los controles se ha motivado el desacople en el control de la posición y orientación de la persona en la imagen. También se ha corroborado el correcto funcionamiento de ambos controles mediante experimentos en dispositivos reales utilizando *ROS*.

Como resultado de la realización de este TFM se ha alcanzado el objetivo de monitorización y se ha obtenido un sistema completamente operativo, capaz de realizar

seguimientos de personas que realizan cualquier tipo de trayectorias.

7.2. Líneas futuras

Como posibles líneas futuras de investigación abiertas tras los resultados obtenidos, se proponen las siguientes:

1. Incluir aspectos realistas de percepción de objetivos con técnicas de visión por computador.

2. Plantear una función de coste con transición suave, en comparación con la función en escalón que se ha utilizado en el TFM.

3. Ampliar la estrategia de control para que considere varios robots, incluyendo restricciones de colisión o estableciendo formaciones entre ellos.

Capítulo 8

Bibliografía

[1] G. López-Nicolás, M. Aranda, and Y. Mezouar, "Formation of differential-drive vehicles with field-of-view constraints for enclosing a moving target," in *IEEE International Conference on Robotics and Automation*, pp. 261–266, 2017.

[2] J. D. P. V. Gabriele Costante, Christian Forster and D. Scaramuzza., "Perception-aware path planning.," in *arXiv preprint arXiv:1605.04151,*, 2016.

[3] R. Carelli, C. M. Soria, and B. Morales, "Vision-based tracking control for mobile robots," in *ICAR '05. Proceedings., 12th International Conference on Advanced Robotics, 2005.*, pp. 148–152, 2005.

[4] J. Satake and J. Miura, "Robust stereo-based person detection and tracking for a person following robot," in *ICAR '09. Proceedings., International Conference on Advanced Robotics, 2009.*, 2009.

[5] A. Leigh, J. Pineau, N. Olmedo, and H. Zhang, "Person tracking and following with 2d laser scanners," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 726–733, 2015.

[6] T. Nägeli, S. Oberholzer, S. Plüss, J. Alonso-Mora, and O. Hilliges, "Flycon," *ACM Transactions on Graphics*, vol. 37, pp. 1–14, jan 2019.

[7] "About ros." `https://www.ros.org/about-ros/`.

[8] "Mpc." `https://es.mathworks.com/products/model-predictive-control`.

Anexos

Anexos A

En este anexo se muestra la metodología que se ha seguido para centrar el píxel en la imagen a través de los giros ψ del servo 1, ver Figuras A.2 y A.3 , y θ del servo 2 , ver Figura A.4, para todas las configuraciones.

A.1. Centrado en *Yaw*

Como se ha mencionado en el Capítulo 2, en cada zona se obtienen seis ecuaciones de las cuales dos son diferentes entre zonas. Si no se especifica nada, las ecuaciones servirán para las dos disposiciones de la cámara, hacia arriba y hacia abajo y para todas las zonas.

El ángulo relativo a girar por el servo 1, ψ^*, se calcula como,

$$\psi^* \;=\; \arctan \frac{X_{pc}}{L}, \tag{A.1}$$

siendo L, la distancia en horizontal desde el punto 3D suprimiendo su componente X_{pc} hasta el eje de giro del servo 1, ecuación (A.2) cámara arriba y ecuación (A.3) cámara abajo,

$$L \;=\; L_3 - L_1, \tag{A.2}$$

$$L \;=\; L_3 + L_1. \tag{A.3}$$

La distancia L_1 se calcula utilizando el ángulo de giro actual del servo 2, θ, y la distancia entre los dos ejes de giro, $d_{\theta\psi}$,

$$L_1 \;=\; d_{\theta\psi} \cos\theta. \tag{A.4}$$

Por otra parte, L_3 es la proyección de L_2 en el eje X del servo 1,

$$L_3 \;=\; L_2 \cos\theta_2, \tag{A.5}$$

siendo L_2 la hipotenusa del ángulo formado por las coordenadas (Y_{pc},Z_{pc}) y θ_2 una combinación de los ángulos θ y θ_1.

– Zona 1: ecuación (A.6) cámara orientada hacia arriba y ecuación (A.7) cámara orientada hacia abajo.

$$\theta_2 = \theta + \theta_1 \tag{A.6}$$

$$\theta_2 = \theta_1 - \theta \tag{A.7}$$

– Zona 2: ecuación (A.8) cámara orientada hacia arriba y ecuación (A.9) cámara orientada hacia abajo.

$$\theta_2 = \theta - \theta_1 \tag{A.8}$$

$$\theta_2 = \theta - \theta_1 \tag{A.9}$$

– Zona 3: ecuación (A.10) cámara orientada hacia arriba y ecuación (A.11) cámara orientada hacia abajo.

$$\theta_2 = \theta_1 - \theta \tag{A.10}$$

$$\theta_2 = \theta_1 + \theta \tag{A.11}$$

Finalmente θ_1 es,

$$\theta_1 = \arctan \frac{|Y_{pc}|}{Z_{pc}}. \tag{A.12}$$

el ángulo formado por Z_{pc}, Y_{pc}.

A.2. Centrado en *Pitch*

En este caso es la pendiente de la recta tangente a la circunferencia descrita al girar el servo 2 la que toma diferente valor según la zona en la que se encuentre el punto objetivo.

El ángulo θ^* relativo a girar se calcula como,

$$\theta^* = \arccos \frac{d_{\theta\psi}}{H}, \tag{A.13}$$

siendo H la distancia desde el eje de giro del servo 2 hasta el corte que genera la recta tangente a la circunferencia con el eje Y de dicho servo. La pendiente de la recta tangente a la circunferencia se calcula resolviendo una ecuación de segundo orden,

$$d^2_{\theta\psi} = \frac{(m + Y - mX)^2}{1 + m^2}, \tag{A.14}$$

siendo m la pendiente de la recta tangente y (X, Y) las coordenadas del punto 3D (Y_{pc}, Z_{pc}) respecto de la cámara trasladadas a la renferencia servo 2, ver Figura A.1.

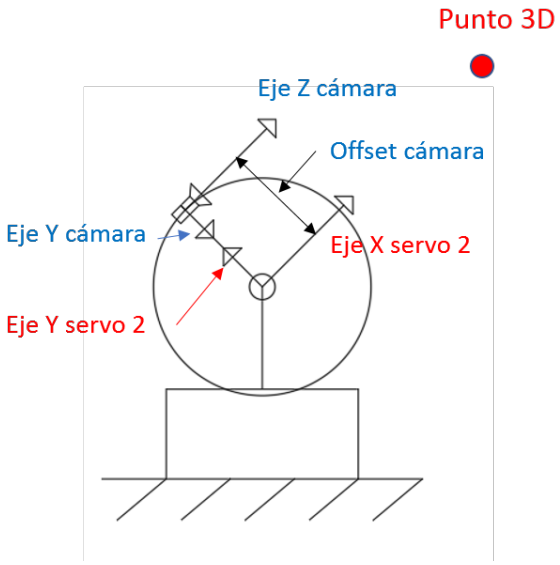
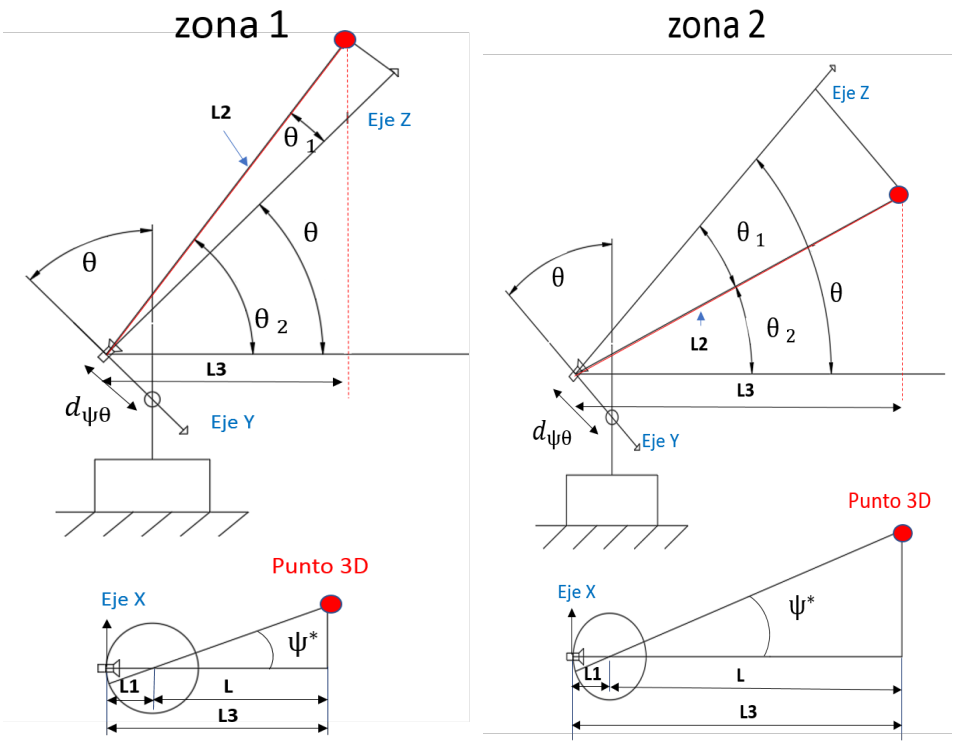


Figura A.1: Cambio de coordenadas desde cámara a servo2.

Al resolver la ecuación (A.14) de segundo grado se obtienen dos pendientes una por cada recta tangente a la circunferencia, el valor de la pendiente y por tanto la recta tangente, se escogerá dependiendo de la zona en la que se encuentre el punto 3D.

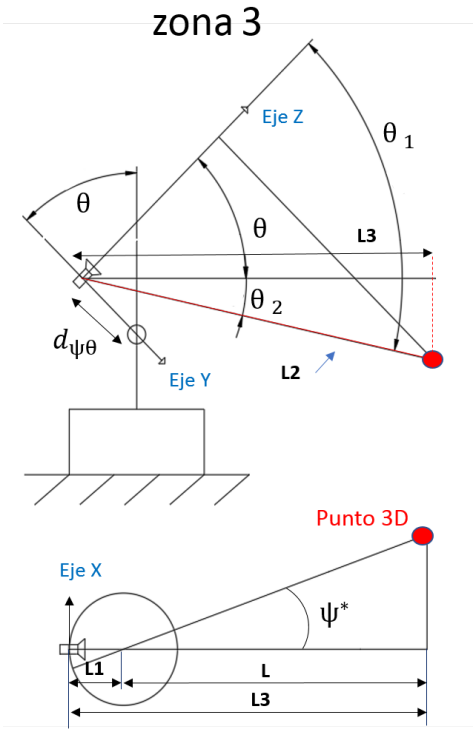
- Zona 1: se escoge la pendiente menor.
- Zona 2: se escoge la pendiente negativa.
- Zona 3: se escoge la pendiente menor.

Finalmente, con la pendiente y las coordenadas del punto objetivo se obtiene al punto de corte H.



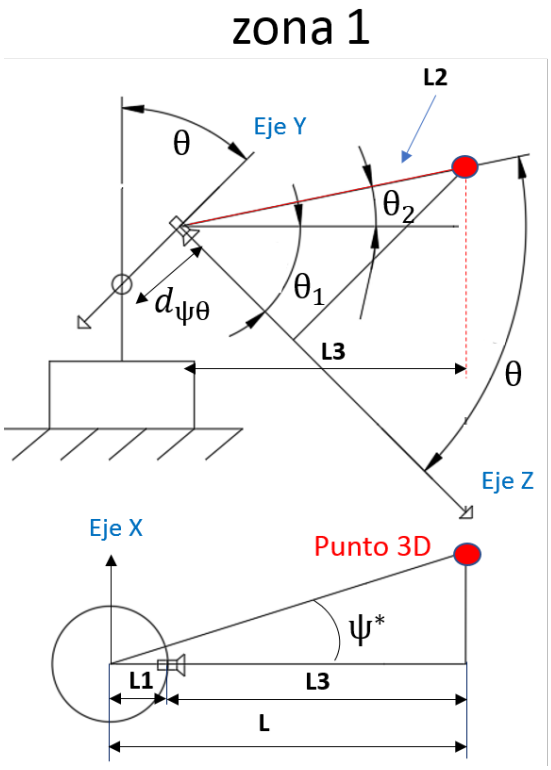
(a) Dibujo cad zona 1

(b) Dibujo cad zona 2

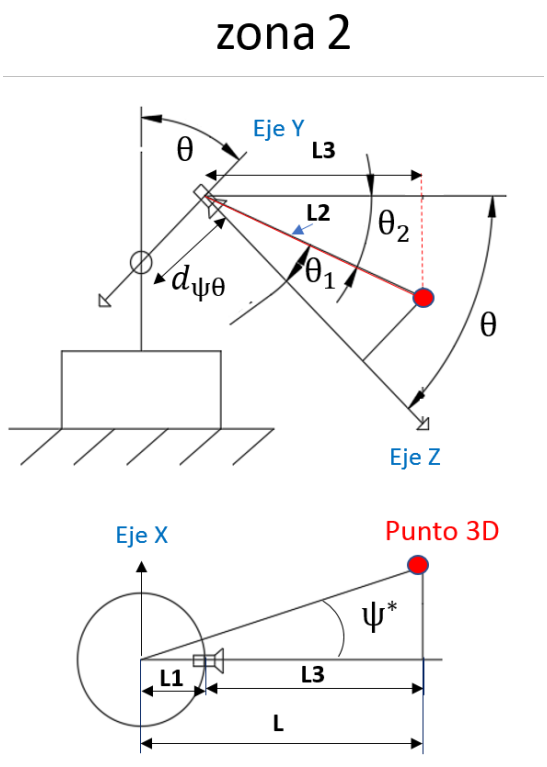


(c) Dibujo cad zona 3

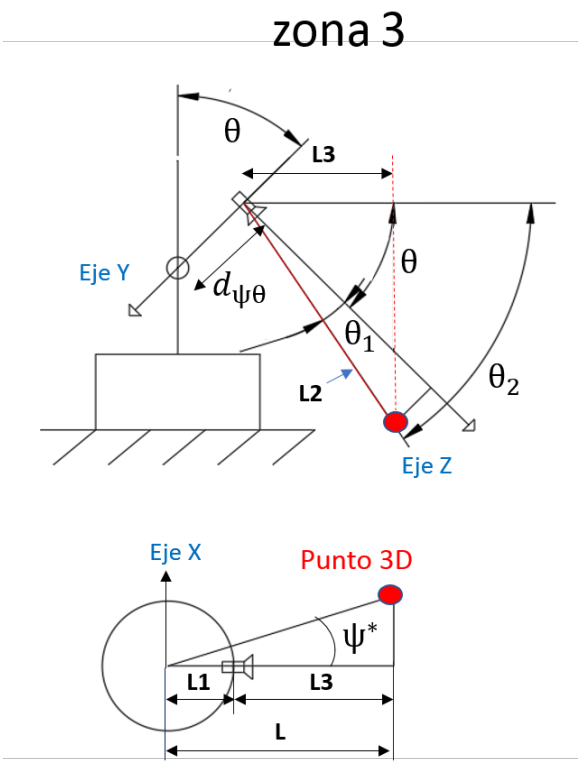
Figura A.2: Centrado del píxel con ψ. Cámara arriba. Planta y Alzado.



(a) Dibujo cad zona 1

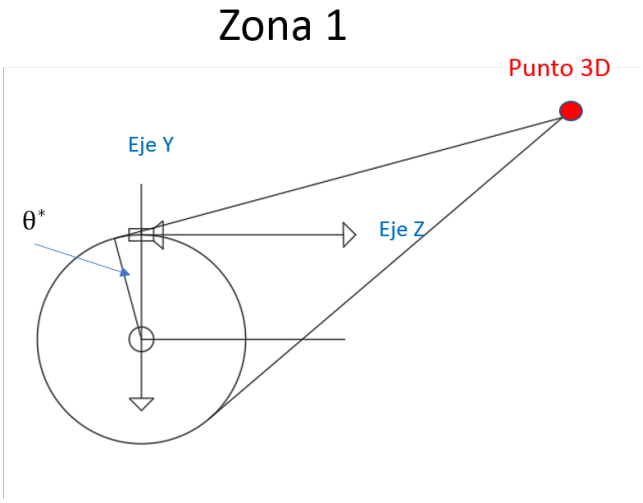


(b) Dibujo cad zona 2

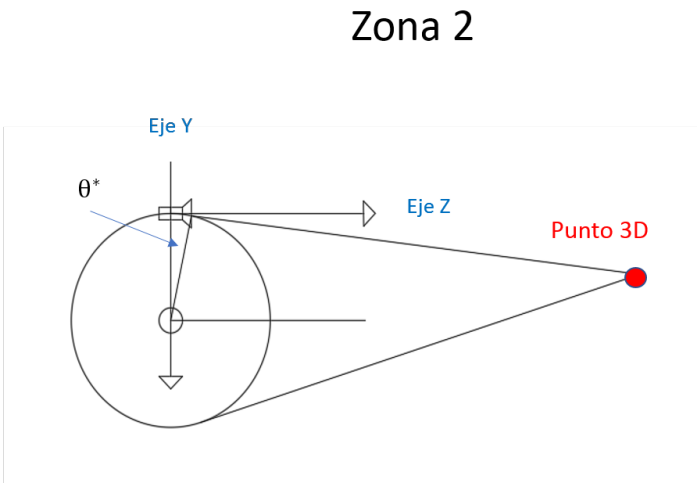


(c) Dibujo cad zona 3

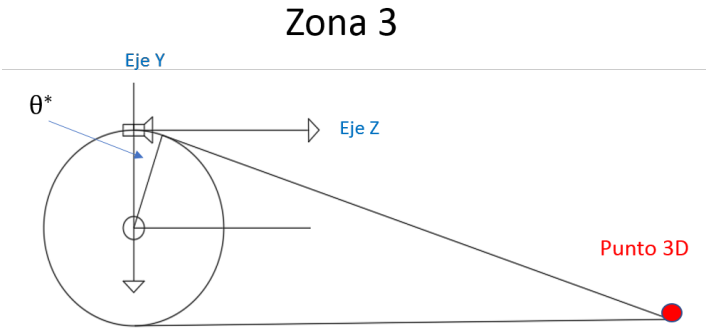
Figura A.3: Centrado del píxel con ψ. Cámara abajo. Planta y Alzado.



(a) Dibujo cad zona 1



(b) Dibujo cad zona 2



(c) Dibujo cad zona 3

Figura A.4: Centrado del píxel con θ. Alzado.